Logistic Regression with Neural Networks

Fatemeh Mansoori

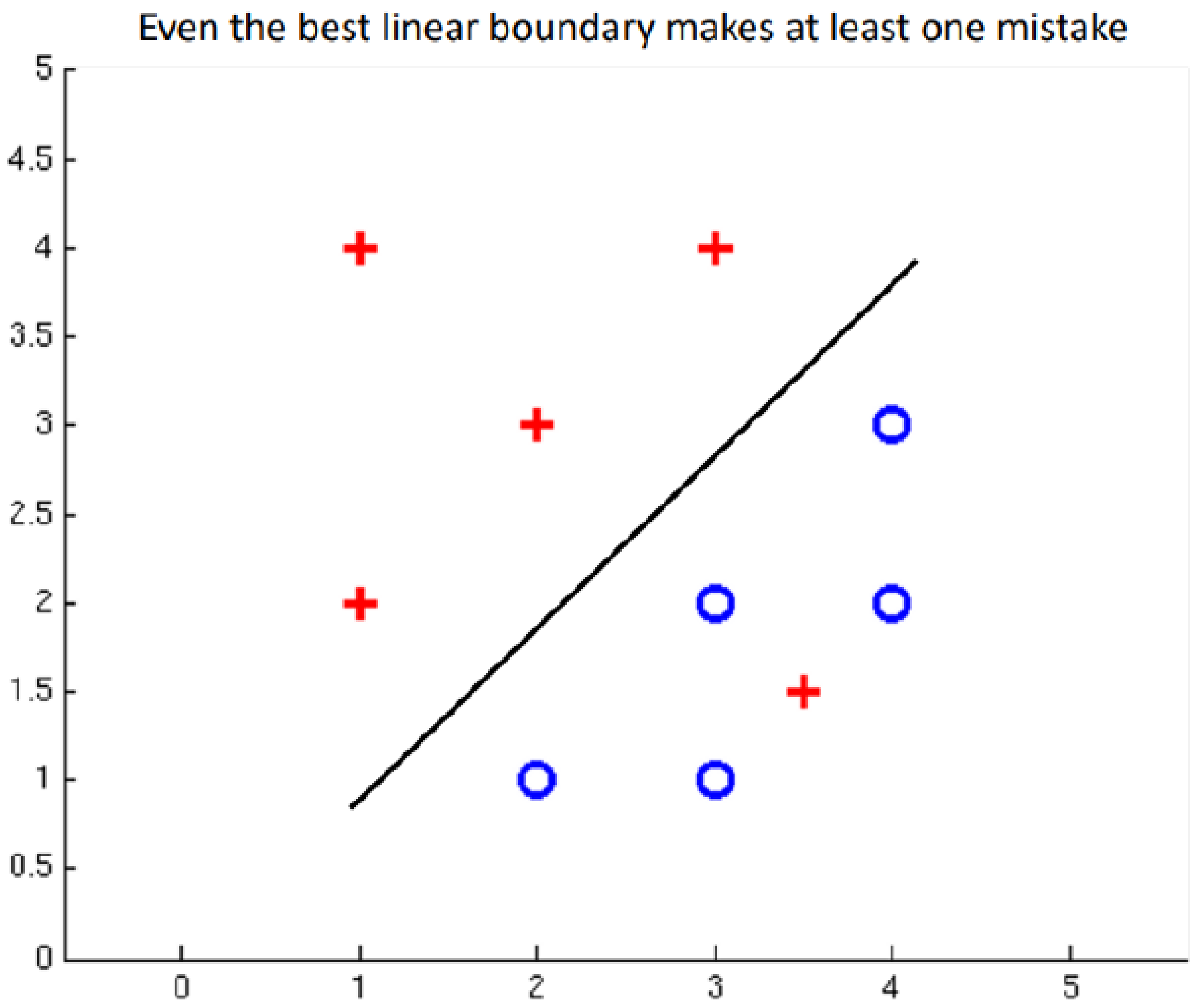
University of Isfahan

Perceptron Conlcultion

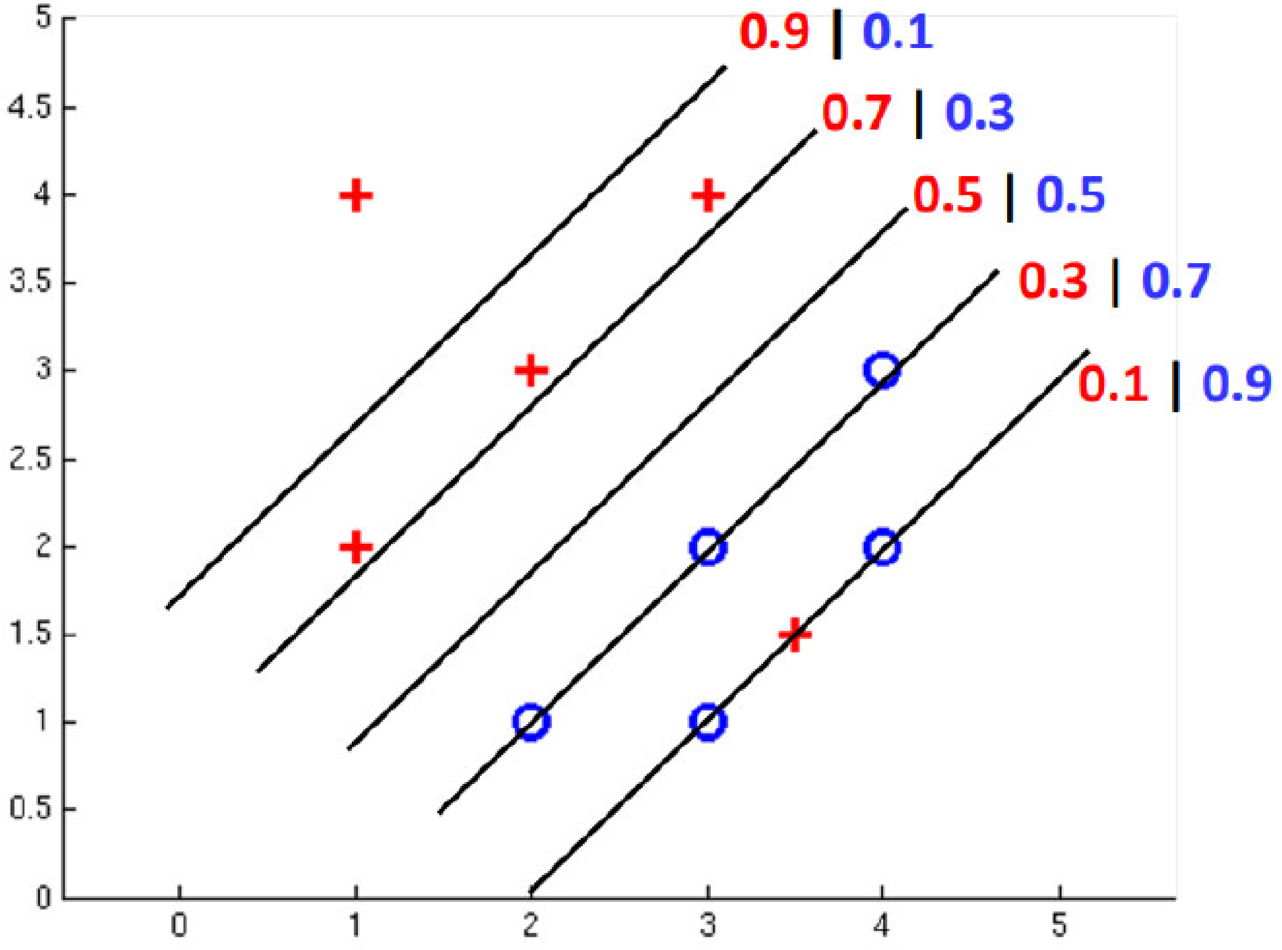
The (classic) Perceptron has many problems
(as discussed in the previous lecture)

- Linear classifier, no non-linear boundaries possible
- Binary classifier
