Loss Functions

## MSELoss()

Measure the mean square error (L2 norm) between each element in input x and target y

[](https://www.codecogs.com/eqnedit.php?latex=L(x%2Cy)%20%3D%20L%20%3D%5C%20%7Bl_1%2C...%2Cl_n%5C%7D%5ET%20%2C%20ln%20%3D%20(x_n%20-%20y_n)%5E2#0) where N is the batch size

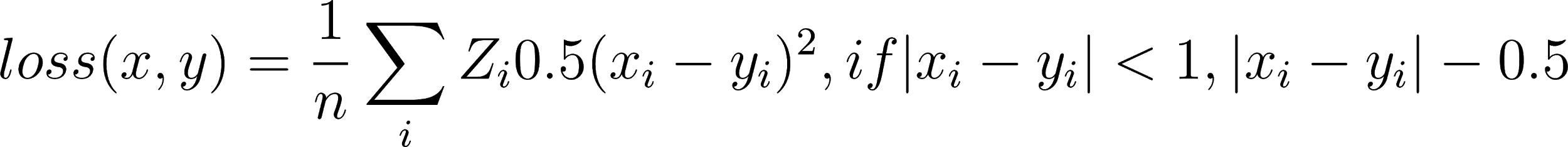
## L1Loss()

Basically the absolute value of the different between desired output and actual output

[](https://www.codecogs.com/eqnedit.php?latex=L(x%2Cy)%20%3D%20L%20%3D%5C%20%7Bl_1%2C...%2Cl_n%5C%7D%5ET%20%2C%20ln%20%3D%20%7Cx_n%20-%20y_n%7C#0) where N is the batch size

Use for robust regression where small error to count more and large not as much to reduce noise in data e.g. to reduce the influent of a few outlier in dataset but L1 loss is not differentiable at the bottom when gradient get closer to the bottom normally is done with soft shrink, some fix of this problem e.g. smooth L1 loss to make bottom smooth like L2 with Hubert loss to protect against outliers

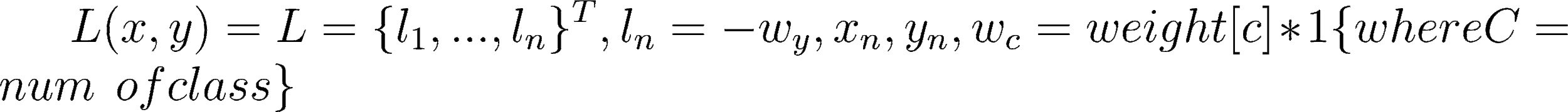
SmoothL1Loss()

[](https://www.codecogs.com/eqnedit.php?latex=loss(x%2Cy)%20%3D%20%5Cfrac%7B1%7D%7Bn%7D%20%5Csum_%7Bi%7D%20Z_i%20%7B%200.5(x_i%20-%20y_i)%5E2%2C%20if%20%7Cx_i%20-%20y_i%7C%20%3C1%2C%7Cx_i%20-%20y_i%7C%20-%200.5#0) otherwise

Note the average value make image blur

NLLLoss()

The negative log likelihood loss useful for reduce discrepancy among dataset

[](https://latex-staging.easygenerator.com/eqneditor/editor.php?latex=L(x%2Cy)%20%3D%20L%20%3D%20%5C%7Bl_1%2C...%2Cl_n%5C%7D%5ET%20%2C%20l_n%20%3D%20-w_y%2Cx_n%2Cy_n%2C%20w_c%20%3D%20weight%5Bc%5D%20*%201%5C%7B%20where%20C%20%3D%20num%20%5Chspace%7B5%7D%20of%20class%5C%7D#0)

Try to pick the correct value in the output vector as large as possible

If there is a negative in front it can be interpret x as energy

Useful for widely different frequency (size) data category this by assign larger weight for the category that have small dataset and smaller weight to category that have large among of data, but better solution is to increase the frequency of minimal category instead by reuse the same set of smaller dataset over until the larger dataset exhaust to equalize them out this method exploit SGD better.Never leave any data unused

## CrossEntropyLoss()

Merging the log softmax(do log after the softmax) and NLLLoss is for numerical reasons

