머신러닝 스터디 3rd week 보조 자료

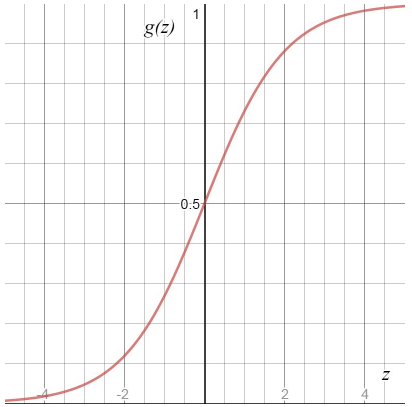
20180119 김성헌

summary

- ch.5 Logistic (regression) classifier
  - logistic hypothesis

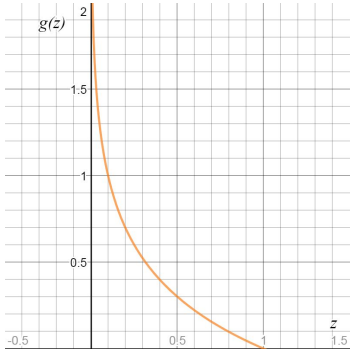
$$H(X) = \frac{1}{1 + e^{-(W^T X)}}$$

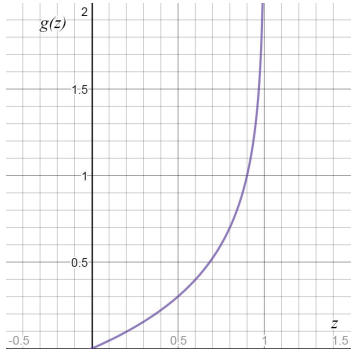
$$g(z) = \frac{1}{(1 + e^{-z})}$$



  - cost

$$cost(W) = -\frac{1}{m}\sum ylog(H(x)) + (1-y)log(1-H(x))$$





  - Gradient descent algorithm

$$W := W - \alpha\frac{\partial}{\partial W}cost(W)$$

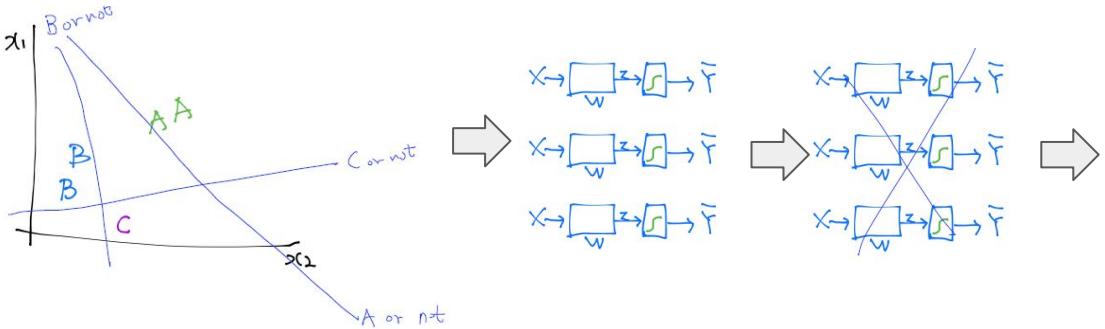
$$c(H(x), y) = \begin{cases} -log(H(x)) & : y = 1 \\ -log(1 - H(x)) & : y = 0 \end{cases}$$

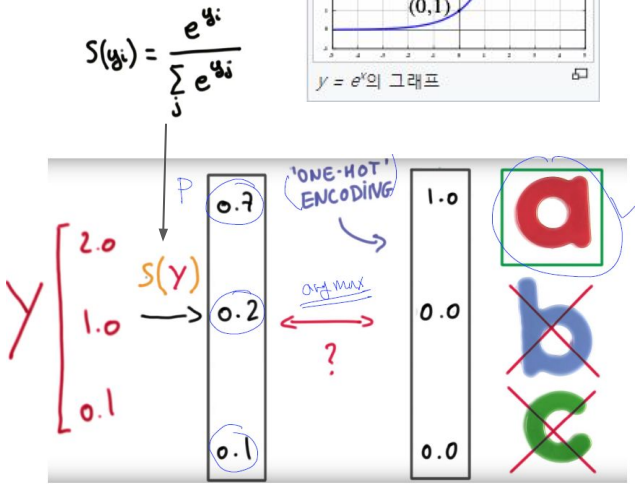
```
# cost function
cost = tf.reduce_mean(-tf.reduce_sum(Y*tf.log(hypothesis) + (1-Y)*tf.log(1-hypothesis)))

# Minimize
a = tf.Variable(0.1) # Learning rate, alpha
optimizer = tf.train.GradientDescentOptimizer(a)
train = optimizer.minimize(cost)
```

summary

- ch.6 Softmax Classifier

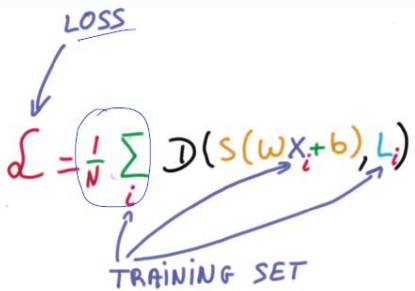


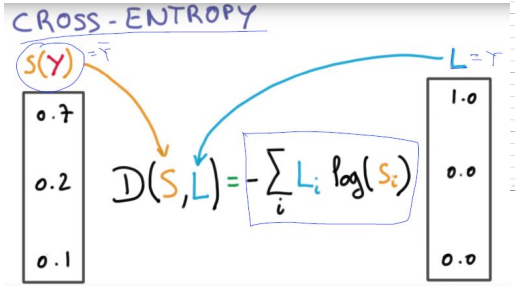


Hypothesis

$S(WX_i+b)$

Cost function





Gradient descent

$-\alpha \frac{\Delta \mathcal{L}(w_1, w_2)}{\text{DERIVATIVE}}$

decision boundary

- tensorflow 사용한 decision boundary
  - https://mubaris.com/2017/10/21/tensorflow-101/
- Neural Network decision boundary
  - http://www.wildml.com/2015/09/implementing-a-neural-network-from-scratch/
- Some Deep Learning with Python, TensorFlow and Keras
  - https://sandipanweb.wordpress.com/2017/11/25/some-deep-learning-with-python-tensorflow-and-keras/