Regularized linear regression

$$\min_{\vec{w},b} J(\vec{\mathrm{w}}, b) = \min_{\vec{w},b}\left[\frac{1}{2m}\sum_{i=1}^{m}\left(f_{\vec{\mathrm{w}},b}\left(\vec{\mathrm{x}}^{(i)}\right) - y^{(i)}\right)^2 + \frac{\lambda}{2m}\sum_{j=1}^{n} w_j^2\right]$$

Gradient descent

repeat {

$$w_j = w_j - \alpha \frac{\partial}{\partial w_j} J(\vec{\mathrm{w}}, b)$$

$$b = b - \alpha \frac{\partial}{\partial b} J(\vec{\mathrm{w}}, b)$$

} simultaneous update

Regularized linear regression

$$\min_{\vec{w},b} J(\vec{w}, b) = \min_{\vec{w},b} \left[\frac{1}{2m} \sum_{i=1}^{m} \left(f_{\vec{w},b}(\vec{x}^{(i)}) - y^{(i)} \right)^2 + \frac{\lambda}{2m} \sum_{j=1}^{n} w_j^2 \right]$$

Gradient descent

repeat {

$$w_j = w_j - \alpha \frac{\partial}{\partial w_j} J(\vec{w}, b)$$

$j = 1, \ldots, n$

$$= \boxed{\frac{1}{m} \sum_{i=1}^{m} \left(f_{\vec{w},b}(\vec{x}^{(i)}) - y^{(i)} \right) x_j^{(i)} + \frac{\lambda}{m} w_j}$$

$$b = b - \alpha \frac{\partial}{\partial b} J(\vec{w}, b)$$

$$= \frac{1}{m} \sum_{i=1}^{m} \left(f_{\vec{w},b}(\vec{x}^{(i)}) - y^{(i)} \right)$$

} simultaneous update

don't have to regularize b

How we get the derivative term (optional)

$$\frac{\partial}{\partial w_j} J(\vec{\mathrm{w}},b) = \frac{\partial}{\partial w_j}\left[\frac{1}{2m}\sum_{i=1}^{m}\Big(\underbrace{f(\vec{x}^{(i)})}_{\vec{w}\cdot\vec{x}^{(i)}+b} - y^{(i)}\Big)^2 + \frac{\lambda}{2m}\sum_{j=1}^{n} w_j^2\right]$$

$$= \frac{1}{\cancel{2}m}\sum_{i=1}^{m}\left[\big(\vec{w}\cdot\vec{x}^{(i)} + b - y^{(i)}\big)\,\cancel{2}\,x_j^{(i)}\right] + \frac{\lambda}{\cancel{2}m}\cancel{2}\,w_j$$

No $\sum_{j=1}^{n}$

$$= \frac{1}{m}\sum_{i=1}^{m}\left[\big(\underbrace{\vec{w}\cdot\vec{x}^{(i)} + b}_{f(\vec{x})} - y^{(i)}\big)\,x_j^{(i)}\right] + \frac{\lambda}{m}w_j$$

