Video 17

Thursday, October 22, 2020 9:51 AM

Binary classification setting:

$$z = \vec{w}^T \vec{x} + b$$

$$y = \begin{cases} 1 & \text{if } z \geq 0 \\ 0 & \text{if } z < 0 \end{cases}$$

0-1 loss to measure the error of our linear classifier:

$$\mathcal{L}_{0-1}(y, t) = \begin{cases} 0 & \text{if } y = t \\ 1 & \text{if } y \neq t \end{cases}$$

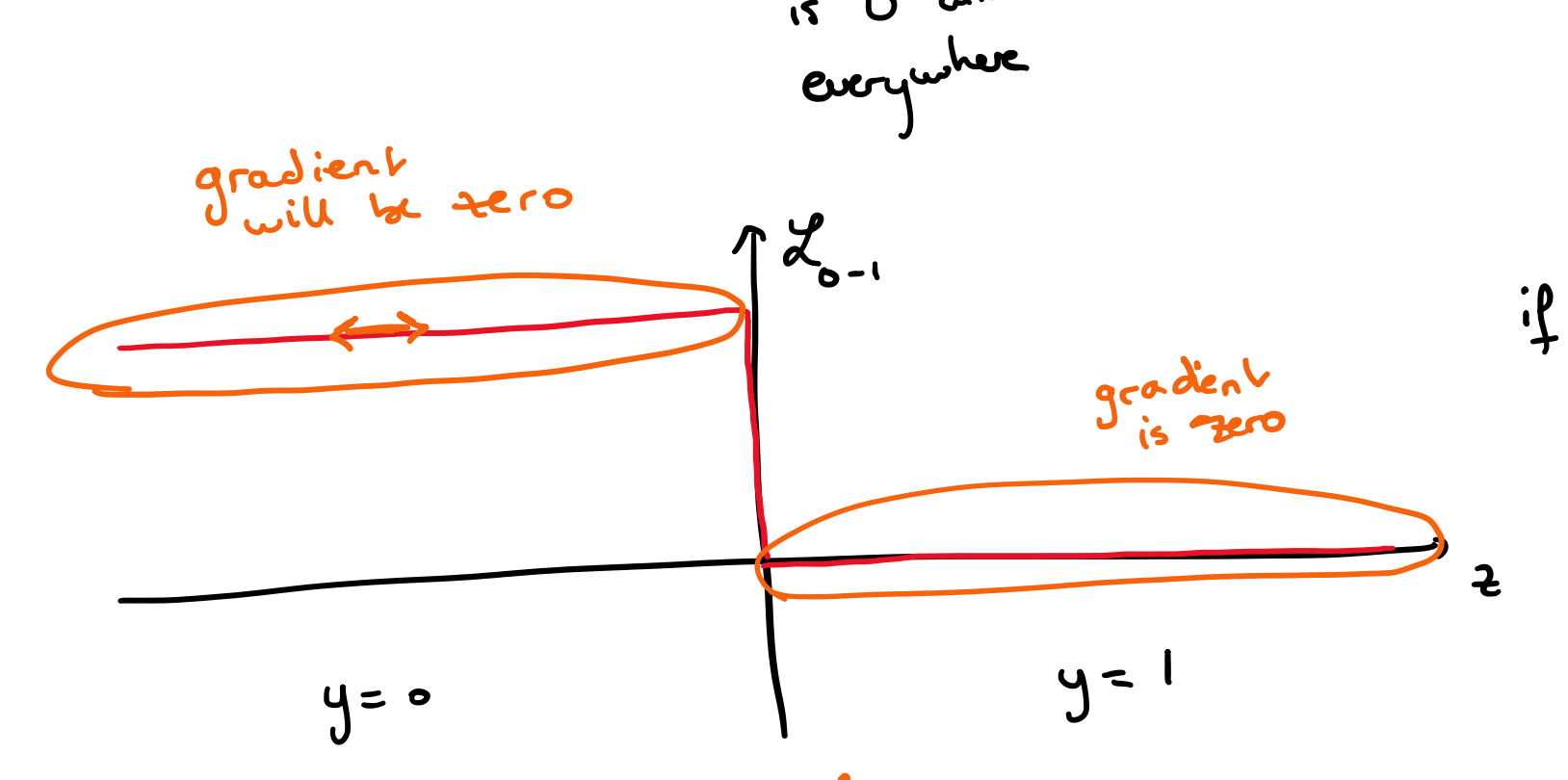
(y: prediction of the model; t: target)

$\min \mathcal{L}_{0-1} \approx$ min the number of training examples which are misclassified.

