عملیات برداری

X = [1, 2, 3, 4, 6, ...]

w = ~

y = [2, 4, 6, 8, ... —]

b = ~

y_hat = w * x[i] + b

J += (y_hat - y[i]) ** 2

عدد

بردار

y_hat = w * x + b

J += (y_hat - y).sum

J += (y_hat - y).sum() / (2*m)

y_hat = [w*1+b, w*2+b, w*3+b, ...]

عملیات ماتریسی

$$y = ax + b$$

$$\hat{y} = w_1 x_1 + w_2 x_2 + \dots + b$$

$$\mathcal{L}_{w,b}(x) = \hat{y} = wx + b$$

وزن‌ها بایاس

$$\hat{y} = \sum_{j=1}^{d} w_j x_j + b \times 1$$

$$b \to w_0 \qquad 1 \to x_0$$

$$\hat{y} = \sum_{j=0}^{d} w_j x_j$$

$$X = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & \dots \\ 1 & 2 & 3 & 4 & 6 & \dots \end{bmatrix}$$

$$w = [\underset{w_0}{-}, \underset{w_1}{-}]$$

$$w_0 \to b$$

for epoch in range(——):

y_hat = w.dot(x)

$$\underset{m \times k}{C} = \underset{m \times n}{A} \cdot \underset{n \times k}{b}$$

ستون   سطر

معادلات نرمال

$$y = Xw$$

$$Xw = y$$

$$X^T X w = X^T y$$

$$\underbrace{\cancel{(X^T X)^{-1} X^T X}}_{1} w = (X^T X)^{-1} X^T y$$

$$w = (X^T X)^{-1} X^T y$$

```
w = np.linalg.inv(X.T.dot(X)).dot(X.T).dot(y)
```

مقایسه روش گرادیان کاهشی با معادله نرمال

گرادیان کاهشی	معادله نرمال
نیاز به انتخاب پارامتر یادگیری	نیاز به پارامتر یادگیری ندارد
به تعداد تکرار زیاد	نیاز به تکرار ندارد
برای مقادیر بزرگ n تر خوب کار می‌کند	برای مقادیر بزرگ n کند می‌شود
	ممکن است ماتریس مورد نظر معکوس پذیر نباشد

معادله نرمال — محاسن: نیاز به پارامتر یادگیری ندارد؛ نیاز به تکرار ندارد.

معادله نرمال — معایب: برای مقادیر بزرگ n کند می‌شود؛ ممکن است ماتریس مورد نظر معکوس پذیر نباشد.