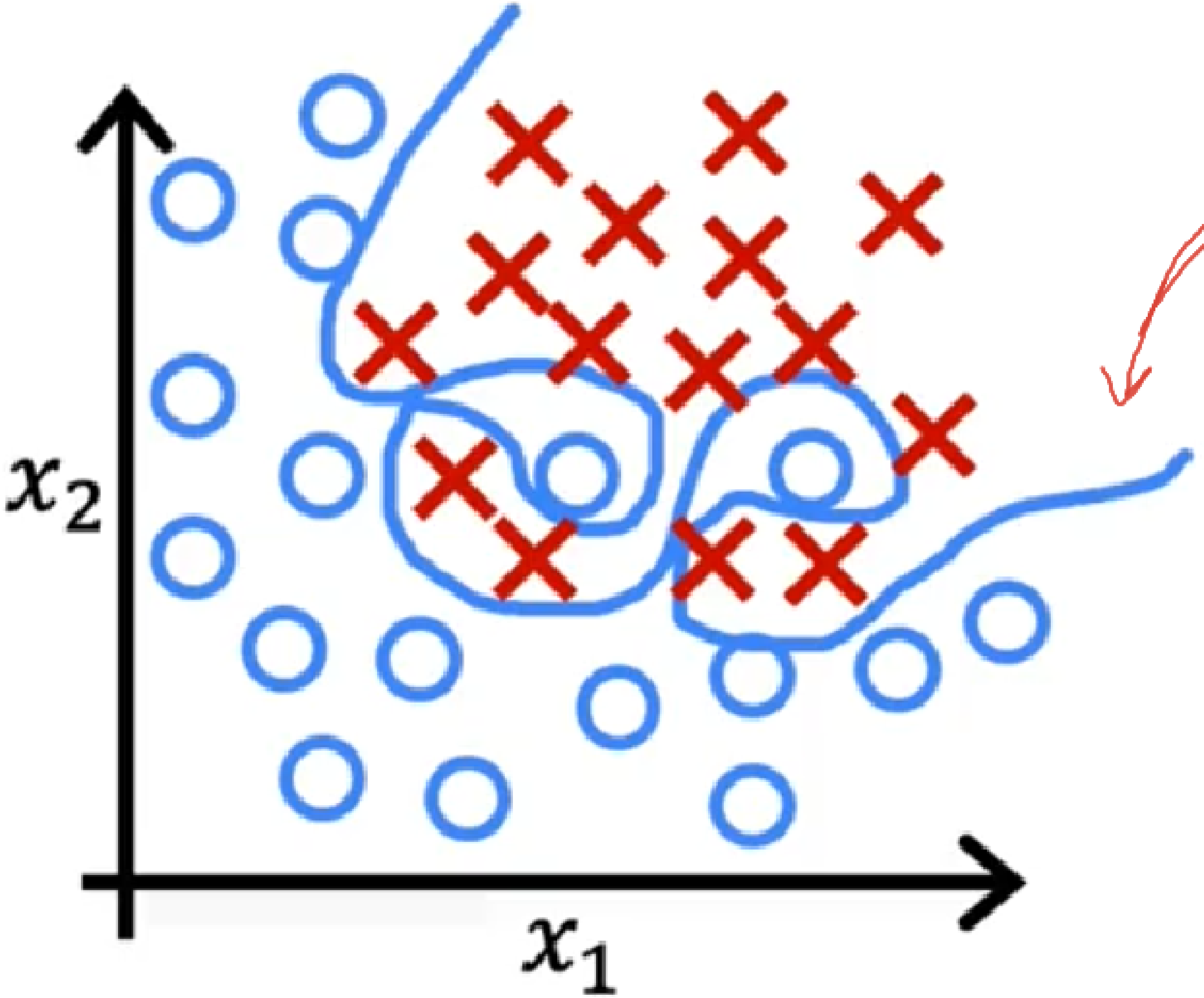
$$= \frac{1}{m}\sum_{i=1}^{m}\left[\big(f_{\vec{\mathrm{w}},b}(\vec{\mathrm{x}}^{(i)}) - y^{(i)}\big)x_j^{(i)}\right] + \frac{\lambda}{m}w_j$$