If we want to use $\mathcal{L}_{0-1}$ with gradient descent to train our linear classifier,

$$\frac{\partial \mathcal{L}_{0-1}}{\partial w_j} = \frac{\partial \mathcal{L}_{0-1}}{\partial z} \frac{\partial z}{\partial w_j}$$

($\frac{\partial \mathcal{L}_{0-1}}{\partial z}$ is 0 almost everywhere)



model is incorrect because $y \neq t$

model is correct $y = t = 1$

loss is discontinuous, gradient is not defined

Gradient descent on the $\mathcal{L}_{0-1}$ is not going to update the weights.

Least squares loss

$$\mathcal{L}_{SE}(y, t) = \frac{1}{2}(y - t)^2$$

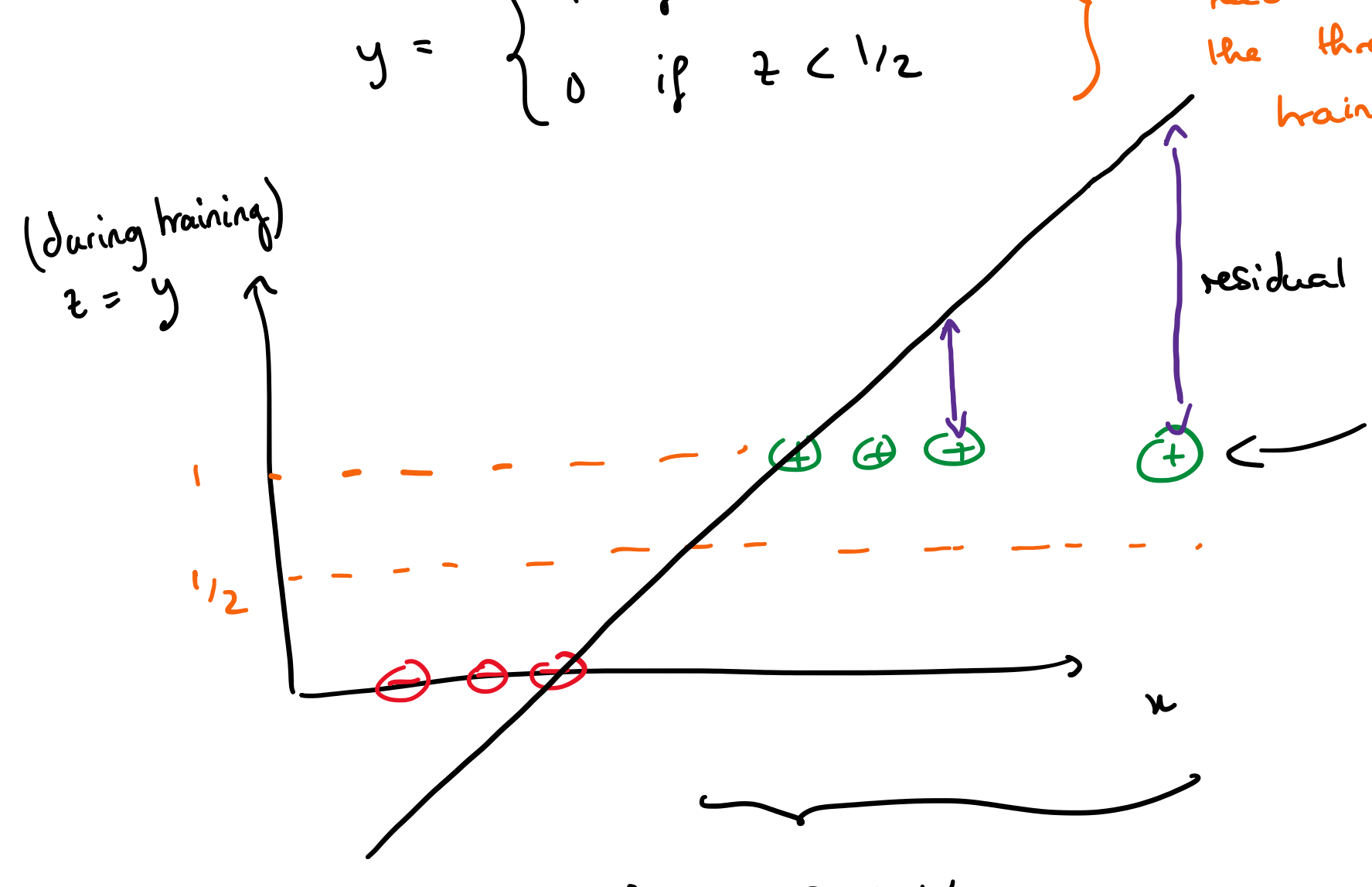
(y was originally a scalar (in the regression setting))