Since the log softmax step might have gradient value infinite in the middle of the step (because some values in softmax might be very small close to 0 value take log of this value result in negative infinity ) and gradient of this value also close to infinity Alway do the softmax and log in the same time to avoid this problem to get a more stable numerical value. By combining these 2 it makes the correct value largest as possible and suppresses all the others scores as small as possible due to normalization. The loss take and X factor and a category a desired category a class then compute the negative log of softmax applied to vector of scores the numerator is the X of the index of the correct class that is the loss the negative log of exponential score of the correct class divided by the sum of the exponential of all the scores. The X can be thought of as negative energy. The negative score of the correct class and to make this value small make the score large and all the sum of exponential of other value small this make the edge small

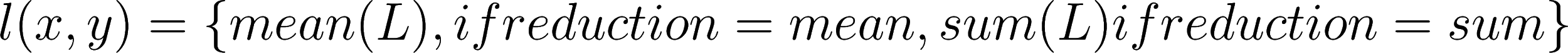
Cross entropy is the cross of distribution of the softmax vector and target value vector (one hot)that have one value that close to 1

AdaptiveLogSoftmaxWithLoss()

Extension to log softmax useful with the dataset that have large categories e.g. in NLP by use some like ignore the small categories and focus on the large one

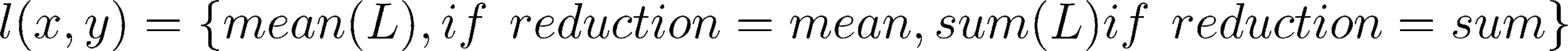
## BCELoss()

[](https://www.codecogs.com/eqnedit.php?latex=L(x%2Cy)%20%3D%20L%20%3D%20%5C%7Bl_1%2C...%2Cl_n%5C%7D%5ET%20%2C%20l_n%20%3D%20-w_n%5By_n%20*%20log%20x_n%20%2B%20(1-y_n)*%20log(1-x_n)%5D#0)

[](https://www.codecogs.com/eqnedit.php?latex=l(x%2Cy)%20%3D%20%5C%7Bmean(L)%2C%20if%20%20reduction%20%3D%20mean%20%2Csum(L)%20if%20reduction%20%3D%20sum%5C%7D#0)

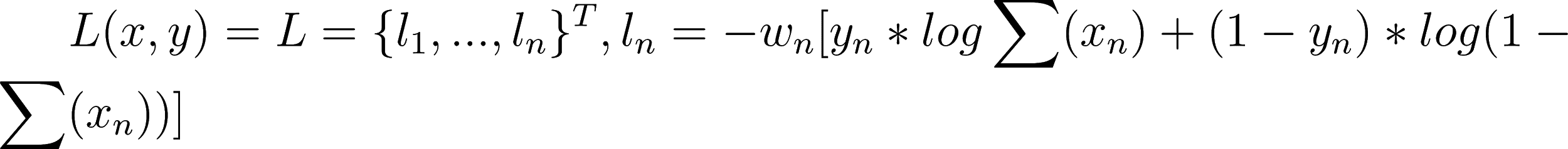
## KLDivLoss()

[](https://www.codecogs.com/eqnedit.php?latex=L(x%2Cy)%20%3D%20L%20%3D%20%5C%7Bl_1%2C...%2Cl_n%5C%7D%20%2C%20l_n%20%3D%20y_n%20*%20(log%20y_n%20-%20x_n)#0) where N span all dimension of input and L have the same shape as input if reduction is not none

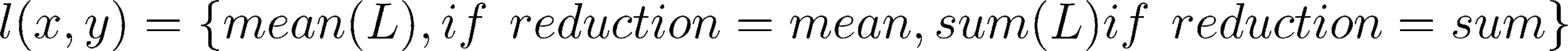
[](https://latex-staging.easygenerator.com/eqneditor/editor.php?latex=l(x%2Cy)%20%3D%5C%7Bmean(L)%2C%20if%20%5Chspace%7B5%7Dreduction%20%3D%20mean%2Csum(L)%20if%5Chspace%7B5%7D%20reduction%20%3D%20sum%5C%7D#0)

May have numerical issue

## BCEWithLogitsLoss()

[](https://www.codecogs.com/eqnedit.php?latex=L(x%2Cy)%20%3D%20L%20%3D%5C%7Bl_1%2C...%2Cl_n%5C%7D%5ET%20%2C%20l_n%20%3D%20-w_n%5By_n%20*%20log%20%5Csum(x_n)%20%2B%20(1-y_n)*%20log(1-%5Csum(x_n))%5D#0)

where N span all dimension of input and L have the same shape as input if reduction is not none