$u^2 \to 2uu'$

$\frac{\partial J}{\partial w_2} \to$

$J = \cancel{2w_1} + \boxed{2w_2} \cancel{\ldots}$

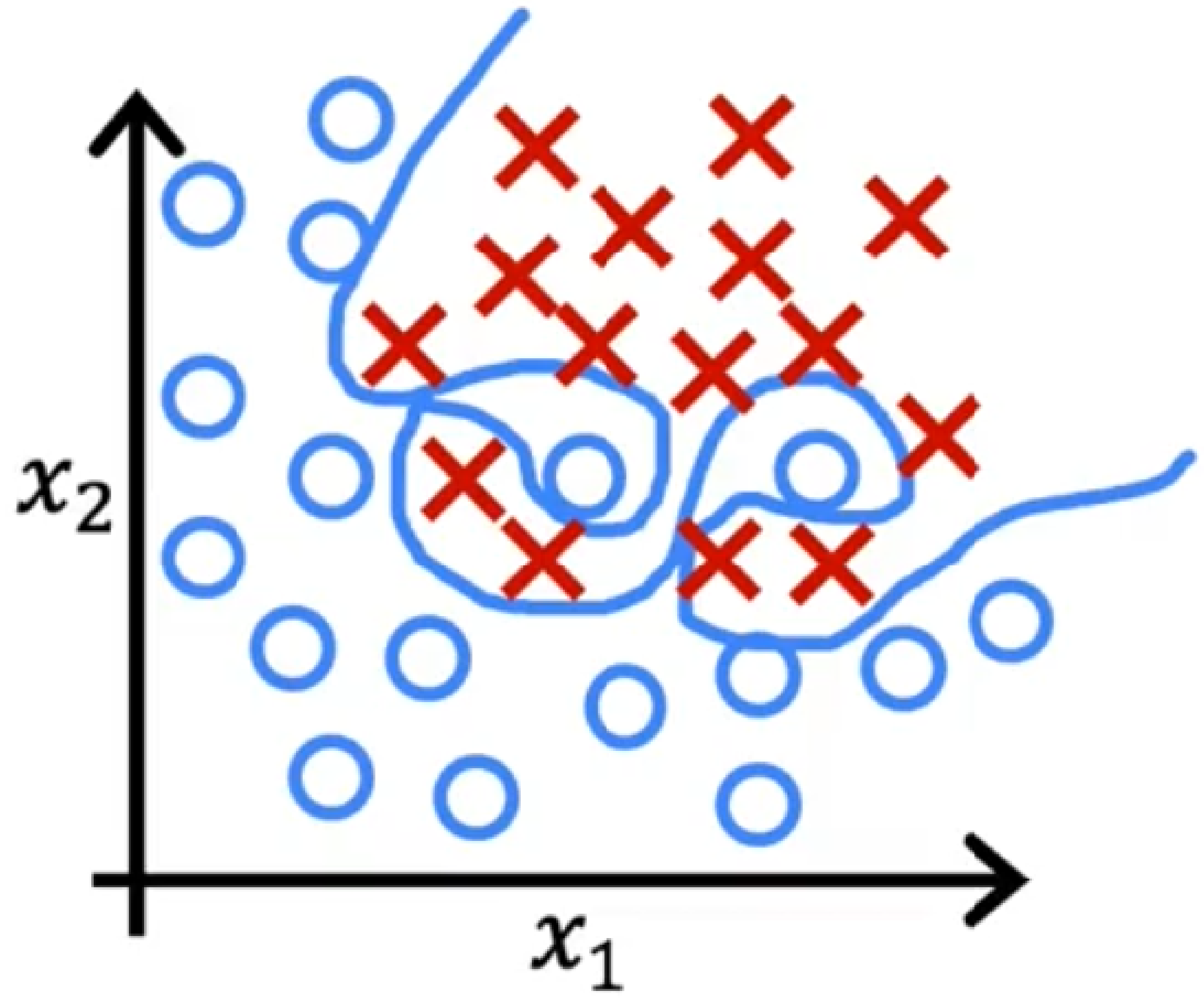
Regularized logistic regression



$$z = w_1x_1 + w_2x_2 + w_3x_1^2x_2 + w_4x_1^2x_2^2 + w_5x_1^2x_2^3 + \cdots + b$$

