SLE LAB

Kaggle Pro 银牌计划

优化论基础与决策树算法

主讲人：黄老师

机器学习基本概念

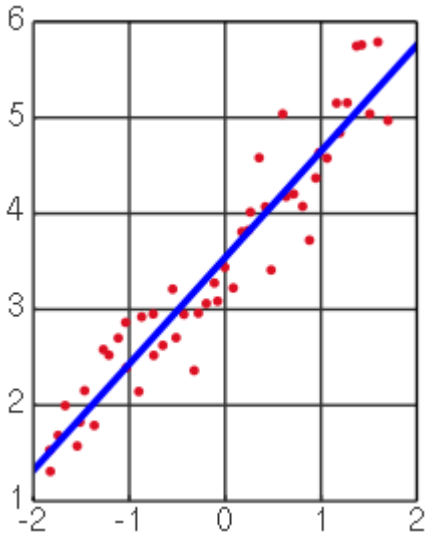
什么是机器学习?

- 根据现有数据找规律
- 基于大数据的统计学 Statistics based on Big Data

• 计算机自动从数据中发现规律，并应用于解决新问题

线性回归就是最简单的机器学习模型之一

线性回归真的简单吗?



高中课本上的线性回归

什么是一元线性回归

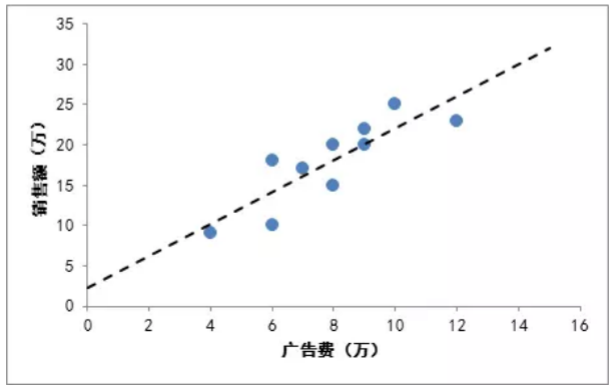
回归分析（Regression Analysis)是确定两种或两种以上变量间相互依赖的定量关系的一种统计分析方法。在回归分析中，只包括一个自变量和一个因变量，且二者的关系可用一条直线近似表示，这种回归分析称为一元线性回归分析。举个例子来说吧：

比方说有一个公司，每月的广告费用和销售额，如下表所示：

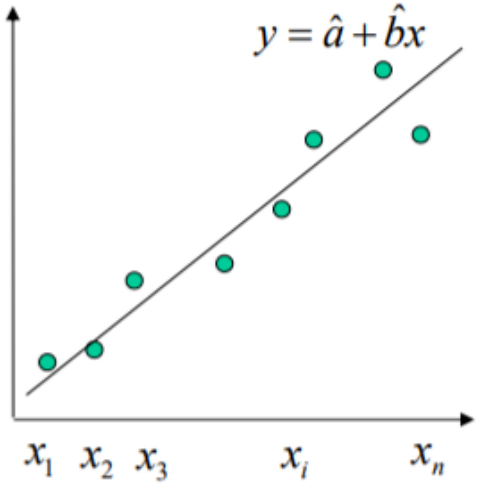
广告费（万）	4	8	9	8	7	12	6	10	6	9
销售额（万）	9	20	22	15	17	23	18	25	10	20

