

<01> Intro to ML

- well-posed 有解的
- ill-posed 给黑白照片上色（解不唯一），比如给花上色
有几种可能

<02> supervised Learning

- 师傅代表标准，我代表学习的系统
- CV 领域基本上是监督学习

<03> 检测分类

- 物体检测
- 特征点检测
- 人脸识别（年龄，性别）
- 输出的是一个数（都是回归类问题）

<04> 分类

- 分割也算分类
- 语义分割（同是鱼，就分为一类） instance seg
- 识别：
分类：
检测：
- 公安局人脸识别数据量是亿，不能是分类否则有上亿分类。
- 人脸识别： 人脸 \rightarrow CNN \rightarrow Feature point \longrightarrow 库
 \rightarrow 相似度

<05> 无监督学习 (Unsupervised Learning)

- 有 1000000 个视频帧，进行分类（短时间内）
- 分类问题：
Label (标注) 成本太高

<06> clustering (聚类)

- 抽出 Feature Vector, 看 distance
- 流程:
 - 1) 5个 centre
 - 2) 数据和 centre 比较距离

<07> Linear Regression

$$y = ax + b \quad (\text{Model})$$

↑ ↗

找到 a, b 的过程, 叫训练

$$h_{\theta}(x) = \theta_0 + \theta_1 x \quad \underline{\text{Hypotheses}} \quad (\text{假设})$$

如何评价函数的好坏?

cost : Difference between Hypotheses and reality
真实和 Hypotheses 差距

lost : 一个结果 lost 越小, Model ↑

cost 代价函数

model : $h_{\theta}(x^i) = \theta_0 + \theta_1 x^i$ → 数据个数

cost : $J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^m (h_{\theta}(x^i) - y^i)^2$ → 距离(误差)
 ↓
 目的 (Goal)
 ——————
 求和
 预测值 真实值
平滑的: 该函数绝对值
 第二项可以和 m 约去

$$\underset{\theta_0, \theta_1}{\text{Minimize}} J(\theta_0, \theta_1)$$

怎么解求 Min 最小 (求导)

梯度下降

$$\theta_j \leftarrow \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$$

θ_j 代价函数某一个参数, 减去该代价函数对该参数的导数上一个学习率 α .

```
while not converge {
   $\theta_j = \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta) \quad \text{for } j = 0 \text{ and } j = 1$ 
}
```

$$\frac{\partial}{\partial \theta_0} J(\theta_0) = \frac{1}{m} \sum_{i=1}^m (h_{\theta}(x^i) - y^i)$$

$$\frac{\partial}{\partial \theta_1} J(\theta_1) = \frac{1}{m} \sum_{i=1}^m (h_{\theta}(x^i) - y^i) \cdot x^i$$

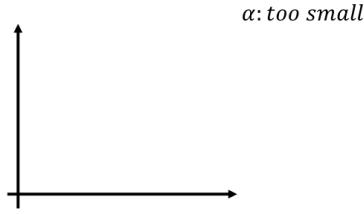
C. Linear Regression:

C3. Illustration-Gradient Descent

$J(\theta_j)$: Quadratic Function Cost Function

$\frac{d}{d\theta_j} J(\theta_j)$: Slope 余斜率

$\theta_j = \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$; Gradient Descent

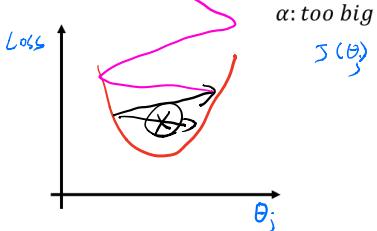


Hypothesis: $h_{\theta}(x^i) = \theta_0 + \theta_1 x^i = \theta^T x^i$

Parameters: θ_0, θ_1

Cost Func: $J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^m (h_{\theta}(x^i) - y^i)^2$

Goal: $\underset{\theta_0, \theta_1}{\text{minimize}} J(\theta_0, \theta_1)$



1) 为什么会荡？ 斜率符号变了，一定是往上荡

$$\theta_j = \theta_j - (1 \times \alpha) \Rightarrow \theta_j \rightarrow \infty$$

2) 梯度爆炸

Loss 在短时间内迅速增大的现象

3) 往下荡(左右) (α 不是那么大)