$$\boxed{f_{\vec{w},b}(\vec{x}) = \frac{1}{1+e^{-z}}}$$

Regularized logistic regression



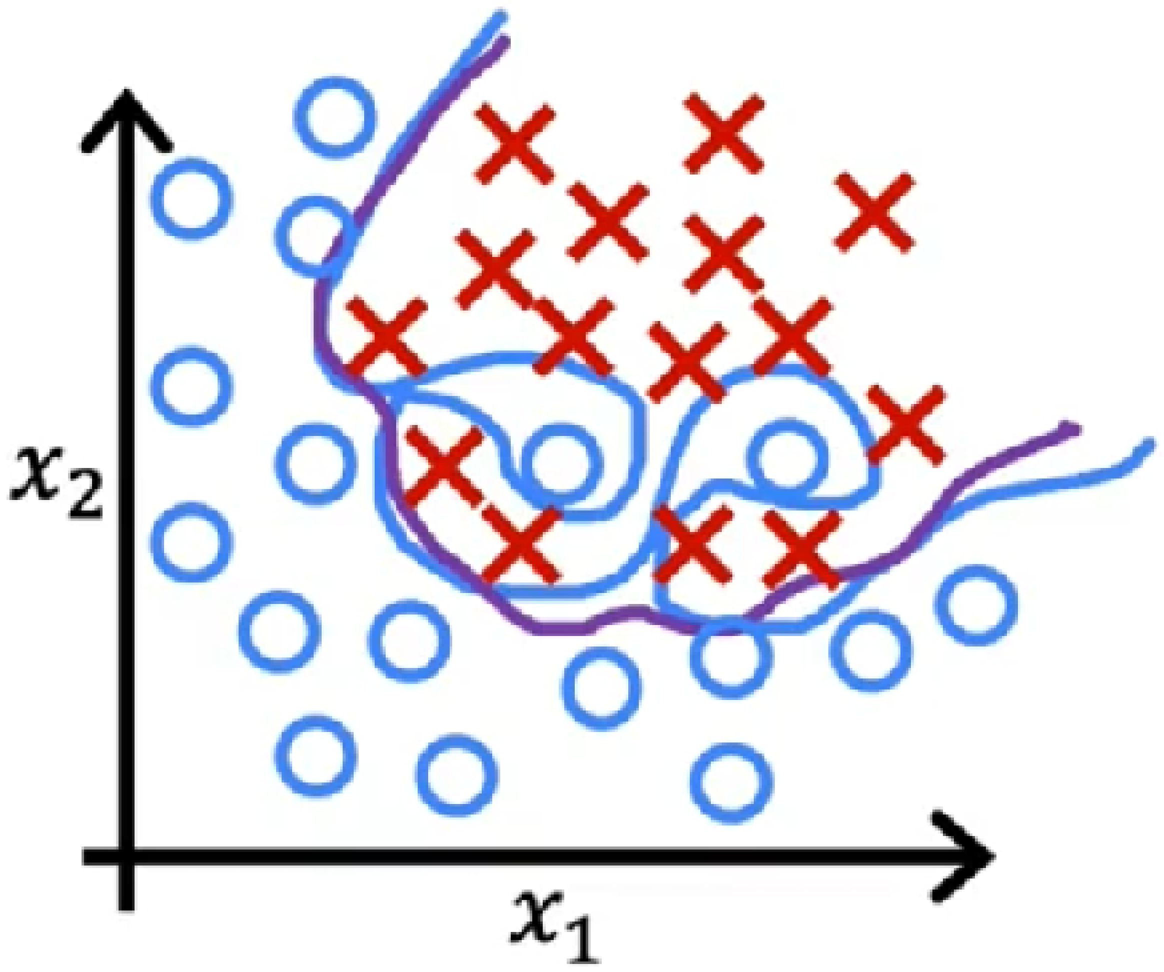
$$z = w_1x_1 + w_2x_2 + w_3x_1^2x_2 + w_4x_1^2x_2^2 + w_5x_1^2x_2^3 + \cdots + b$$

$$f_{\vec{\mathrm{w}},b}(\vec{\mathrm{x}}) = \frac{1}{1+e^{-z}}$$

Cost function

$$J(\vec{\mathrm{w}},b) = -\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)}\log\left(f_{\vec{\mathrm{w}},b}(\vec{\mathrm{x}}^{(i)})\right) + (1-y^{(i)})\log\left(1-f_{\vec{\mathrm{w}},b}(\vec{\mathrm{x}}^{(i)})\right)\right]$$

Regularized logistic regression



$$z = w_1x_1 + w_2x_2 + w_3x_1^2x_2 + w_4x_1^2x_2^2 + w_5x_1^2x_2^3 + \cdots + b$$

$$f_{\vec{\mathrm{w}},b}(\vec{\mathrm{x}}) = \frac{1}{1+e^{-z}}$$

Cost function

$$J(\vec{\mathrm{w}},b) = -\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)}\log\left(f_{\vec{\mathrm{w}},b}(\vec{\mathrm{x}}^{(i)})\right) + (1-y^{(i)})\log\left(1-f_{\vec{\mathrm{w}},b}(\vec{\mathrm{x}}^{(i)})\right)\right] + \boxed{\frac{\lambda}{2m}\sum_{j=1}^{n} w_j^2}$$

$$\min_{\vec{w},b} J(\vec{\mathrm{w}},b) \rightarrow w_j \downarrow$$

Regularized logistic regression

$$J(\vec{\mathrm{w}}, b) = -\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)}\log\left(f_{\vec{\mathrm{w}},b}(\vec{\mathrm{x}}^{(i)})\right) + (1-y^{(i)})\log\left(1-f_{\vec{\mathrm{w}},b}(\vec{\mathrm{x}}^{(i)})\right)\right] + \frac{\lambda}{2m}\sum_{j=1}^{n} w_j^2$$

$\min_{\vec{w},b}$

Gradient descent

repeat {

$$w_j = w_j - \alpha\frac{\partial}{\partial w_j}J(\vec{\mathrm{w}}, b) \qquad j=1\ldots n$$

$$= \boxed{\frac{1}{m}\sum_{i=1}^{m}\left[\left(f_{\vec{\mathrm{w}},b}(\vec{\mathrm{x}}^{(i)}) - y^{(i)}\right)x_j^{(i)}\right]} + \frac{\lambda}{m}w_j$$