案例数据

如果我们把广告费和销售额画在二维坐标内，就能够得到一个散点图，如果想探索广告费和销售额的关系，就可以利用一元线性回归做出一条拟合直线：



拟合直线



$$\text{MSE}: Q(a,b) = \sum_{i=1}^{n} (y_i - a - bx_i)^2$$

求估计 $\hat{a}, \hat{b}$, 使 $Q(\hat{a}, \hat{b}) = \min_{a,b} Q(a,b)$

$$\frac{\partial Q}{\partial a} = -2\sum_{i=1}^{n}(y_i - a - bx_i) = 0$$

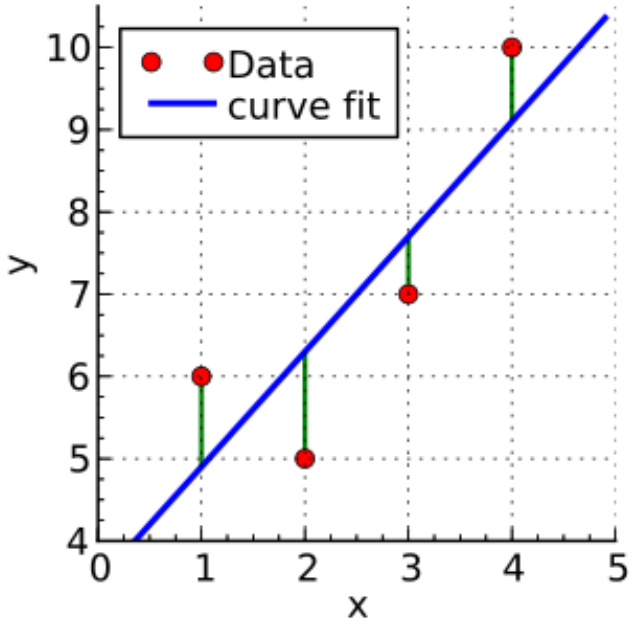
$$\frac{\partial Q}{\partial b} = -2\sum_{i=1}^{n}(y_i - a - bx_i)x_i = 0$$

$$a = \frac{n\sum x_i y_i - \sum x_i \sum y_i}{n\sum x_i^2 - (\sum x_i)^2}$$

$$b = \frac{\sum y_i}{n} - a\frac{\sum x_i}{n}$$

多元线性回归 – 特征变为多元向量

ID	NAME	CLASS	MARK	GENDER
1	John Deo	Four	75	female
2	Max Ruin	Three	85	male
3	Arnold	Three	55	male
4	Krish Star	Four	60	female
5	John Mike	Four	60	female
6	Alex John	Four	55	male
7	My John Rob	Five	78	male
8	Asruid	Five	85	male
9	Tes Qry	Six	78	male
10	Big John	Four	55	female



$$\left\{\left(x_1^{(i)}, x_2^{(i)}, y^{(i)}\right)\right\} \longrightarrow \left\{\left(\mathbf{x}^{(i)}, y^{(i)}\right)\right\}, \mathbf{x}^{(i)} = \begin{bmatrix} x_1^{(i)} \\ x_2^{(i)} \end{bmatrix}$$

- 试图学习： $f(x) = wx + b$ 使得 $f\left(x^{(i)}\right) \approx y^{(i)}$
- 试图学习： $f(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + b$ 使得 $f\left(\mathbf{x}^{(i)}\right) \approx y^{(i)}$
- 核心在于怎么学?

多元线性回归目标函数

- 试图学习: $f(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + b$ 使得 $f\left(\mathbf{x}^{(i)}\right) \approx y^{(i)}$

待求参数 $\overline{\mathbf{w}} = \begin{bmatrix} \mathbf{w} \\ b \end{bmatrix}$

$$f\left(\mathbf{x}^{(1)}\right) = \mathbf{w}^T\mathbf{x}^{(1)} + b = w_1x_1^{(1)} + w_2x_2^{(1)} + \cdots + w_dx_d^{(1)} + b = \left[x_1^{(1)}, x_2^{(1)}, \cdots, x_d^{(1)}, 1\right] \cdot \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_d \\ b \end{bmatrix}$$

$$f\left(\mathbf{x}^{(2)}\right) = \mathbf{w}^T\mathbf{x}^{(2)} + b = w_1x_1^{(2)} + w_2x_2^{(2)} + \cdots + w_dx_d^{(2)} + b = \left[x_1^{(2)}, x_2^{(2)}, \cdots, x_d^{(2)}, 1\right] \cdot \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_d \\ b \end{bmatrix}$$