[](https://latex-staging.easygenerator.com/eqneditor/editor.php?latex=l(x%2Cy)%20%3D%20%5C%7Bmean(L)%2C%20if%5Chspace%7B5%7D%20reduction%20%3D%20mean%2C%20sum(L)%20if%20%5Chspace%7B5%7Dreduction%20%3D%20sum%5C%7D#0)

It take value and pass to sigmoid and make sure the value is between 0 and 1

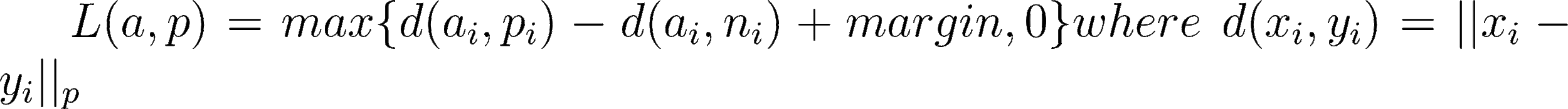
## MarginRankingLoss()

[](https://www.codecogs.com/eqnedit.php?latex=loss(x%2Cy)%20%3D%20max(0%2C%20-y%20*%20(x_1%20-%20x_2)%20%2B%20margin)#0)

Make one of the input larger than the other by value of the margin, y is the value that control x1 and x2

## TripletMarginLoss()

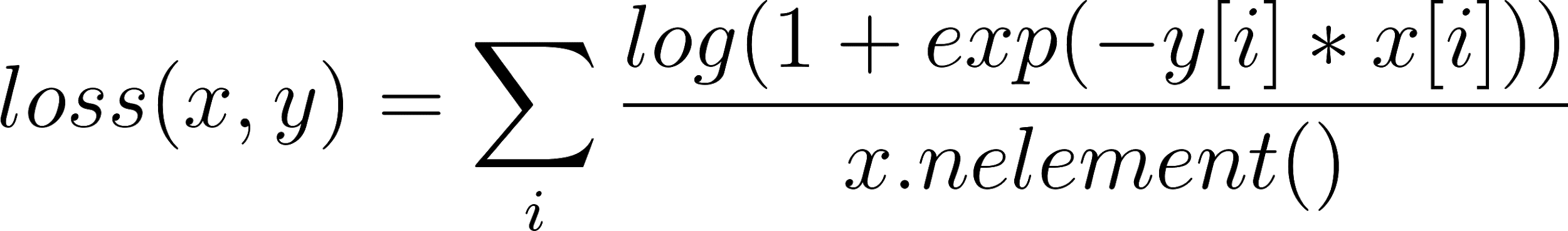
Use to measuring a relative similarity between samples. It compose of anchor,p positive,n negative

[](https://latex-staging.easygenerator.com/eqneditor/editor.php?latex=L(a%2Cp)%20%3D%20max%5C%7Bd(a_i%2Cp_i)%20-%20d(a_i%2Cn_i)%20%2B%20margin%2C%200%5C%7D%20where%5Chspace%7B5%7D%20d(x_i%2Cy_i)%20%3D%20%7C%7C%20x_i%20-%20y_i%7C%7C_p%20%20#0)

Try to get the distance (value ) good pair smaller than the distance of the bad pair

## SoftMarginLoss()

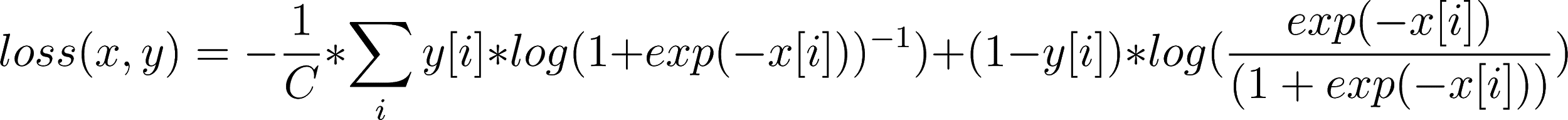
Optimize a two-class classification logistic loss between input tensor x and target tensor y (containing 1 or -1)

[](https://www.codecogs.com/eqnedit.php?latex=loss(x%2Cy)%20%3D%20%5Csum_%7Bi%7D%20%5Cfrac%7B%20log(1%2Bexp(-y%5Bi%5D%20*%20x%5Bi%5D))%7D%7Bx.nelement()%7D#0)