Looks same as for linear regression!

$$b = b - \alpha\frac{\partial}{\partial b}J(\vec{\mathrm{w}}, b)$$

$$= \frac{1}{m}\sum_{i=1}^{m}\left(f_{\vec{\mathrm{w}},b}(\vec{\mathrm{x}}^{(i)}) - y^{(i)}\right)$$

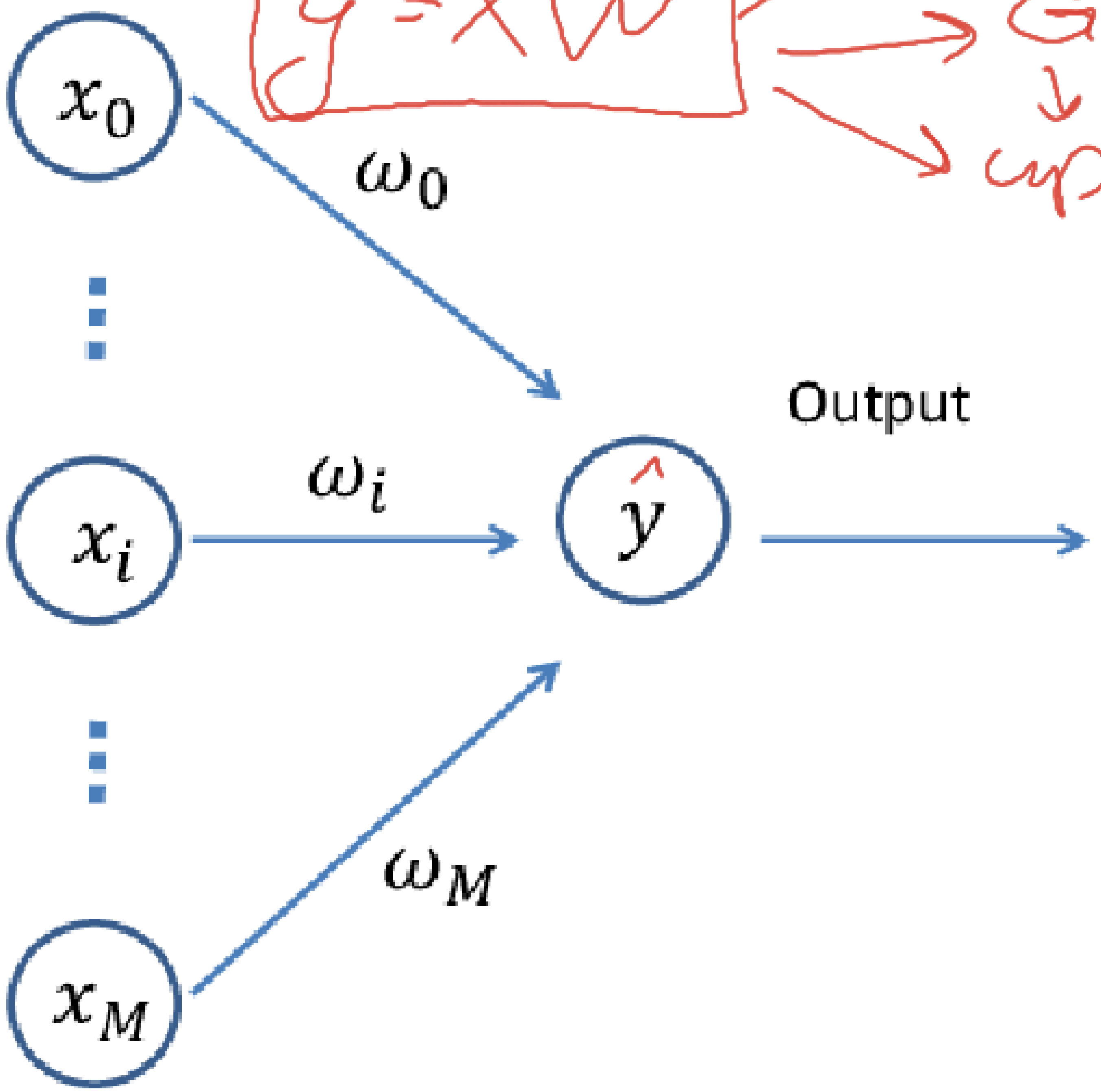