$$\begin{bmatrix} f\left(\mathbf{x}^{(1)}\right) \\ f\left(\mathbf{x}^{(2)}\right) \end{bmatrix} = \begin{bmatrix} x_1^{(1)} & x_2^{(1)} & \cdots & x_d^{(1)} & 1 \\ x_1^{(2)} & x_2^{(2)} & \cdots & x_d^{(2)} & 1 \end{bmatrix} \cdot \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_d \\ b \end{bmatrix}$$

$$\mathbf{X} = \begin{bmatrix} x_1^{(1)} & x_2^{(1)} & \cdots & x_d^{(1)} & 1 \\ x_1^{(2)} & x_2^{(2)} & \cdots & x_d^{(2)} & 1 \\ x_1^{(3)} & x_2^{(3)} & \cdots & x_d^{(3)} & 1 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ x_1^{(N)} & x_2^{(N)} & \cdots & x_d^{(N)} & 1 \end{bmatrix}_{N\times(d+1)} \Rightarrow \begin{bmatrix} f\left(\mathbf{x}^{(1)}\right) \\ f\left(\mathbf{x}^{(2)}\right) \\ f\left(\mathbf{x}^{(3)}\right) \\ \vdots \\ f\left(\mathbf{x}^{(N)}\right) \end{bmatrix} = \begin{bmatrix} x_1^{(1)} & x_2^{(1)} & \cdots & x_d^{(1)} & 1 \\ x_1^{(2)} & x_2^{(2)} & \cdots & x_d^{(2)} & 1 \\ x_1^{(3)} & x_2^{(3)} & \cdots & x_d^{(3)} & 1 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ x_1^{(N)} & x_2^{(N)} & \cdots & x_d^{(N)} & 1 \end{bmatrix} \cdot \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_d \\ b \end{bmatrix} = \mathbf{X}\overline{\mathbf{w}}$$

- 未知 $\overline{\mathbf{w}} = \begin{bmatrix} \mathbf{w} \\ b \end{bmatrix}$，已知 $\mathbf{X} = \begin{bmatrix} \mathbf{x}^{(1)T} & 1 \\ \vdots & \vdots \\ \mathbf{x}^{(N)T} & 1 \end{bmatrix}_{N\times(d+1)}$，则有

$$\mathbf{y} \approx \mathbf{X}\overline{\mathbf{w}}$$

- 损失函数 $||\mathbf{y} - \mathbf{X}\overline{\mathbf{w}}||_2^2$，求解

$$\min ||\mathbf{y} - \mathbf{X}\overline{\mathbf{w}}||_2^2$$

■ **范数：具有“长度”概念的函数。**

定义：设V是实数域R（或复数域C）上的n维线性空间，对于V中的任意一个向量 α 按照某一确定法则对应着一个实数，这个实数称为该向量 α 的范数记为 $\|\alpha\|$，并且要求范数满足下列条件：

（1）非负性：当 $\alpha \neq 0$，$\|\alpha\| > 0$

（2）齐次性：$\| k\alpha \| = | k | \| \alpha \|$，k为实数(或复数)

（3）三角不等式：$\|\alpha + \beta\| \leq \|\alpha\| + \|\beta\| \quad (\alpha, \beta \in V)$

■ 常用范数：p 范数

$$\|\boldsymbol{x}\|_p = \left(\sum_{i=1}^{n} |x_i|^p \right)^{1/p}, \quad 1 \leq p < +\infty$$