→ Can we use this loss in classification as well?

$$z = \vec{w}^T \vec{x} + b$$

$$y = \begin{cases} 1 & \text{if } z \geq 1/2 \\ 0 & \text{if } z < 1/2 \end{cases}$$

now we don't really need to implement the threshold during training



$y = z < 1/2$ negative.

$y = z \geq 1/2$ positive

⚠ residual for points correctly classified with a large confidence is large

⇒ loss will be large for these points whereas (if the loss reflects how wrong the model is) it should be small

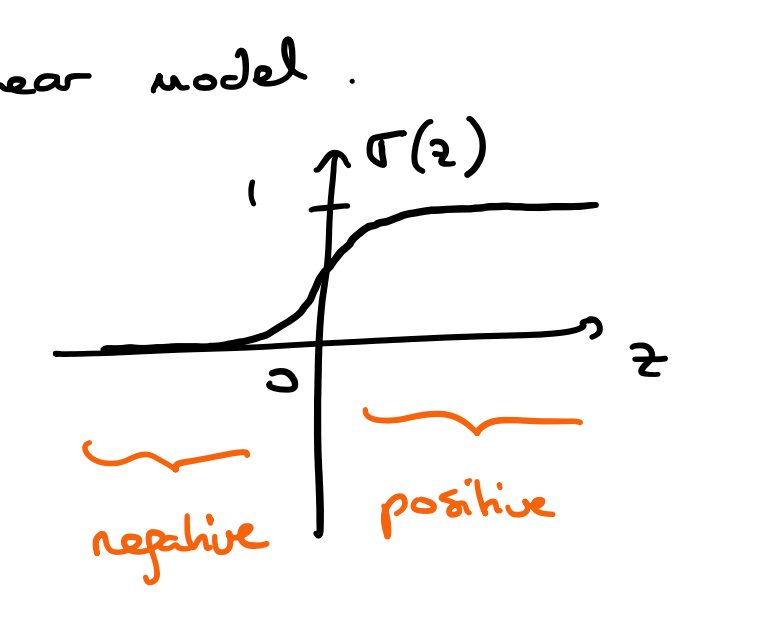
We can't use the $\mathcal{L}_{SE}$ to train our binary classifier with gradient descent.

Logistic function

Turn our linear model into a log-linear model.

Logistic function (non-linearity)

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



Log-Linear model:

