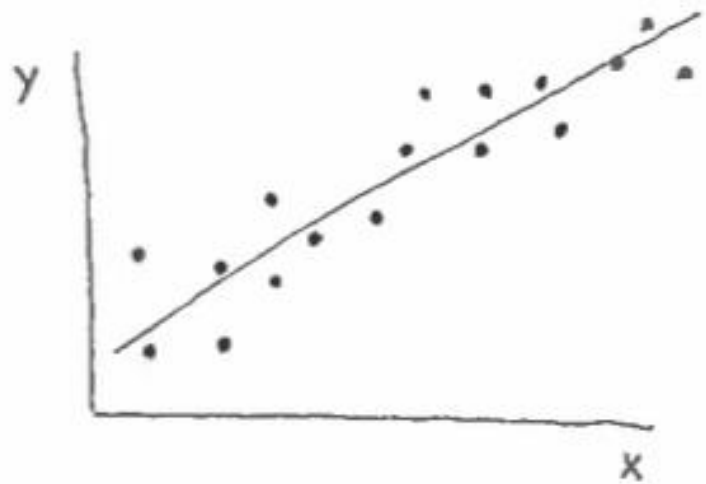
Gradient Descent

E190BO

Sept 2, 2026

Goal: fit regression line to data



Data (x_i, y_i), $i = 1, 2, \ldots, N$

Model $\hat{y}_i = a x_i + b$

Loss $MSE = \frac{1}{N} \sum_{i=1}^{N} (y_i - \hat{y}_i)^2$

Q: How to select model weights?

Approach #1: Analytical

 closed form solution

 only works with simple models

Approach #2: Gradient Descent

 can work with complex models

 only guarantees local minimum

 initialization & settings matter

Gradient descent is an iterative approach to optimization.

1. start with random guess
2. take step in direction of negative gradient
3. repeat until convergence

Pseudocode

pick $w^{(0)}$ randomly

for i in range(numIterations):

$$w^{(i+1)} = w^{(i)} - \gamma \cdot \frac{de}{dw}\Big|_{w=w^{(i)}}$$

iterate until convergence

$w^{(i+1)}$: updated estimate

$w^{(i)}$: current estimate

γ: learning rate

$\frac{de}{dw}$: gradient of loss w.r.t. parameters

$w = w^{(i)}$: evaluate gradient at the current estimate

$-$: gradient is steepest ascent, neg. gradient is steepest descent

Exer : (a) derive gradient

(b) implement in notebook

(c) plot MSE vs iteration for $\gamma = 1, 0.1, 0.01$

(a) $e = \frac{1}{N} \sum_{i=1}^{N} (y_i - ax_i - b)^2$

$\frac{de}{da} = \frac{1}{N} \sum_{i=1}^{N} 2(y_i - ax_i - b)(-x_i)$

$\frac{de}{db} = \frac{1}{N} \sum_{i=1}^{N} 2(y_i - ax_i - b)(-1)$

(b)

```
w = [0,0]
for i in range(numIterations):
    agrad = np.mean(-2 * x * (y - a*x - b))
    bgrad = np.mean(-2 * (y - a*x - b))
    w = w - γ * [agrad, bgrad]
```

(a, b)

Goal: fit regression line to data (m-dim)

	scalar	matrix	comments
Data	$(\bar{x}_i, y_i)$, $i=1,2,\dots,N$ $\bar{x}_i \in \mathbb{R}^m$, $y_i \in \mathbb{R}$	$X \in \mathbb{R}^{N\times m}$, $\bar{y} \in \mathbb{R}^{N\times 1}$	• m features, scalar output, N data samples
Model	$\hat{y}_i = \sum_{j=0}^{m-1} w_j x_{ij}$	$\hat{\bar{y}} = X\bar{w}$, $\bar{w} \in \mathbb{R}^{m\times 1}$	◦ can add constant 1 as feature to handle offset
Loss	$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i-\hat{y}_i)^2$	$\frac{1}{N}(\bar{y}-\hat{\bar{y}})^T(\bar{y}-\hat{\bar{y}})$	

Derive gradient:

$$e = \frac{1}{N} (\bar{y} - X\bar{w})^T (\bar{y} - X\bar{w})$$

$$= \frac{1}{N} \bar{y}^T \bar{y} - \frac{2}{N} \bar{w}^T X^T \bar{y} + \frac{1}{N} \bar{w}^T X^T X \bar{w}$$

$$\frac{\partial e}{\partial w} = -\frac{2}{N} X^T \bar{y} + \frac{2}{N} X^T X \bar{w}$$

Gradient Descent:

initialize w randomly

for i in range(numIterations):

$$wgrad = -\frac{2}{N} X^T y + \frac{2}{N} X^T X \bar{w}$$

$$w = w - \gamma \cdot wgrad$$

Key ideas:
- gradient descent is a flexible + powerful way to fit a model
- define model & loss → derive gradient → gradient descent