- Does not converge if classes are not linearly separable
- Many "optimal" solutions in terms of 0/1 loss on the training data, most will not be optimal in terms of generalization performance

Non-Separable Case



Non-Separable Case: Probabilistic Decision



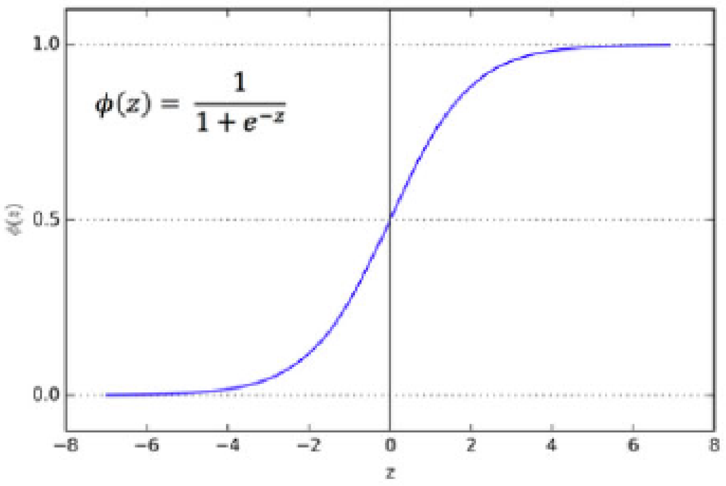
How to get probabilistic decisions?