$$z = \vec{w}^T \vec{x} + b$$

$$y = \sigma(z)$$

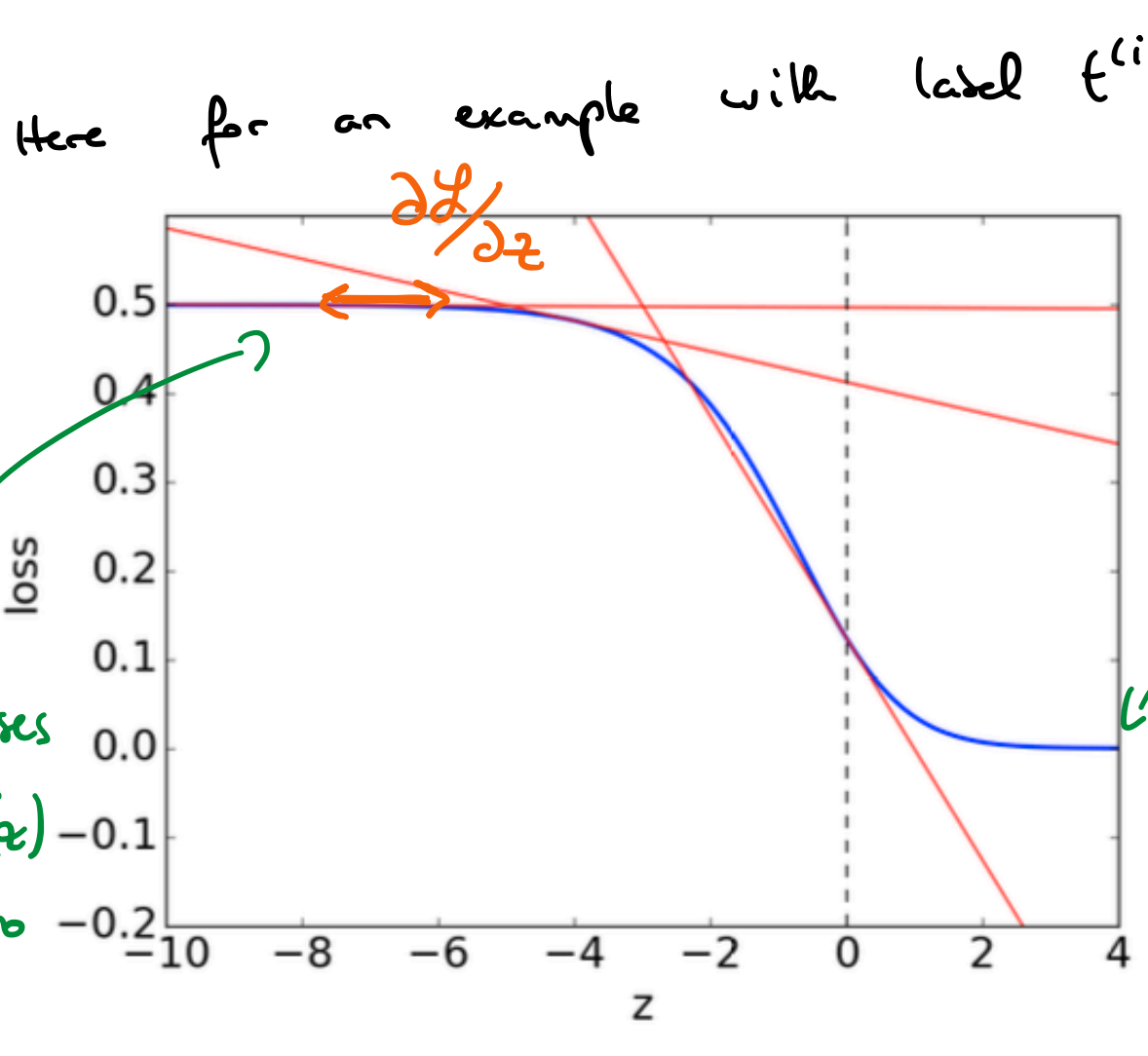
(this is called an activation function; z: logits)

y will always be between 0 and 1. As before, we can interpret:

- $y < 1/2$ as negative
- $y \geq 1/2$ as positive

Combined with our $\mathcal{L}_{SE}$.

Here for an example with label $t^{(i)} = 1$



negative (incorrect prediction)

positive (correct prediction)

as z decreases $y = \sigma(z)$ goes to 0 while $t^{(i)}$ was 1, so $\mathcal{L}_{SE}$ increases towards 1/2

$$\mathcal{L}_{SE} = \frac{1}{2}(y - t)^2 \to 1/2$$

($y \to 0$, $t = 1$)

as z increases $y = \sigma(z)$ (logistic function) increases towards 1, $\mathcal{L}_{SE}$ decrease to 0

Recall $w_j \leftarrow w_j - \alpha \frac{\partial \mathcal{L}}{\partial w_j}$

Using the chain rule, we have:

$$\frac{\partial \mathcal{L}}{\partial w_j} = \frac{\partial \mathcal{L}}{\partial z} \frac{\partial z}{\partial w_j}$$

For a positive example ($t^{(i)} = 1$), when model is confident in a negative prediction ($z \to -\infty$)

then $\frac{\partial \mathcal{L}}{\partial z} \to 0$

$\frac{\partial \mathcal{L}}{\partial w_j} \to 0$ / instable.

⚠ Gradient descent won't be able to optimize our weights properly if the model's parameters $\vec{w}$ leads to a very incorrect prediction on one of the training examples.