Which the positive value closer and negative value further

## MultiLabelSoftMarginLoss()

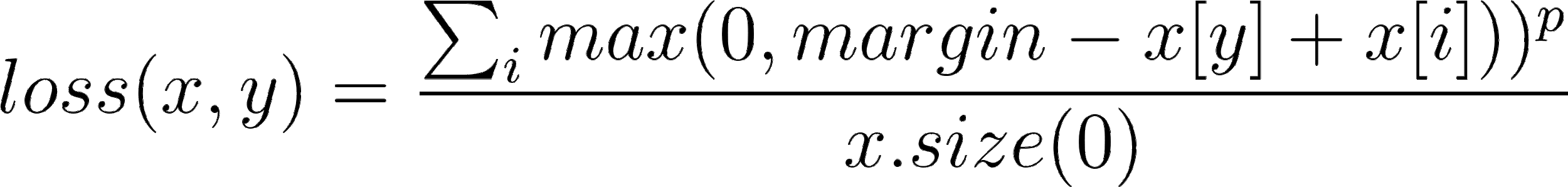
Optimize a multi-label one-versus-all loss based on max-energy, between input x and target y of size (N,C). for each sample in the minibatch

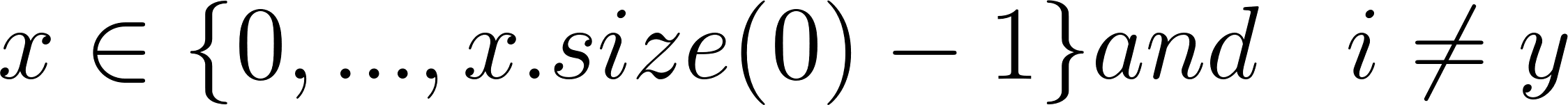
[](https://www.codecogs.com/eqnedit.php?latex=loss(x%2Cy)%20%3D%20-%5Cfrac%7B1%7D%7BC%7D%20*%20%20%5Csum_%7Bi%7D%20y%5Bi%5D%20*%20log(1%2Bexp(-x%5Bi%5D))%5E%7B-1%7D)%20%2B%20(1-y%5Bi%5D)%20*log(%20%5Cfrac%7B%20exp(-x%5Bi%5D)%7D%7B(1%2Bexp(-x%5Bi%5D))%7D%20)#0)

Where [](https://www.codecogs.com/eqnedit.php?latex=i%20%5Cin%20%5C%7B0%2C...%2Cx.nelement()%20-1%5C%7D%2Cy%5Bi%5D%20%5Cin%20%5C%7B0%2C1%5C%7D#0)

Allow multiple correct output (want multiple categories can have high score and all lower score get suppressed) which for the desire category want the value to be higher and the undesired value suppress

## MultiMarginLoss()

[](https://latex-staging.easygenerator.com/eqneditor/editor.php?latex=loss(x%2Cy)%20%3D%5Cfrac%7B%20%5Csum_%7Bi%7D%20%5Cfrag%7Bmax(0%2Cmargin%20-%20x%5By%5D%20%2B%20x%5Bi%5D))%5Ep%7D%7D%7B%7B%20x.size(0)%7D%7D#0)

Where [](https://latex-staging.easygenerator.com/eqneditor/editor.php?latex=%20x%20%5Cin%20%5C%7B0%2C...%2Cx.size(0)-%201%20%5C%7D%20and%20%5Chspace%7B10%7Di%20%20%5Cneq%20y#0)

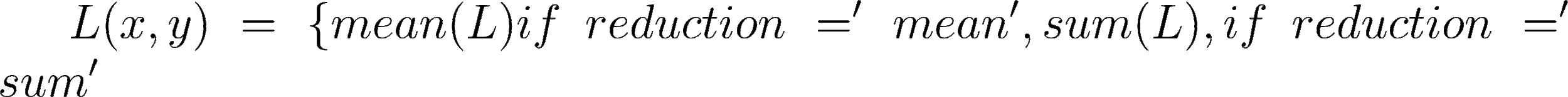
## HingeEmbeddingLoss()

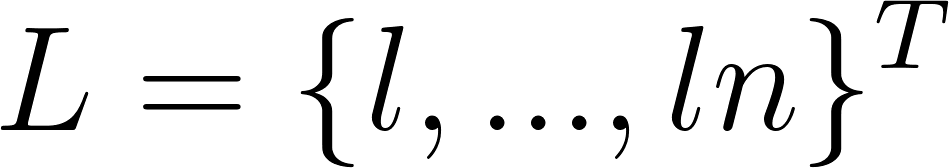
Measure the loss given an input tensor x and a label tensor y (containing 1 or -1). This is usually used for measuring whether two input are similar or dissimilar, e.g. using the L1 pairwise distance as x, and is typically used for learning nonlinear embeddings or semi-supervised learning