- Perceptron scoring: $z = w \cdot f(x)$
- If $z = w \cdot f(x)$ very positive → want probability of + going to 1
- If $z = w \cdot f(x)$ very negative → want probability of + going to 0

- Sigmoid function

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

$$= \frac{e^z}{e^z + 1}$$



How to get probabilistic decisions?

- Perceptron scoring: $z = w \cdot f(x)$
- If $z = w \cdot f(x)$ very positive → want probability of + going to 1
- If $z = w \cdot f(x)$ very negative → want probability of + going to 0

- Sigmoid function

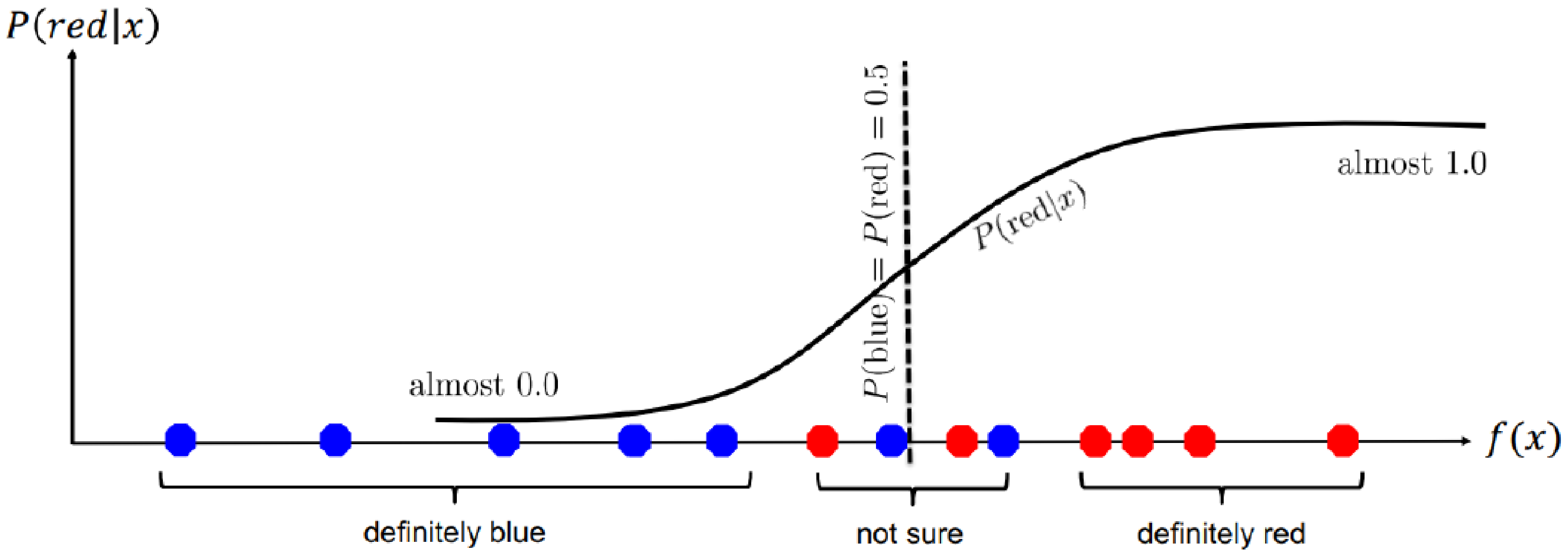
$$\phi(z) = \frac{1}{1 + e^{-z}}$$

$$P(y = +1 \mid x ; w) = \frac{1}{1+e^{-w \cdot f(x)}}$$

$$P(y = -1 \mid x ; w) = 1 - \frac{1}{1+e^{-w \cdot f(x)}}$$

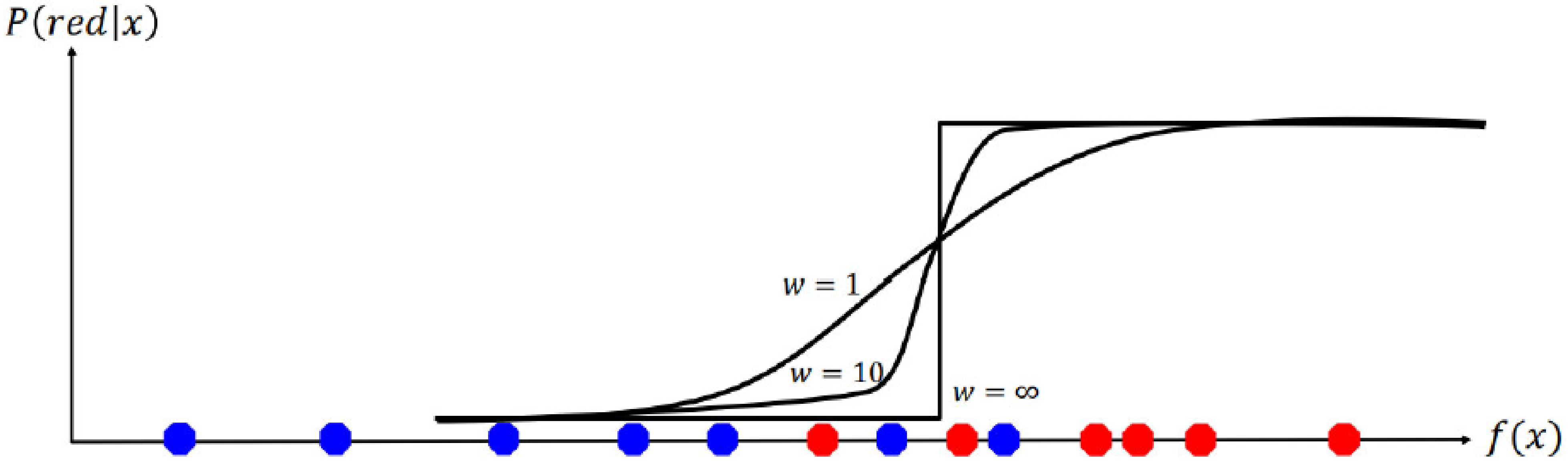
= Logistic Regression

A 1D Example



$$P(red|x\,;w) = \phi(w \cdot f(x)) = \frac{1}{1 + e^{-w \cdot f(x)}}$$

A 1D Example: varying w



$$P(red|x\,;w) = \phi(w \cdot f(x)) = \frac{1}{1 + e^{-w \cdot f(x)}}$$

Best w?

$$\begin{aligned}
\text{Likelihood} &= P(\text{training data}|w) \\
&= \prod_i P(\text{training datapoint } i \mid w) \\
