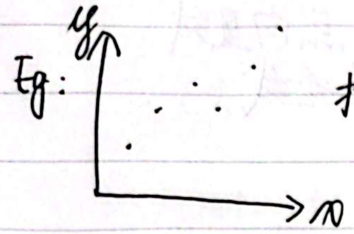
No.
Date

# Regression 回归

Goal: Given $\{(\vec{x}, y)\}$ dataset, learn a function:
$y' = h(\vec{x})$, 应与 $y$ 接近

Eg:



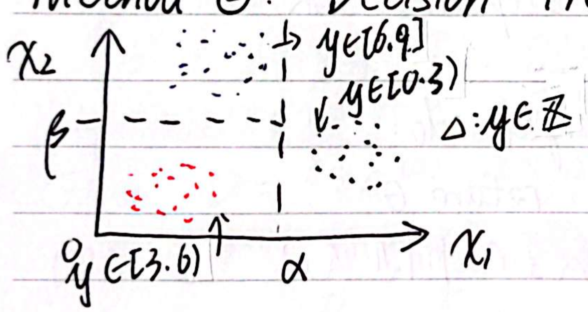

找函数拟合左曲线

## Method ①: K-NN 回归

$k=1$: pick the nearest $x$, return its $y$

or: $k=2$: weighted / unweighted 效果见下


## Method ②: Decision Tree Regression




$x_1 < \alpha$
yes → $x_2 < \beta$
No → range $[0,3)$ ⇒ 1

$x_2 < \beta$
yes → range $[3,6)$ ⇒ 4
No → range $[6,9]$ ⇒ 7.5

划分 $x_i$; 然后 $\{y_i\}$ 聚为一类, 看 mean (Decision Tree 分至最后时)

## Method ③: Linear Functions, Residuals & MSE

⚠ Linear functions ≠ Linear decision Boundaries

Def: Regression is predicting real-valued outputs

$D = \{(\vec{x}^{(i)}, y^{(i)})\}_{i=1}^{n}$, $\vec{x}^{(i)} \in \mathbb{R}^M$, $y^{(i)} \in \mathbb{R}$

Eg: $y = w^T x + b$ Eg: $y = \mathrm{sign}(w^T x + b)$

目标

Key idea: Find the linear function $h$ (with $w, b$) that minimizes the $\boxed{\text{squares of residual}}$ for training set.

Residual: $e_i = |y^{(i)} - h(w^T x^{(i)} + b)|$ 不是点向直线作垂线

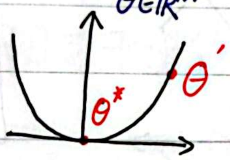
MSE: $Loss = \frac{1}{n}\sum_{i=1}^{n} e_i^2$

有MSE有什么用? The big picture: Optimization for ML

Def: Given function: $J(\theta)$, $J: \mathbb{R}^M \to \mathbb{R}$

Goal: $\hat{\theta} = \underset{\theta \in \mathbb{R}^M}{\arg\min}\, J(\theta)$

形象理解:




在θ'如何一步步到θ*呢?

Naive one: 随机采样θ

☆: Gradient Descent: $\nabla J(\theta) = \begin{bmatrix} \frac{\partial J(\theta)}{\partial \theta_1} \\ \vdots \\ \frac{\partial J(\theta)}{\partial \theta_m} \end{bmatrix}$ 梯度下降!

$\vec{\theta} \leftarrow \vec{\theta} - \delta_t \nabla J(\vec{\theta})$. where $\delta_t$ is step size, $\in \mathbb{R}$

Pseudocode: procedure GD($J, \theta^{(0)}$):

$\theta \leftarrow \theta^{(0)}$; while not converged do:

$\theta \leftarrow \theta - \gamma \nabla_\theta J(\theta)$ ; reture $\theta$

那么 not converged => 如何判定θ已收敛至θ*附近呢?