The loss function for n-th sample in the mini-batch is

[](https://latex-staging.easygenerator.com/eqneditor/editor.php?latex=L_n%20%3D%5C%7Bx_n%2C%20if%20%5Chspace%7B5%7Dy_n%20%3D%201%20max%5C%7B0%2C%5CDelta%20-%20x_n%5C%7D%2C%20if%5Chspace%7B6%7D%20y_n%20%3D%20-1#0)

And the total loss function is

[](https://latex-staging.easygenerator.com/eqneditor/editor.php?latex=L(x%2Cy)%20%3D%20%5C%7Bmean(L)%20if%5Chspace%7B6%7D%20reduction%20%3D%20'mean'%20%2C%20sum(L)%2C%20if%5Chspace%7B6%7D%20reduction%20%3D%20'sum'#0)

Where [](https://www.codecogs.com/eqnedit.php?latex=L%20%3D%20%5C%7Bl%2C...%2Cln%5C%7D%5ET#0)

## CosinEmbeddingLoss()

Measure the loss given input tensors x1,x2 and a Tensor label y with values 1 or -1. This is used for measuring whether two inputs are similar or dissimilar using the cosine distance, and is typically used for learning nonlinear embeddings or semi-supervised learning

The loss function for each sample is:

1 - cos(x\_1,x\_2), & \quad \text{if} y = 1 \\

max(0,cos(x\_1,x\_2) - margin) & \quad \text{if} y = -1

f(n) =

\begin{cases}

n/2 & \quad \text{if } n \text{ is even}\\

-(n+1)/2 & \quad \text{if } n \text{ is odd}

\end{cases}

For positive case make the two vector align a much as possible

For negative try to make the cos value orthogonal to each other

CTCLoss() (connectionist Temporal Classification loss)

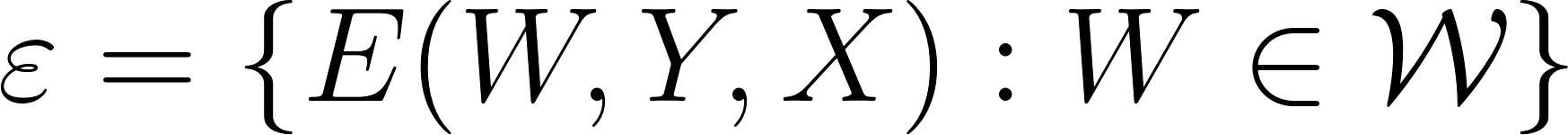
Calculate loss between a continuous (unsegmented) time series and a target sequence. CTCLoss sums over the probability of possible alignments of input to target, producing a loss value which is differentiable with respect to each input node. The alignment of input to target is assumed to be “many to one”, which limits the length of the target sequence such that it must be ≤ the input length

Use in waveform classification where the speed may play a role e.g. a word was pronoun slowly may generate multiple similar waveform/fast may generate less waveform should be map to the same word regardless of speed of speaker

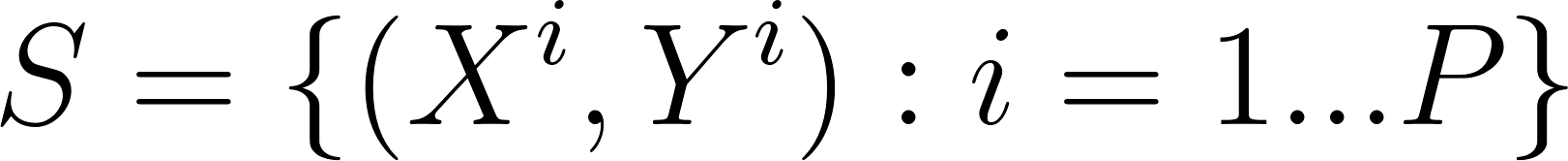
# Architecture and Loss Function

General form of energy base model

### Family of energy function

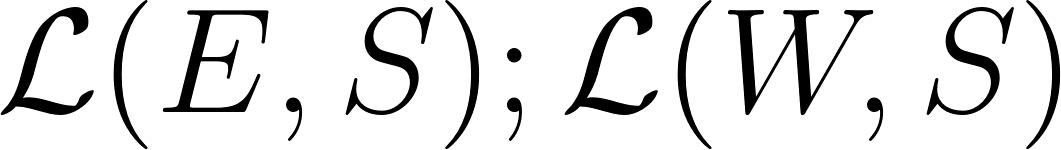
[](https://www.codecogs.com/eqnedit.php?latex=%5Cvarepsilon%20%3D%5C%7BE(W%2CY%2CX)%3A%20W%20%5Cin%20%5Cmathcal%7BW%7D%20%5C%7D#0)