}

$$\underbrace{\boxed{\hat{y} = XW}}_{NN}$$

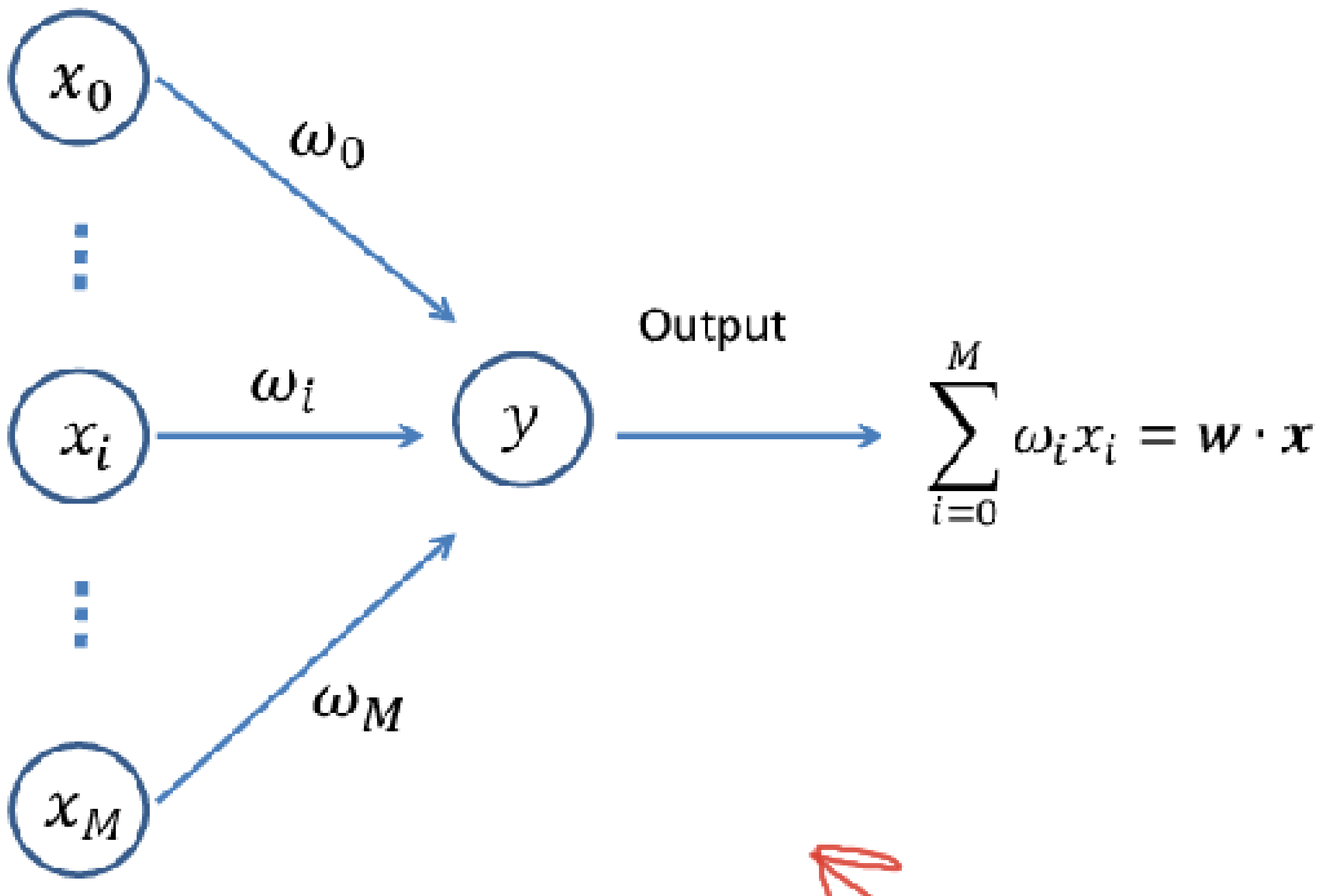
loss

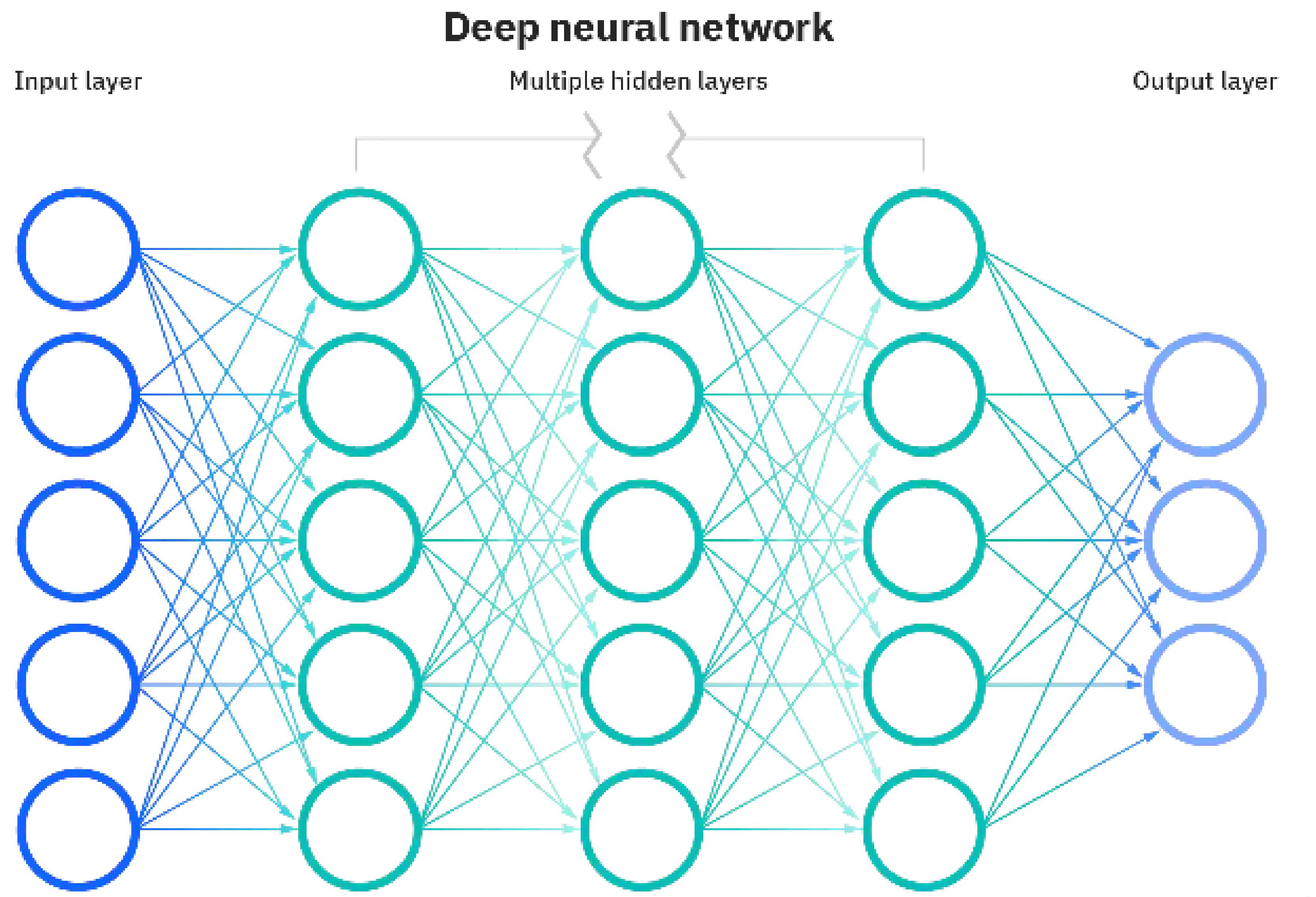
GD

update



$$\sum_{i=0}^{M} \omega_i x_i = \boldsymbol{w} \cdot \boldsymbol{x}$$


x_0
ω_0
x_i
ω_i
y
Output
x_M
ω_M
$\sum_{i=0}^{M} \omega_i x_i = \boldsymbol{w} \cdot \boldsymbol{x}$


Deep neural network
Input layer
Multiple hidden layers
Output layer

Newton's Method

$f(x) = 0$

$\downarrow$

?