&= \prod_i P(\text{point } x^{(i)} \text{ has label } y^{(i)}|w) \\
&= \prod_i P(y^{(i)}|x^{(i)}; w) \\
\text{Log Likelihood} &= \sum_i \log P(y^{(i)}|x^{(i)}; w)
\end{aligned}$$

Best w?

$$\max_w \quad ll(w) = \max_w \quad \sum_i \log P(y^{(i)}|x^{(i)}; w)$$

Logistic regression cost function

If $\quad y = 1: \qquad p(y|x) = \hat{y}$

If $\quad y = 0: \qquad p(y|x) = 1 - \hat{y}$

$$\boxed{p(y|x) = \hat{y}^{y}\,(1-\hat{y})^{(1-y)}}$$

If $y=1$: $p(y|x) = \hat{y}\;\underbrace{(1-\hat{y})^{0}}_{=1}$

If $y=0$: $p(y|x) = \underbrace{\hat{y}^{0}}_{=1}\;(1-\hat{y})^{(1-y)} = 1 \times (1-\hat{y}) = 1-\hat{y}$

$$\uparrow \log p(y|x) = \log \hat{y}^{y}(1-\hat{y})^{(1-y)} = \boxed{y\log\hat{y} + (1-y)\log(1-\hat{y})}$$

$$= -\mathcal{L}(\hat{y},y) \downarrow$$

Loss Optimization

We want to find the network weights that ***achieve the lowest loss***

$$\boldsymbol{W}^* = \underset{\boldsymbol{W}}{\operatorname{argmin}} \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}\left(f\left(x^{(i)}; \boldsymbol{W}\right), y^{(i)}\right)$$

$$\boldsymbol{W}^* = \underset{\boldsymbol{W}}{\operatorname{argmin}}\, J(\boldsymbol{W})$$