### Training set

[](https://www.codecogs.com/eqnedit.php?latex=%20S%20%3D%20%5C%7B%20(X%5Ei%2CY%5Ei)%3A%20i%20%3D%201...P%5C%7D#0)

### Loss functional/Loss function

Functional is function of a function

[](https://www.codecogs.com/eqnedit.php?latex=%5Cmathcal%7BL%7D(E%2CS)%3B%20%5Cmathcal%7BL%7D(W%2CS)#0)

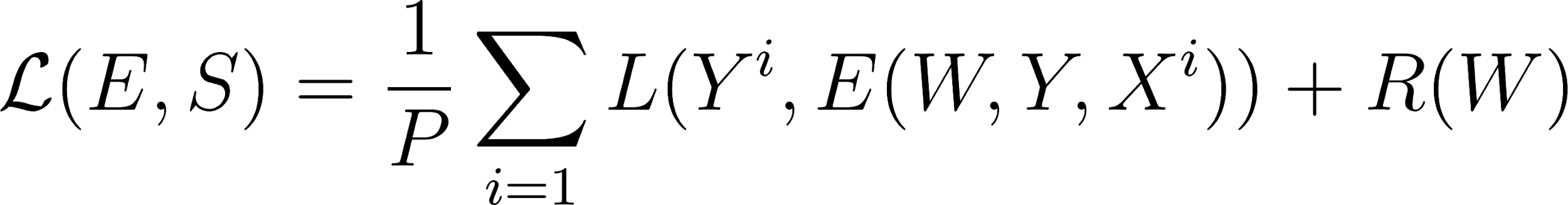
Measure the quality of an energy function on training set

### Training

[](https://www.codecogs.com/eqnedit.php?latex=W%5E*%20%3D%20min_%7B%5Csubstack%7BW%20%5Cin%20W%7D%7D%20%5Cmathcal%7BL%7D(W%2CS)#0)

### Form of the loss function

Invariant under permutations and repetition of the samples

[](https://www.codecogs.com/eqnedit.php?latex=%5Cmathcal%7BL%7D(E%2CS)%3D%20%5Cfrac%7B1%7D%7BP%7D%5Csum_%7Bi%20%3D%201%7DL(Y%5Ei%2CE(W%2CY%2CX%5Ei))%2BR(W)#0)

Where L per sample loss

Y desired answer

E energy surface for a given Xi as Y varies

R regularizer

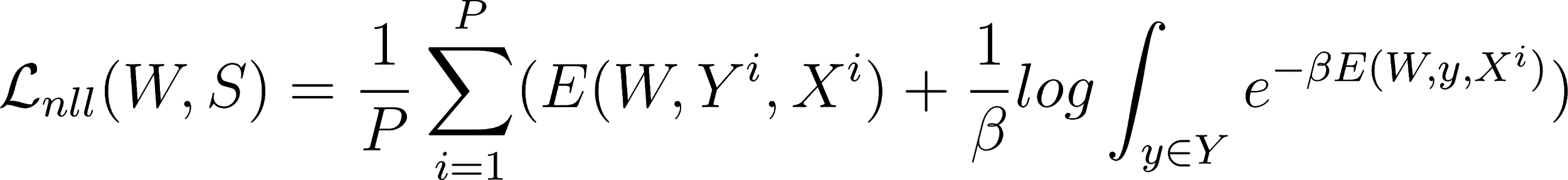
## Energy Loss function example

Energy loss

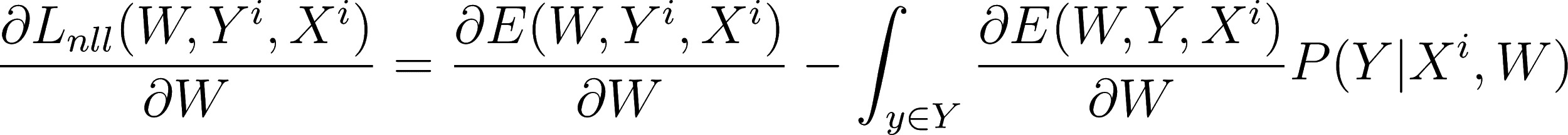
[](https://www.codecogs.com/eqnedit.php?latex=L_%7Benergy%7D(Y%5Ei%2CE(W%2CY%2CX%5Ei))%20%3D%20E(W%2CY%5Ei%2CX%5Ei)#0)