1-范数 $\|\boldsymbol{x}\|_1 = \sum_{i=1}^{n} |x_i|$

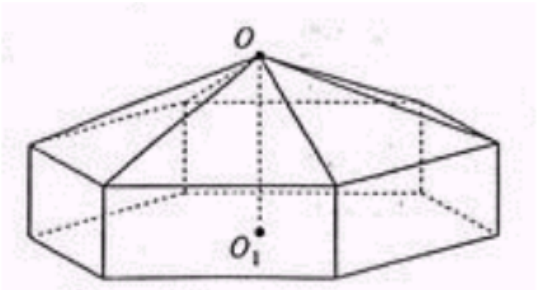
2-范数 $\|\boldsymbol{x}\|_2 = \left(\sum_{i=1}^{n} |x_i|^2 \right)^{1/2} = \sqrt{\boldsymbol{x}^{\mathrm{T}}\boldsymbol{x}}$

无穷范数 $\|\boldsymbol{x}\|_\infty = \max_{1\leq i\leq n} |x_i|$

无约束优化理论：导数为0

例 11．请您设计一个帐篷。它下部的形状是高为 1m 的正六棱柱，上部的形状是侧棱长为 3m 的正六棱锥（如右图所示）。试问当帐篷的顶点 O 到底面中心 o_1 的距离为多少时，帐篷的体积最大？

本小题主要考查利用导数研究函数的最大值和最小值的基础知识，以及运用数学知识解决实际问题的能力。



$$V(x)=\frac{3\sqrt{3}}{2}(8+2x-x^2)\left[\frac{1}{3}(x-1)+1\right]=\frac{\sqrt{3}}{2}(16+12x-x^3)$$

求导数，得 $V'(x)=\frac{\sqrt{3}}{2}(12-3x^2)$；

令 $V'(x)=0$ 解得 x=-2（不合题意，舍去），x=2。

梯度与海森矩阵

- 一阶导数和梯度 (gradient vector)

$$f'(x); \qquad \mathbf{g}(\mathbf{x}) = \nabla f(\mathbf{x}) = \frac{\partial f(\mathbf{x})}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial f(\mathbf{x})}{\partial x_1} \\ \vdots \\ \frac{\partial f(\mathbf{x})}{\partial x_n} \end{bmatrix}$$

- 二阶导数和 Hessian 矩阵

$$f''(x); \qquad \mathbf{H}(\mathbf{x}) = \nabla^2 f(\mathbf{x}) = \begin{bmatrix} \frac{\partial^2 f(\mathbf{x})}{\partial x_1^2} & \frac{\partial^2 f(\mathbf{x})}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f(\mathbf{x})}{\partial x_1 \partial x_n} \cdots \\ \frac{\partial^2 f(\mathbf{x})}{\partial x_2 \partial x_1} & \frac{\partial^2 f(\mathbf{x})}{\partial x_2^2} & & \\ & & \ddots & \\ \frac{\partial^2 f(\mathbf{x})}{\partial x_n \partial x_1} & \frac{\partial^2 f(\mathbf{x})}{\partial x_n \partial x_2} & & \frac{\partial^2 f(\mathbf{x})}{\partial x_n^2} \end{bmatrix} = \nabla \left(\nabla f(\mathbf{x})\right)^T$$

二次型（自学）

在数学中，**二次型（Quadratic form）**是关于一些变量的二次齐次多项式。例如

$$4x^2 + 2xy - 3y^2$$

- 给定矩阵 $\mathbf{A} \in \mathbb{R}^{n\times n}$，函数

$$\mathbf{x}^T\mathbf{A}\mathbf{x} = \sum_{i=1}^{n} x_i \left(\mathbf{A}\mathbf{x}\right)_i = \sum_{i=1}^{n} x_i \left(\sum_{j=1}^{n} a_{ij}x_j\right) = \sum_{i=1}^{n}\sum_{j=1}^{n} x_i x_j a_{ij}$$

$$4x^2 + 2xy - 3y^2 = (x, y)\begin{pmatrix} 4 & 1 \\ 1 & -3 \end{pmatrix}\begin{pmatrix} x \\ y \end{pmatrix} \qquad x^2 + y^2 + 3z^2 - 4xy = (x, y, z)\begin{pmatrix} 1 & -2 & \\ -2 & 1 & \\ & & 3 \end{pmatrix}\begin{pmatrix} x \\ y \\ z \end{pmatrix}$$