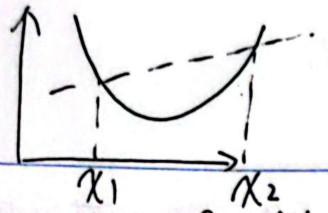
△: $\|\nabla_\theta J(\theta)\|_2 \leq \varepsilon$

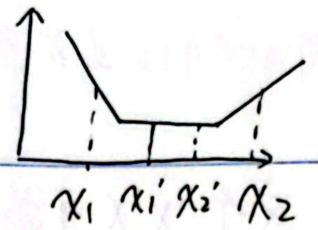
而MSE Loss: $J^{(i)}(\theta) = (y^{(i)} - \theta^T x^{(i)})^2$, $J(\theta) = \frac{1}{N}\sum_{i=1}^{N} J^{(i)}(\theta)$

则 $\frac{d}{d\theta_k} J^{(i)}(\theta) = 2(\theta^T x^{(i)} - y^{(i)}) \frac{d}{d\theta_k}(\theta^T x^{(i)} - y^{(i)})$

$= 2(\theta^T x^{(i)} - y^{(i)}) \frac{d}{d\theta_k}\left(\sum_{j=1}^{K} \theta_j x_j^{(i)} - y^{(i)}\right) = 2(\theta^T x^{(i)} - y^{(i)}) x_k^{(i)}$

$\therefore \frac{d}{dk} J(\theta) = \sum_{i=1}^{N} (\theta^T x^{(i)} - y^{(i)}) x_k^{(i)}$, $\nabla_\theta J(\theta) = \begin{bmatrix} \frac{d}{d\theta_1} J(\theta) \\ \vdots \\ \frac{d}{d\theta_m} J(\theta) \end{bmatrix} = \sum_{i=1}^{N} (\theta^T x^{(i)} - y^{(i)}) x^{(i)}$




No.

Date

Optimization for ML:

Convexity: A function $f: \mathbb{R}^D \to \mathbb{R}$, convex if:
$\forall x^{(1)} \in \mathbb{R}^D, x^{(2)} \in \mathbb{R}^D$, $c \in [0,1]$, 有:

$$f(cx^{(1)} + (1-c)x^{(2)}) \le cf(x^{(1)}) + (1-c)f(x^{(2)})$$

上述为 not strictly convex*; 下述为 strictly*: →如fig1
$c \in (0,1)$, 有: $f(cx^{(1)} + (1-c)x^{(2)}) < cf(x^{(1)}) + (1-c)f(x^{(2)})$
*: 如fig2, 取$x_1'$ $x_2'$, 上面这行实现不了!

对于凸函数来说: global min: $f(x^*) \le f(x)$, $\forall x \in \mathbb{R}^D$
local min: $\exists \varepsilon$ s.t. $f(x^*) \le f(x)$. 且 s.t. $\forall x$. $\|x - x^*\|_2 < \varepsilon$

Closed form Optimization:
key: 找到 $\theta^*$, s.t. $\nabla J(\theta^*) = 0$. 其中J凸
若无法确保J凸. 则找 $\{\theta_i\}$, s.t. $\nabla J(\theta_i) = 0$ for $i \in$ range
并确认它们是极大 or 小值

Given $D = \{(x^{(n)}, y^{(n)})\}_{n=1}^N$, 令: $X = \begin{bmatrix} 1 & x_1^{(1)} & \cdots & x_D^{(1)} \\ 1 & x_1^{(2)} & \ddots & \vdots \\ \vdots & \vdots & & \\ 1 & x_1^{(N)} & & x_D^{(N)} \end{bmatrix} \in \mathbb{R}^{N \times (D+1)}$
而 $y = [y^{(1)}, \dots, y^{(N)}] \in \mathbb{R}^N$ 是目标:

$$J(\theta) = \frac{1}{N}\sum_{i=1}^N \overset{*}{\frac{1}{2}} (y^{(i)} - \theta^T x^{(i)})^2 = \frac{1}{2N}\sum_{i=1}^N (x^{(i)T}\theta - y^{(i)})^2$$

$$= \frac{1}{2N}(X\theta - y)^T(X\theta - y) = \frac{1}{2N}(\theta^T X^T X\theta - 2\theta^T X^T y + y^T y)$$

$$\nabla_\theta J(\theta)^* = \frac{1}{2N}(2X^T X\theta - 2X^T y) = 0 \quad \therefore X^T X\hat{\theta} = X^T y$$

$$\hat{\theta} = (X^T X)^{-1} X^T y$$ *: 矩阵求导!!