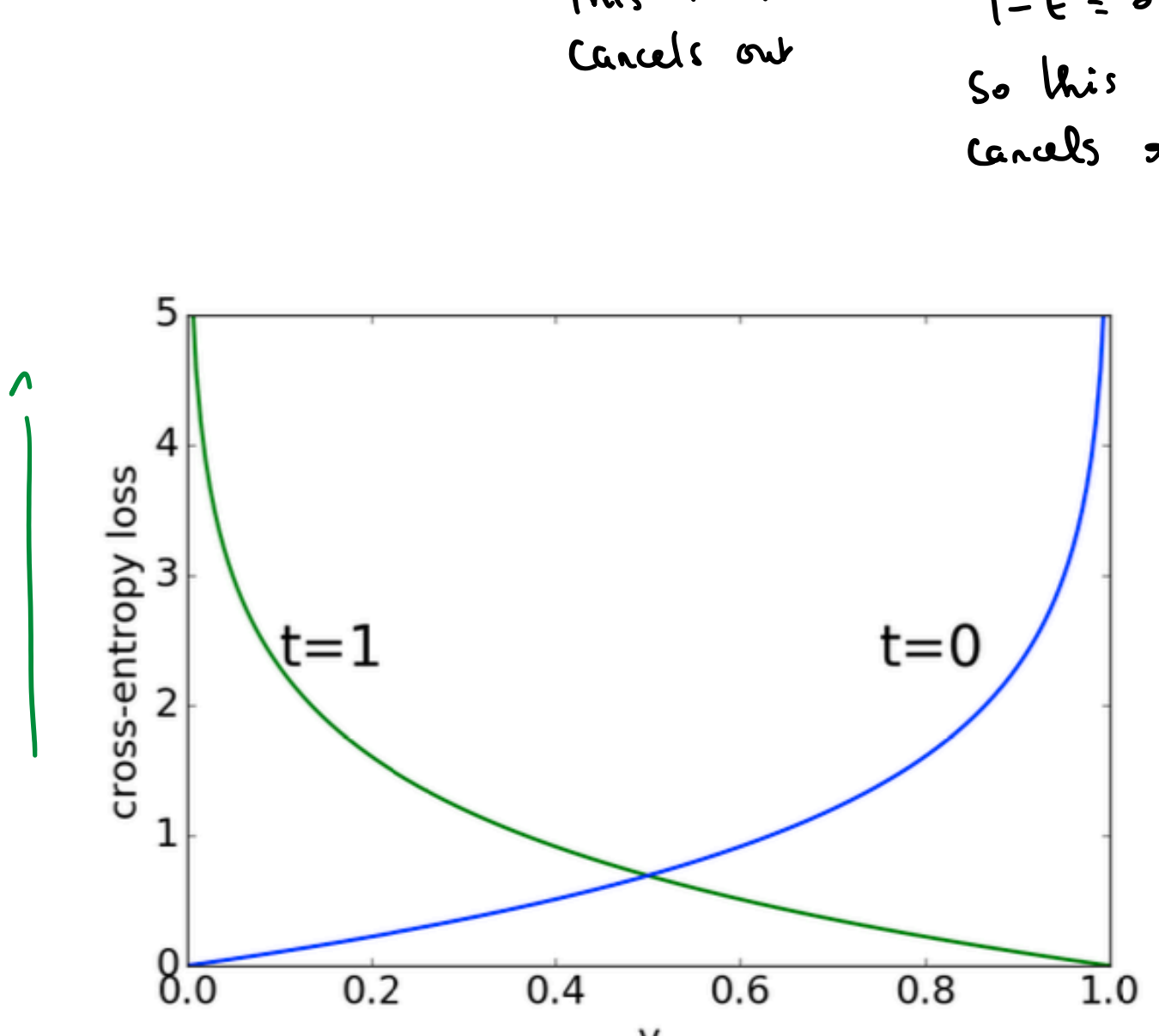
Cross-entropy loss

$$\mathcal{L}_{CE}(y, t) = \begin{cases} -\log y & \text{if } t = 1 \\ -\log(1 - y) & \text{if } t = 0 \end{cases}$$

interpreting $y \in [0, 1]$ (because we applied the logistic function) as a probability.