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

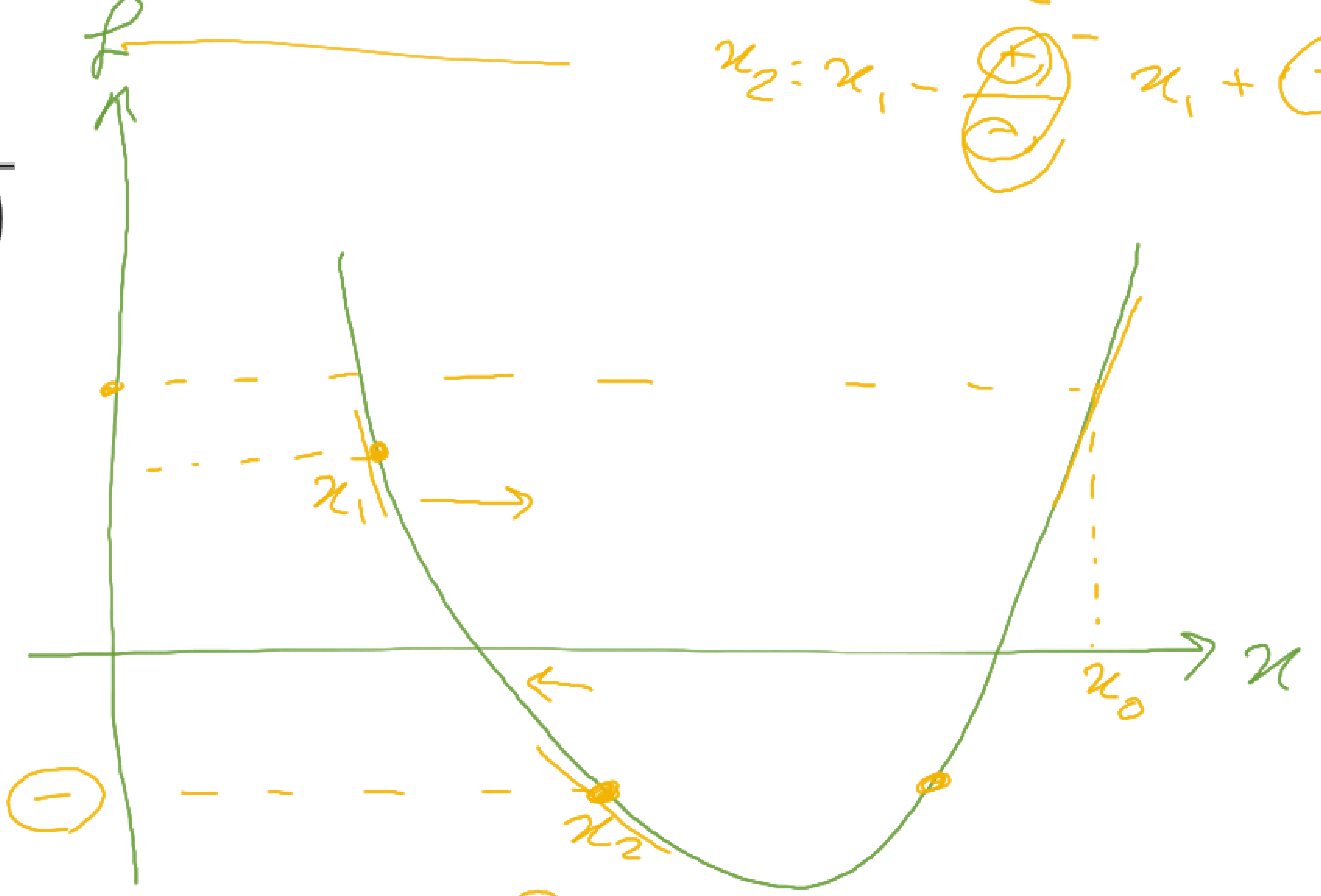
$$x^{new} = x^{old} - \frac{f(x^{old})}{f'(x^{old})}$$

```
def newton(f,Df,x0,epsilon,max_iter):
    xn = x0
    for n in range(0,max_iter):
        fxn = f(xn)
        if abs(fxn) < epsilon:
            print('Found solution after',n,'iterations.')
            return xn
        Dfxn = Df(xn)
        if Dfxn == 0:
            print('Zero derivative. No solution found.')
            return None
        xn = xn - fxn/Dfxn
    print('Exceeded maximum iterations. No solution found.')
    return None
```

```
p = lambda x: x**3 - x**2 - 1
Dp = lambda x: 3*x**2 - 2*x
approx = newton(p,Dp,1,1e-10,100)
print(approx)
```

$$x_1 = x_0 - \frac{+}{+}$$

$$x_2 = x_1 - \frac{+}{-} \qquad x_1 + (+)$$



$$x_3 = x_2 - \frac{-}{-} \qquad x_2 - (+)$$