梯度计算（自学）

- 向量 $\mathbf{a}$ 和 $\mathbf{x}$ 无关, 则 $\nabla\left(\mathbf{a}^T\mathbf{x}\right)=\mathbf{a}$, $\nabla^2\left(\mathbf{a}^T\mathbf{x}\right)=0$

$$\mathbf{a}^T\mathbf{x}=a_1x_1+a_2x_2+\cdots+a_nx_n$$

$$\nabla\left(\mathbf{a}^T\mathbf{x}\right)=\begin{pmatrix}a_1\\a_2\\\vdots\\a_n\end{pmatrix}=\mathbf{a}$$

- 对称矩阵矩阵 $\mathbf{A}$ 与 $\mathbf{x}$ 无关, 则 $\nabla\left(\mathbf{x}^T\mathbf{A}\mathbf{x}\right)=2\mathbf{A}\mathbf{x}$, $\nabla^2\left(\mathbf{x}^T\mathbf{A}\mathbf{x}\right)=2\mathbf{A}$

多元线性回归目标函数 $\min \|\mathbf{y}-\mathbf{X}\overline{\mathbf{w}}\|_2^2$

$$
\begin{aligned}
f(\overline{\mathbf{w}}) &= \|\mathbf{X}\overline{\mathbf{w}}-\mathbf{y}\|_2^2 = (\mathbf{X}\overline{\mathbf{w}}-\mathbf{y})^T(\mathbf{X}\overline{\mathbf{w}}-\mathbf{y}) \\
&= \overline{\mathbf{w}}^T\mathbf{X}^T\mathbf{X}\overline{\mathbf{w}} - \mathbf{y}^T\mathbf{X}\overline{\mathbf{w}} - \overline{\mathbf{w}}^T\mathbf{X}^T\mathbf{y} + \mathbf{y}^T\mathbf{y} \\
&= \overline{\mathbf{w}}^T\mathbf{X}^T\mathbf{X}\overline{\mathbf{w}} - \mathbf{y}^T\mathbf{X}\overline{\mathbf{w}} - (\mathbf{X}\overline{\mathbf{w}})^T\mathbf{y} + \mathbf{y}^T\mathbf{y} \\
&= \overline{\mathbf{w}}^T\mathbf{X}^T\mathbf{X}\overline{\mathbf{w}} - 2\left(\mathbf{X}^T\mathbf{y}\right)^T\overline{\mathbf{w}} + \mathbf{y}^T\mathbf{y}
\end{aligned}
$$

$$
\nabla f(\overline{\mathbf{w}}) = 2\mathbf{X}^T\mathbf{X}\overline{\mathbf{w}} - 2\mathbf{X}^T\mathbf{y} = 2\mathbf{X}^T(\mathbf{X}\overline{\mathbf{w}}-\mathbf{y})
$$

线性回归解法一：梯度为0

- 试图学习：$f(\mathbf{x}) = \mathbf{w}^T\mathbf{x}+b$ 使得 $f\left(\mathbf{x}^{(i)}\right) \approx y^{(i)}$
- 未知 $\overline{\mathbf{w}} = \begin{bmatrix} \mathbf{w} \\ b \end{bmatrix}$，已知 $\mathbf{X} = \begin{bmatrix} \mathbf{x}^{(1)T} & 1 \\ \vdots & \vdots \\ \mathbf{x}^{(N)T} & 1 \end{bmatrix}_{N\times(d+1)}$，则有

$$
\mathbf{y} \approx \mathbf{X}\overline{\mathbf{w}}
$$

- 损失函数 $\|\mathbf{y}-\mathbf{X}\overline{\mathbf{w}}\|_2^2$，求解

$$
\min \|\mathbf{y}-\mathbf{X}\overline{\mathbf{w}}\|_2^2
$$

- $g(\overline{\mathbf{w}}) = 0 \Rightarrow 2\mathbf{X}^T(\mathbf{X}\overline{\mathbf{w}}-\mathbf{y}) = 0 \Rightarrow \overline{\mathbf{w}}^* = \left(\mathbf{X}^T\mathbf{X}\right)^{-1}\mathbf{X}^T\mathbf{y}$