Simply pushes down on the energy o the correct answer

## Negative Log-Likelihood Loss

[](https://www.codecogs.com/eqnedit.php?latex=%5Cmathcal%7BL%7D_%7Bnll%7D(W%2CS)%20%3D%20%5Cfrac%7B1%7D%7BP%7D%20%5Csum_%7Bi%3D1%7D%5EP(E(W%2CY%5Ei%2CX%5Ei)%20%2B%20%5Cfrac%7B1%7D%7B%5Cbeta%7Dlog%20%5Cint_%7By%20%5Cin%20Y%7De%5E%7B-%5Cbeta%20E(W%2Cy%2CX%5Ei)%7D)%20#0)

Have a behavior of pushing down the correct answer push up on energy of incorrect value in proportion to their probability

[](https://www.codecogs.com/eqnedit.php?latex=%5Cfrac%7B%5Cpartial%20L_%7Bnll%7D(W%2CY%5Ei%2CX%5Ei)%7D%7B%5Cpartial%20W%7D%3D%20%5Cfrac%7B%5Cpartial%20E(W%2CY%5Ei%2CX%5Ei)%7D%7B%5Cpartial%20W%20%7D%20-%20%5Cint_%7By%20%5Cin%20Y%7D%20%5Cfrac%7B%5Cpartial%20E(W%2CY%2CX%5Ei)%7D%7B%5Cpartial%20W%7D%20P(Y%7CX%5Ei%2CW)%20%20#0)

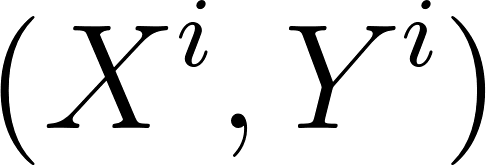
## A simpler Loss function: Perceptron Loss

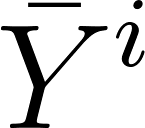
[](https://www.codecogs.com/eqnedit.php?latex=L_%7Bperceptron%7D(Y%5Ei%2CE(W%2CY%2CX%5Ei))%20%3D%20E(W%2CY%5Ei%2CX%5Ei)%20-%20min_%7By%20%5Cin%20Y%7D%20E(W%2CY%2CX%5Ei)#0)

* Push down the energy of right answer
* Push up on energy of wrong answer
* Always, positive
* Zero when answer is correct
* No margin does not prevent energy surface from being almost flat
* Work well in practice, particularly if the energy parameterization does not allow flat surfaces
* Call “discriminative Viterbi training” in speech and hand written literature
* Not work well with non linear system

## Generalized Margin Loss

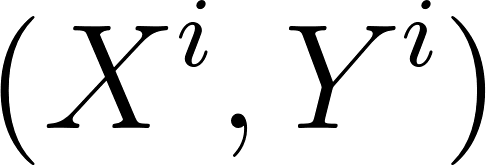
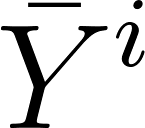
Most offending Incorrect answer: discrete case

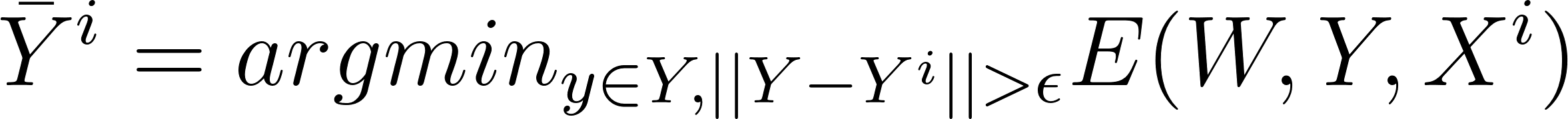
Definition 1 Let Y be a discrete variable. Then for a training sample[](https://www.codecogs.com/eqnedit.php?latex=%20(X%5Ei%2CY%5Ei)#0)

