→ Polynomial Regression

$y = f(x) = b + w_1 x + w_2 x^2 + w_3 x^3 + \ldots + w_n x^n$

$\vec{x} = [x, x^2, x^3, \ldots, x^n]^T$

$\vec{w} = [w_1, w_2, \ldots, w_n]^T$

$y = \langle \vec{w} \cdot \vec{x} \rangle + \text{bias}$

$\Delta \vec{w} = \eta (t_i - y_i) \vec{x_i}$

$\Delta b = \eta (t_i - y_i)$

= my CODE =

```
# first i define my bessel_0 function

import numpy as np
import scipy.special as sp
import matplotlib.pyplot as plt
N = 20, 100, 1000, 10000
x = np.linspace ( 0, 8 , 1000 )
rng = np.random. default_rng()
rng.shuffle (x)
x = np. sort ( x [:N] )
```

① def J_0(x):
    n = 0
    J_ = sp1. jv (n, x)
    return J_0

② # create my noisy bessel function wit gaussion
J_0(x) + 0,1 x N (0,1)

```
def f(x):
    gauss = np.random.random (0,1, 1000)
    f = J_0(x) + (0,1) * gauss
    return f
```

③ Train an 8th order polynomial regression