Cross Entropy Loss Optimization

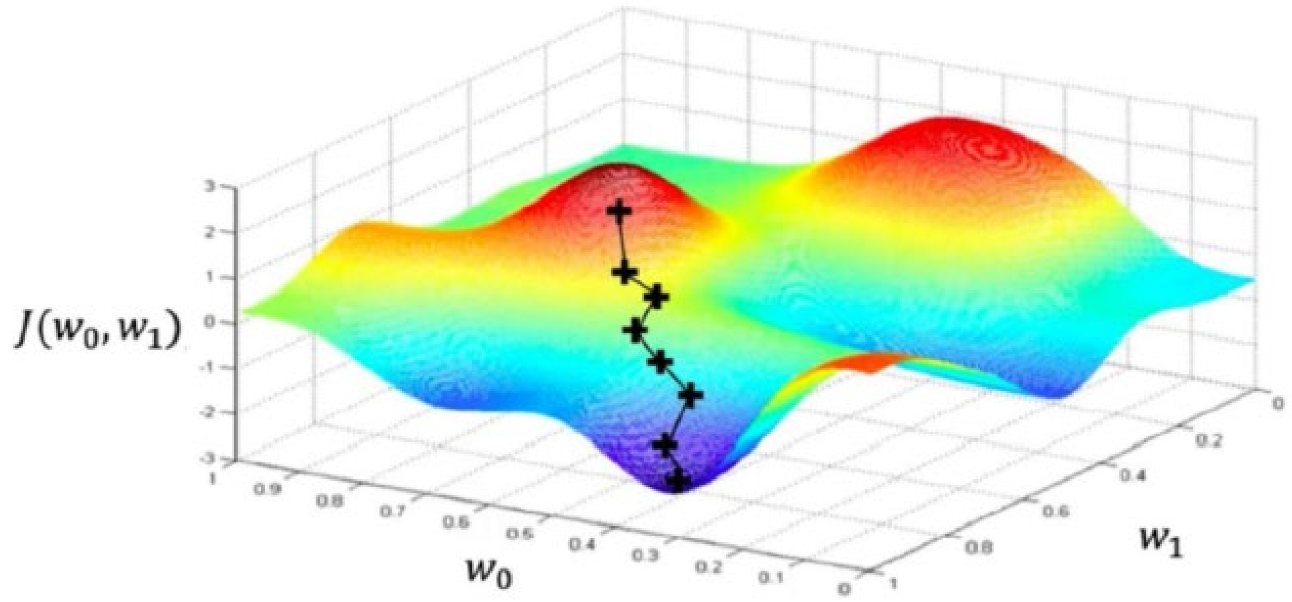
Recap: $\hat{y} = \sigma(w^T x + b)$, $\sigma(z) = \frac{1}{1+e^{-z}}$

$$J(w,b) = \frac{1}{m}\sum_{i=1}^{m} \mathcal{L}(\hat{y}^{(i)}, y^{(i)}) = -\frac{1}{m}\sum_{i=1}^{m} y^{(i)} \log \hat{y}^{(i)} + (1 - y^{(i)}) \log(1 - \hat{y}^{(i)})$$

Want to find w, b that minimize $J(w, b)$

Loss Optimization

Repeat until convergence



Gradient Descent

"Walking downhill and always taking a step in the direction that goes down the most."

Algorithm

1. Initialize weights randomly $\sim\mathcal{N}(0, \sigma^2)$
2. Loop until convergence:
3. Compute gradient, $\frac{\partial J(\boldsymbol{W})}{\partial \boldsymbol{W}}$
4. Update weights, $\boldsymbol{W} \leftarrow \boldsymbol{W} - \eta \frac{\partial J(\boldsymbol{W})}{\partial \boldsymbol{W}}$
5. Return weights

Logistic regression derivatives

$$
\left.\begin{array}{l} x_1 \\ w_1 \\ x_2 \\ w_2 \\ b \end{array}\right\} \to \boxed{z = w_1 x_1 + w_2 x_2 + b} \to \boxed{a = \sigma(z)} \to \boxed{\mathcal{L}(a, y)}
$$

$$dz = \frac{d\mathcal{L}}{dz} = \frac{d\mathcal{L}(a,y)}{dz}$$

$$= a - y$$

$$= \frac{d\mathcal{L}}{da} \cdot \boxed{\frac{da}{dz}} \quad a(1-a)$$

$$\text{"}da\text{"} = \frac{d\mathcal{L}(a,y)}{da}$$

$$= -\frac{y}{a} + \frac{1-y}{1-a}$$

$$\frac{\partial \mathcal{L}}{\partial w_1} = \text{"}dw_1\text{"} = x_1 \cdot dz. \qquad dw_2 = x_2 \cdot dz. \qquad db = dz.$$

$$w_1 := w_1 - \alpha\, dw_1$$

$$w_2 := w_2 - \alpha\, dw_2$$

$$b := b - \alpha\, db.$$