The most offending incorrect answer [](https://www.codecogs.com/eqnedit.php?latex=%20%5Cbar%7BY%7D%5Ei%20#0) is the answer that has the lowest energy among all the answer that are incorrect:

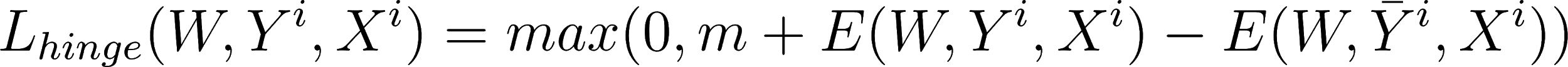
[](https://www.codecogs.com/eqnedit.php?latex=%5Cbar%7BY%7D%5Ei%20%3D%20argmin_%7By%20%5Cin%20Y%20and%20Y%5Cneq%20Y%5Ei%7D%20E(W%2CY%2CX%5Ei)#0)

Most offending Incorrect answer: discrete continuous case

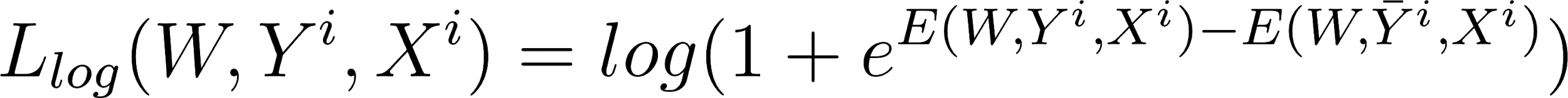
Definition 2 Let Y be a continuous variable. Then for a training sample [](https://www.codecogs.com/eqnedit.php?latex=(X%5Ei%2CY%5Ei)#0) the most offending incorrect answer [](https://www.codecogs.com/eqnedit.php?latex=%5Cbar%7BY%7D%5Ei#0) is the answer that ha the lowest energy among all answer that are at least \in away from the correct answer;

[](https://www.codecogs.com/eqnedit.php?latex=%5Cbar%7BY%7D%5Ei%20%3D%20argmin_%7By%20%5Cin%20Y%2C%7C%7CY-Y%5Ei%7C%7C%3E%5Cepsilon%7D%20E(W%2CY%2CX%5Ei)#0)

## Hinge Loss

[](https://www.codecogs.com/eqnedit.php?latex=%20L_%7Bhinge%7D(W%2CY%5Ei%2CX%5Ei)%20%3D%20max(0%2Cm%20%2B%20E(W%2CY%5Ei%2CX%5Ei)%20-%20E(W%2C%5Cbar%7BY%7D%5Ei%2CX%5Ei))#0)

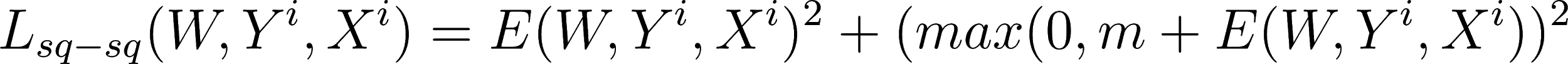
With the linearly-parameterized binary classifier architecture, we get linear SVM

[](https://www.codecogs.com/eqnedit.php?latex=%20L_%7Blog%7D(W%2CY%5Ei%2CX%5Ei)%20%3D%20log(1%20%2Be%5E%7B%20E(W%2CY%5Ei%2CX%5Ei)%20-%20E(W%2C%5Cbar%7BY%7D%5Ei%2CX%5Ei)%7D)#0)

Log loss a soft hinge loss

With the linearly-parameterized binary classifier architecture, we get linear Logistic Regression

## Square-Square Loss

[](https://www.codecogs.com/eqnedit.php?latex=%20L_%7Bsq-sq%7D(W%2CY%5Ei%2CX%5Ei)%20%3D%20E(W%2CY%5Ei%2CX%5Ei)%5E2%20%2B%20(%20max(0%2Cm%20%2B%20E(W%2CY%5Ei%2CX%5Ei)%20)%5E2%20#0)

Appropriate for positive energy function

## A more general form of Hinge type Loss

[](https://www.codecogs.com/eqnedit.php?latex=%20L(W%2CX%5Ei%2CY%5Ei)%20%3D%20%5Csum_y%20H(E(W%2CY%5Ei%2CX%5Ei)%20-%20E(W%2Cy%2CX%5Ei)%20%2B%20C(Y%5Ei%2Cy))#0)

H is Hinge energy

E another energy

C margin