梯度为0的局限性

计算 $f(x) = x^4 + \sin(x^2) - \ln(x)e^x + 7$ 的导数

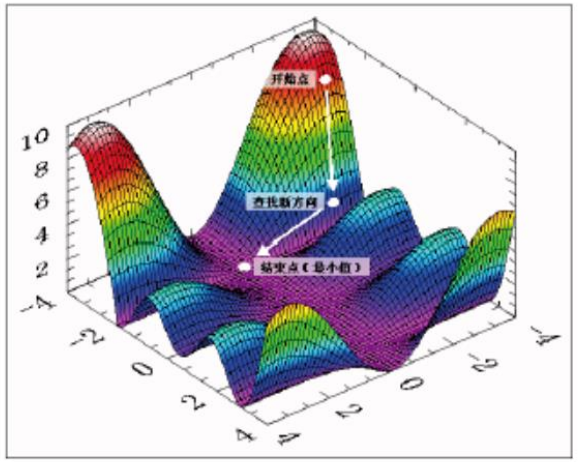
$$
\begin{aligned}
f'(x) &= 4x^{(4-1)} + \frac{d\left(x^2\right)}{dx}\cos(x^2) - \frac{d\left(\ln x\right)}{dx}e^x - \ln(x)\frac{d\left(e^x\right)}{dx} + 0 \\
&= 4x^3 + 2x\cos(x^2) - \frac{1}{x}e^x - \ln(x)e^x
\end{aligned}
$$

思考求 $f'(x) = 0$?

无约束优化迭代法

迭代法的基本结构 (最小化 $f(\mathbf{x})$)

1. 选择一个初始点，设置一个 convergence tolerance ϵ，计数 $k = 0$
2. 决定搜索方向 $\mathbf{d}_k$, 使得函数下降. (核心)
3. 决定步长 α_k 使得 $f(\mathbf{x}_k + \alpha_k \mathbf{d}_k)$ 对于 $\alpha_k \geq 0$ 最小化，构建 $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{d}_k$
4. 如果 $\|\mathbf{d}_k\| < \epsilon$，则停止输出解 $\mathbf{x}_{k+1}$；否则继续重复迭代.



梯度下降法

泰勒级数展开 (标量和向量)

- 输入为标量的泰勒级数展开

$$f(x_k+\delta) \approx f(x_k) + f'(x_k)\,\delta + \frac{1}{2}f''(x_k)\,\delta^2 + \cdots + \frac{1}{k!}f^k(x_k)\,\delta^k + \cdots$$