会一直往下走，梯度振荡，也会收敛

{ α 过大 \Rightarrow 梯度爆炸
 α 不大(没那么大) \Rightarrow 振荡收敛
 α 正合适 \Rightarrow 正常收敛
 α 过小 \Rightarrow 收敛过慢

局部最优

<08> 如何选取学习率

○ 调参(试一试), 如 $\alpha = 0.01$

<09> 真实的情况, 影响因素很多
 x_1, x_2, x_3, \dots 每个因素都要有
 一个 θ

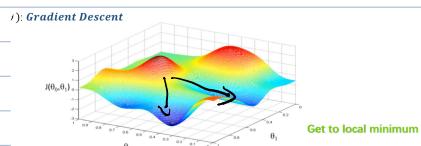
Price:

x_1 Size, Community, Floor, Far, School, Hospital,

x_2 Subway, Traffic, Business, Entertainment,

x_3 Market, Policy, $x_0 = 1$

$$h_{\theta}(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \dots + \theta_n x_n = \theta^T x$$



<10> 多个参数的 Hypothesis

$$\begin{cases} h_{\theta}(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \dots + \theta_n x_n = \theta^T x \\ h_{\theta}(x^i) = \theta_0 + \theta_1 x_1^i + \theta_2 x_2^i + \dots + \theta_n x_n^i = \theta^T x^i \end{cases} \rightarrow \text{第几个样本 (上标)}$$

$$\theta_j \leftarrow \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$$

$$= \theta_j - \alpha \frac{1}{m} \sum_{i=1}^m (h_{\theta}(x^i) - y^i) \cdot x_j^i$$

这周公式一定要掌握

<11> Feature Scaling

① 为什么长这样 zigzag

$$J(\theta_j) = \frac{1}{2m} \sum_{i=1}^m (h_{\theta_j}(x^i) - y^i)^2$$

○ 代价函数的导数

○ 两个参数所以是椭圆

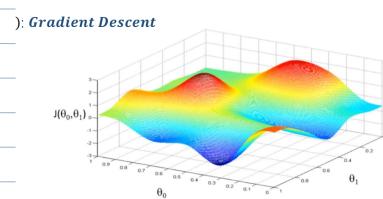
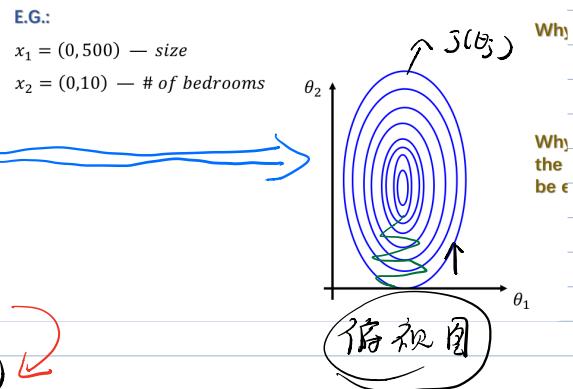
○ 由于 $\theta_1 > \theta_2$ 所以是这种细长的椭圆

② 为什么会 zigzag? (两次原因不一样)

两个学习率不一样，两个方向梯度不一样

③ 为什么 zigzag 没飞?

还有个分量往前走，不够彻底飞出去。



<12> Normalization (规范化)

$$\begin{cases} x_1 = (0, 500) & x_2 = (0, 10) \\ x_j' = \frac{x_j - \bar{x}_j}{s_j} & s.t. N \sim (0, 1) \end{cases}$$

④ 为什么不直接求导，而用梯度下降？



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

$$J(\theta) = ((X\theta)^T - y^T)(X\theta - y)$$

$$J(\theta) = (X\theta)^T X\theta - (X\theta)^T y - y^T (X\theta) + y^T y$$

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

$$X^T X\theta = X^T y$$

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

<14> Sigmoid Function

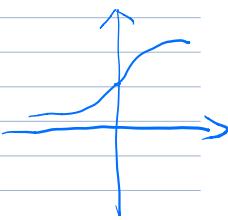
$$h_{\theta}(x) = \theta^T x \quad (\text{Hypothesis})$$