*: $\frac{1}{2}$ 无伤大雅; $J(\theta)$ 含义为: 最小Mean Square Error!!

△: 原: $w^T x + b$, 令 $x = \begin{bmatrix} 1 \\ x \end{bmatrix}$, $\theta = \begin{bmatrix} b \\ w \end{bmatrix}$, 则 => $\theta^T x$

No. *：直觉上：$N=2$，那么超平面显然有很多种划分

Date

$\hat{\theta}=(X^TX)^{-1}X^Ty$，这个 $X^TX$ 可逆么？一般认为 $N>>D+1$ 时，几乎总是 full-rank 的*。但是关键在于计算 $X^TX$ 逆时间复杂度多少？算 $X^TX$：$O(ND^2)$；算它的逆：$O(D^3)$

那么，有没有不那么精确但复杂度能够接受的近似求法呢？

GD & SGD:

Gradient Descent:

$\theta \leftarrow \theta^{(0)}$
while not converged do
　$\theta \leftarrow \theta - \gamma \nabla J(\theta)$
return $\theta$

$N$ 个项，$\theta \in \mathbb{R}^M$
$\Downarrow$
△：$J(\theta)=\frac{1}{N}\sum_{i=1}^{N} J^{(i)}(\theta)$ *

而 Stochastic Gradient Descent:

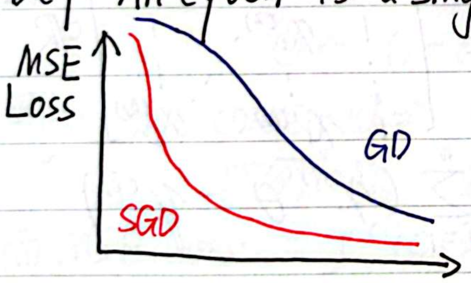
$\theta \leftarrow \theta^{(0)}$
while not converged do
　$\boxed{i \sim \text{uniform}(\{1,2,\cdots,N\})}$
　$\theta \leftarrow \theta - \gamma \nabla_\theta J^{(i)}(\theta)$
return $\theta$

实质：原来用全部样本，现在一步用一个

why SGD 能 work？因为 $E[\nabla_\theta J^{(i)}(\theta)]=\sum_{i=1}^{N} p(x^{(i)},y^{(i)})\nabla_\theta J^{(i)}(\theta)$
$=\frac{1}{N}\sum_{i=1}^{N}\nabla_\theta J^{(i)}(\theta)=\nabla_\theta J(\theta)$

黑：GD
蓝：SGD

Def: An epoch is a single passing through training data




GD: One update per epoch

SGD: $N$ update per epoch

| Theoretical Comparison: | step to convergence | Computation per step |
| --- | --- | --- |
| GD | $O(\log 1/\varepsilon)$ | $O(NM)$ |
| SGD | $O(1/\varepsilon)$ | $O(M)$ |

理论上 SGD 收敛更慢，但实践中更快

那么在上一面的 Linear Regression 情景中，$J(\theta)=\frac{1}{N}\sum_{i=1}^{N}(y^{(i)}-\theta^Tx^{(i)})^2$

则 SGD:　$\theta \leftarrow \theta^{(0)}$
while not converged do:
　for $i \in$ shuffle $(\{1,\cdots,N\})$ do ⇐ 为了保证 $N$ 次可过一个 epoch 故是 range(1,N) shuffle 以模拟 Uniform
　　$\theta \leftarrow \theta - \gamma(\theta^Tx^{(i)}-y^{(i)})x^{(i)}$
return $\theta$

GD: $\theta \leftarrow \theta - \gamma \frac{1}{N}\sum_{i=1}^{N}(\theta^Tx^{(i)}-y^{(i)})x^{(i)}$ ⇐

Campus

扫描全能王 创建