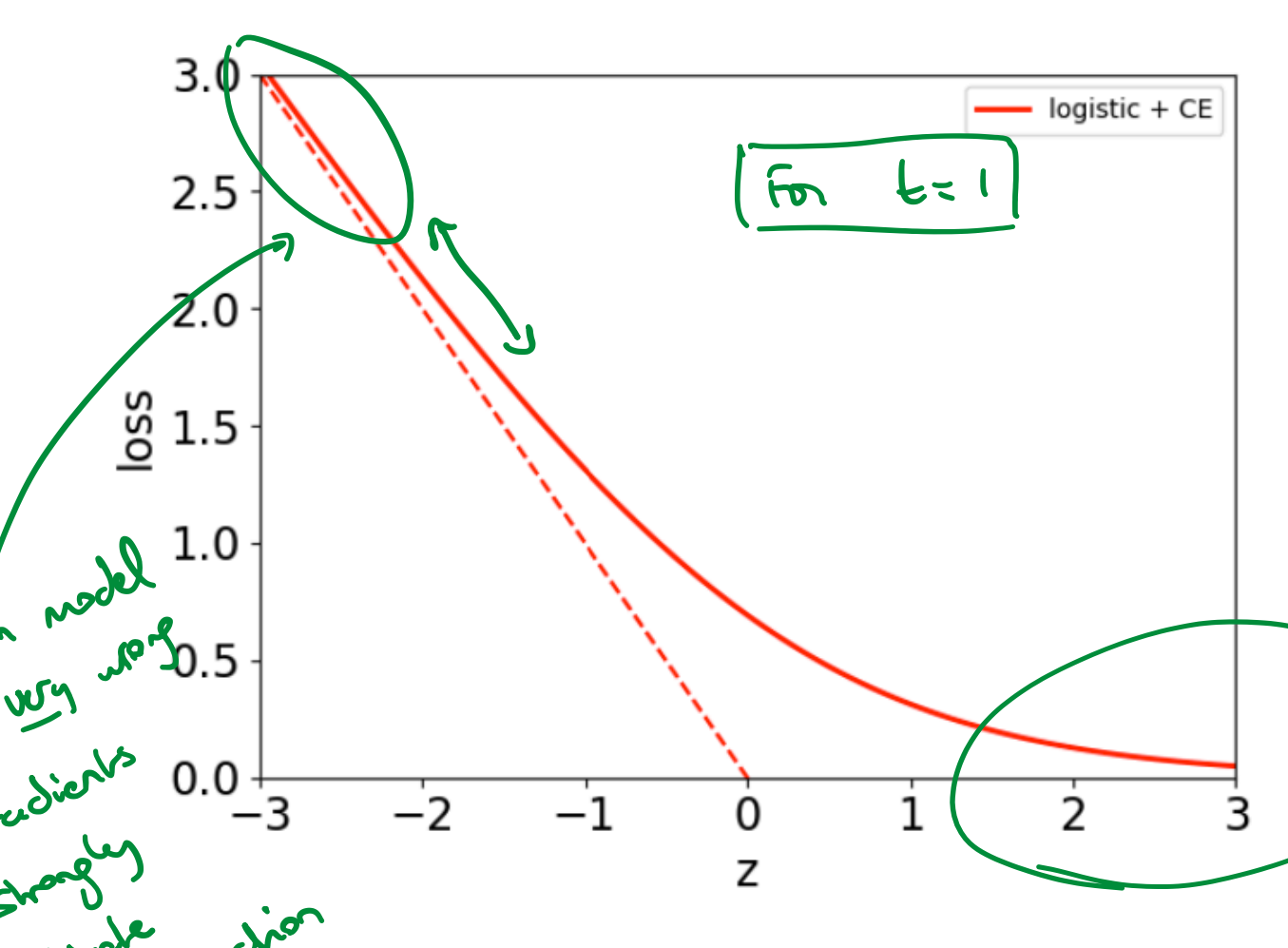
$$= -t \log y - (1 - t)\log(1 - y)$$

(if $t = 0$ this term [$-t \log y$] cancels out; if $t = 1$, $1 - t = 0$ so this term [$-(1-t)\log(1-y)$] cancels out.)