$$\text{Sigmoid: } h_{\theta}(x) = g(\theta^T x)$$

$$= g(z)$$

$$= \frac{1}{1 + e^{-z}}$$

$$= \frac{1}{1 + e^{-\theta^T x}}$$



$$h_{\theta}(x) = \theta_0 x_0 + \theta_1 x_1 + \dots + \theta_n x_n$$

$$J(\theta_{0:n}) = \frac{1}{2m} \sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)})^2$$

$$\begin{pmatrix} \theta_0 \\ \theta_1 \\ \dots \\ \theta_n \end{pmatrix} \in \mathbb{R}^{n+1}$$

$$h_{\theta}(x) = \theta^T x$$

<15> 为什么 sigmoid 的 Loss Function 求导之后和 Linear 的一样？

Hypothesis:

$$h_{\theta}(x) = \frac{1}{1 + e^{-\theta^T x}}$$

算出来也一样

Cost Function:

$$J(\theta) = -\frac{1}{m} \sum_{i=1}^m [y^i \log(h_{\theta}(x^i)) + (1-y^i) \log(1-h_{\theta}(x^i))]$$

Gradient Descent: (求导)

$$\frac{\partial}{\partial \theta_j} J(\theta) = \frac{1}{m} \sum_{i=1}^m (h_{\theta}(x^i) - y^i) \cdot x_j \quad \text{和之前完全一样}$$

<16> 公式推导 ($g(z)$ 推导) 对 z 导，这里也有 θ

$$h_{\theta}(x) = \frac{1}{1 + e^{-\theta^T x}} = \frac{1}{1 + e^{-z}}$$

$$g(z) = \left[\frac{1}{1 + e^{-z}} \right]' = \frac{e^{-z}}{(1 + e^{-z})^2} = \frac{1}{1 + e^{-z}} \cdot \frac{e^{-z}}{1 + e^{-z}}$$

$$\begin{aligned} (1 + e^{-z})' &= (1 + \frac{1}{e^z})' \\ &= -\frac{e^z}{e^{2z}} \\ &= -e^{-z} \end{aligned}$$

$$= g(z) \cdot (1 - g(z))$$

结论: $g'(z) = g(z)(1 - g(z))$

$$<17> J(\theta) = y \log(g) + (1-y) \log(1-g) \quad (\text{对数导数})$$

(?) 为什么 $J(\theta)$ 这么写

$$\frac{\partial J(\theta)}{\partial \theta} = y \frac{g'}{g} + (1-y) \frac{(1-g')}{1-g} = y \frac{1}{g} g' + (1-y) \frac{g'}{1-g}$$

$$= \left(\frac{y}{g} + \frac{1-y}{1-g} \right) g' = \left[\frac{y(1-g)}{g(1-g)} + \frac{g(y-1)}{g(1-g)} \right] g'$$

$$= \frac{y(1-g) - (1-y)g}{g(1-g)} g' \quad (g') = g(1-g)$$

$$= (y-g) \underset{\leftarrow \text{因为是对应的}}{\geq} \leftarrow \text{因为是对应的} \quad g(z) \underset{z=\theta^T x}{\hookrightarrow}$$

<18> 分类 (Multi-classes)

<19> 代码化的 Linear Regression

○ 预测值 ○ 梯度 dw, db ○ 总的 $\frac{\partial J(\theta)}{\partial \theta}$

def inference(w, b, x):

$$\text{pred_y} = w * x + b$$

return pred_y (预测值)

get dw, db
def gradient(pred_y, gt_y, x):

$$\text{diff} = \text{pred_y} - \text{gt_y}$$

$$\boxed{\begin{aligned} dw &= \text{diff} * x \\ db &= \text{diff} \end{aligned}}$$

return dw, db

注解 $dw = \text{diff} * x$ 由来 $\rightarrow h_{\theta}(x) = \theta^T x$

$$\frac{\partial J(\theta)}{\partial \theta} = \left[\frac{1}{2m} \sum_{i=1}^m (\text{pred_y} - \text{gt_y})^2 \right]$$

$$= (\text{pred_y} - \text{gt_y}) x$$

求和后面加起来

$$J(\theta) = \frac{1}{2m} \sum_{i=1}^m (h_{\theta}(x) - y(\theta))^2$$

def cal→step→gradient C batch→X→list,