# What would happen if we use MSE on binary classification?

## Context

Some comments on the "Bonus" section of this article from AI Summer.

## Screenshot

### Bonus: What would happen if we use MSE on binary classification?

So far I presented the basics. This is a bonus question that I was asked during an ML interview: What if we use MSE on binary classification?

When $\hat{y}^{(i)} = 1$:

$$\text{MSE} = \frac{1}{m}\sum_{i=1}^{m}\left\|-y^{(i)}\right\|^2 = \sum_{i=1}^{m}\left\|-\sigma(\boldsymbol{\theta}^T\mathbf{x})\right\|^2 = \sum_{i=1}^{m}\left\|\sigma(\boldsymbol{\theta}^T\mathbf{x})\right\|^2$$

When $\hat{y}^{(i)} = 0$:

$$\text{MSE} = \frac{1}{m}\sum_{i=1}^{m}\left\|1-y^{(i)}\right\|^2 = \frac{1}{m}\sum_{i=1}^{m}\left\|1-1-\sigma(\boldsymbol{\theta}^T\mathbf{x})\right\|^2 = \sum_{i=1}^{m}\left\|\sigma(\boldsymbol{\theta}^T\mathbf{x})\right\|^2$$

Both losses would push the gradients in the same direction. So my best after-interview guess is that no decision boundary can be formed. Gradients will push the model's weights to the same direction for both classes. This means that our choice of using MSE to approach binary classification is a poor one. Remember, no learning happens without assumptions and design choices.

## Review

1) "When $\hat{y}^{(i)} = 1$" and "When $\hat{y}^{(i)} = 0$" are inverted.
2) I have no idea why the authors replace $y^{(i)}$ with $\sigma(\theta^\top x)$ or $1 - \sigma(\theta^\top x)$ when the class changes. If properly trained, the model weights should "push" the sigmoid to output 0 or 1 depending on the input $x$.
3) The two equations don't prove anything:
    1) When $y^{(i)} = 0$, negative class:
    $MSE = \frac{1}{m}\sum_i \|-\hat{y}^{(i)}\|^2 = \frac{1}{m}\sum_i \|\sigma(\theta^\top x)\|^2$
    2) When $y^{(i)} = 1$, positive class:
    $MSE = \frac{1}{m}\sum_i \|1-\hat{y}^{(i)}\|^2 = \frac{1}{m}\sum_i \|1-\sigma(\theta^\top x)\|^2$

## Proposed demonstration

- Let's assume we have a simple neural network with weights $\theta$ such as $z = \theta^\top x$, and outputs $\hat{y} = \sigma(z)$ with a sigmoid activation.
- $\frac{\partial L}{\partial \theta} = \frac{\partial L}{\partial \hat{y}}\frac{\partial \hat{y}}{\partial z}\frac{\partial z}{\partial \theta}$ with $L$ being the loss.
- with MSE:
    - $L(y, \hat{y}) = \frac{1}{2}(y - \hat{y})^2$
    - $\frac{\partial L}{\partial \theta} = -(y - \hat{y})\sigma(z)(1 - \sigma(z))x$
    - $\frac{\partial L}{\partial \theta} = -(y - \hat{y})\hat{y}(1 - \hat{y})x$

  - $\sigma(z)(1-\sigma(z))$ makes the gradient really small or null if $\sigma(z)$ is close to 1. The neural net can't train.
- with BCE:
  - $L(y, \hat{y}) = -ylog(\hat{y}) - (1-y)log(1-\hat{y})$
  - for $y = 0$
    - $\frac{\partial L}{\partial \theta} = \frac{1-y}{1-\hat{y}}\sigma(z)(1-\sigma(z))x$
    - $\frac{\partial L}{\partial \theta} = \frac{1-y}{1-\hat{y}}\hat{y}(1-\hat{y})x$
    - $\frac{\partial L}{\partial \theta} = (1-y)(\hat{y})x$
    - $\frac{\partial L}{\partial \theta} = \hat{y}x$
    - If the network is right, $\hat{y} = 0$, the gradient is null.
  - for $y = 1$:
    - $\frac{\partial L}{\partial \theta} = -\frac{y}{\hat{y}}\sigma(z)(1-\sigma(z))x$
    - $\frac{\partial L}{\partial \theta} = -\frac{y}{\hat{y}}\hat{y}(1-\hat{y})x$
    - $\frac{\partial L}{\partial \theta} = -y(1-\hat{y})x$
    - $\frac{\partial L}{\partial \theta} = -(1-\hat{y})x$
    - If the network is right, $\hat{y} = 1$, the gradient is null.