for $t = 1$, smaller values of y (up to 0) are wrong

for $t = 0$, larger values of y (up to 1) are wrong, we should instead $y \to 0$



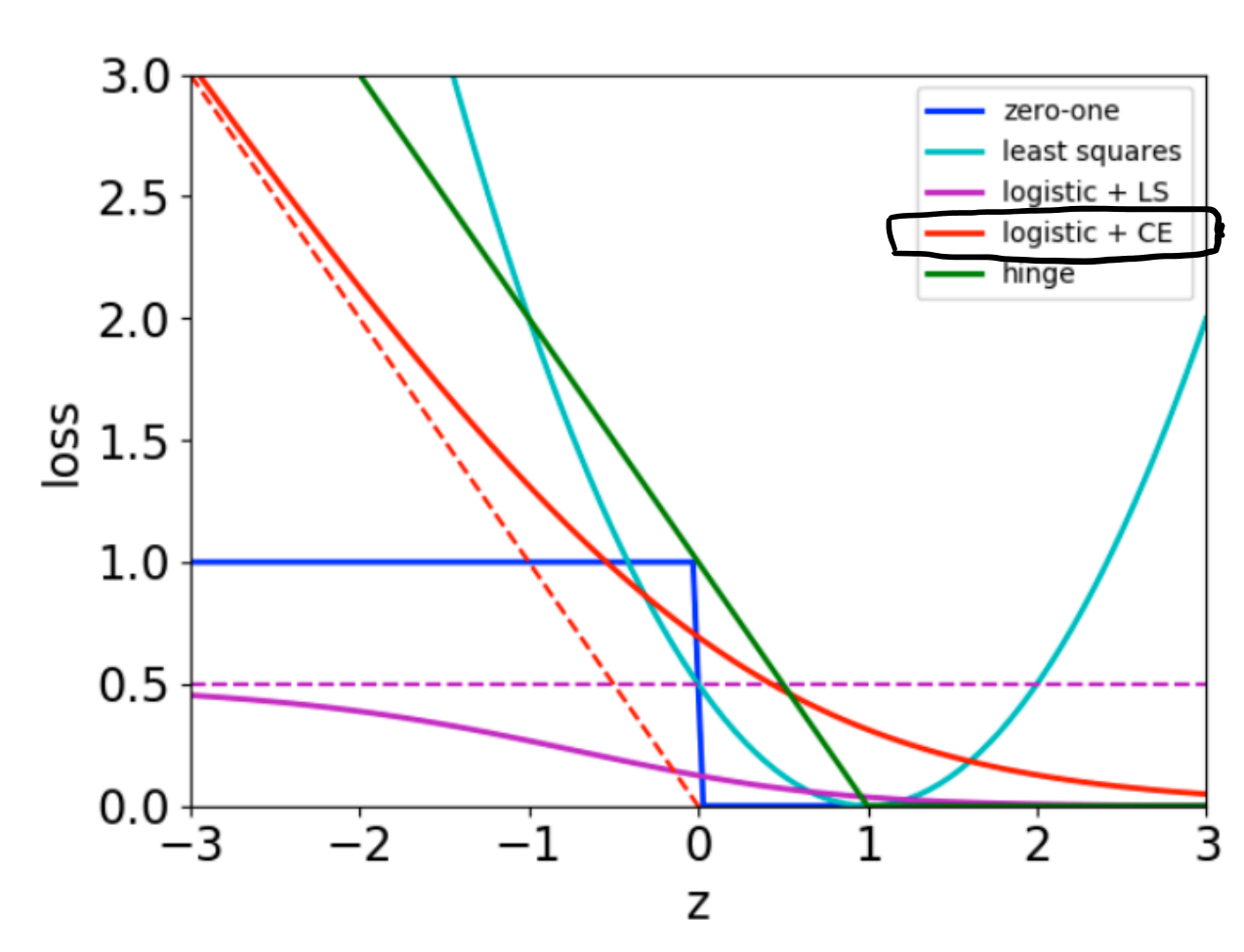
when model is very wrong gradients strongly indicate the direction of steepest descent

model is correct loss goes to 0.

$z = \vec{w}^T \vec{x} + b$

$y = \sigma(z)$

Comparison of loss functions



target $t = 1$

positive values of z are correct predictions

Hinge loss:

$$\mathcal{L}_H(y, t) = \max(0, 1 - ty)$$