- 输入为向量的泰勒级数展开

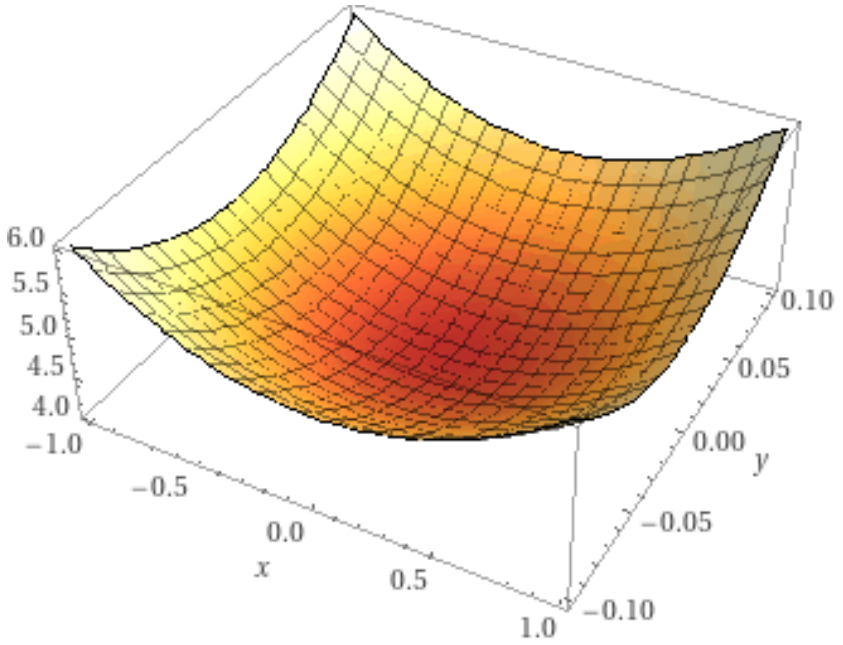
$$f(\mathbf{x}_k+\boldsymbol{\delta}) \approx f(\mathbf{x}_k) + \mathbf{g}^T(\mathbf{x}_k)\,\boldsymbol{\delta} + \frac{1}{2}\boldsymbol{\delta}^T\mathbf{H}(\mathbf{x}_k)\,\boldsymbol{\delta}$$

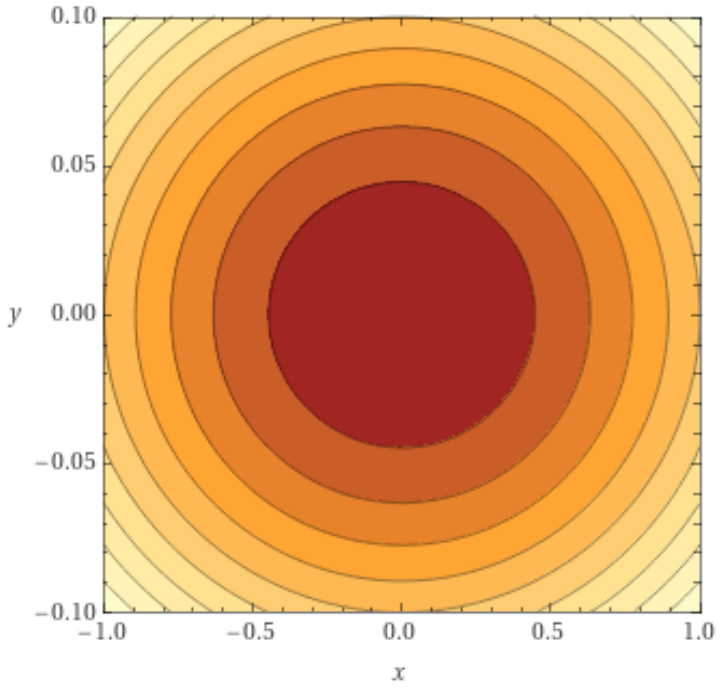
$$f(\mathbf{x}_k+\mathbf{d}_k) \approx f(\mathbf{x}_k) + \mathbf{g}^T(\mathbf{x}_k)\,\mathbf{d}_k$$

- 需要 $f(\mathbf{x}_k+\mathbf{d}_k)\downarrow$，则 $f(\mathbf{x}_k)$ 加个负数
- 回忆两个向量的内积，$\mathbf{a}\cdot\mathbf{b} = \mathbf{a}^T\mathbf{b} = \|\mathbf{a}\|\,\|\mathbf{b}\|\cos\theta$

梯度下降

使用梯度下降算法求解 $f(x, y) = x^2 + 100y^2 + 4$ 的最小值。





线性回归解法二：梯度下降法

- 梯度下降法

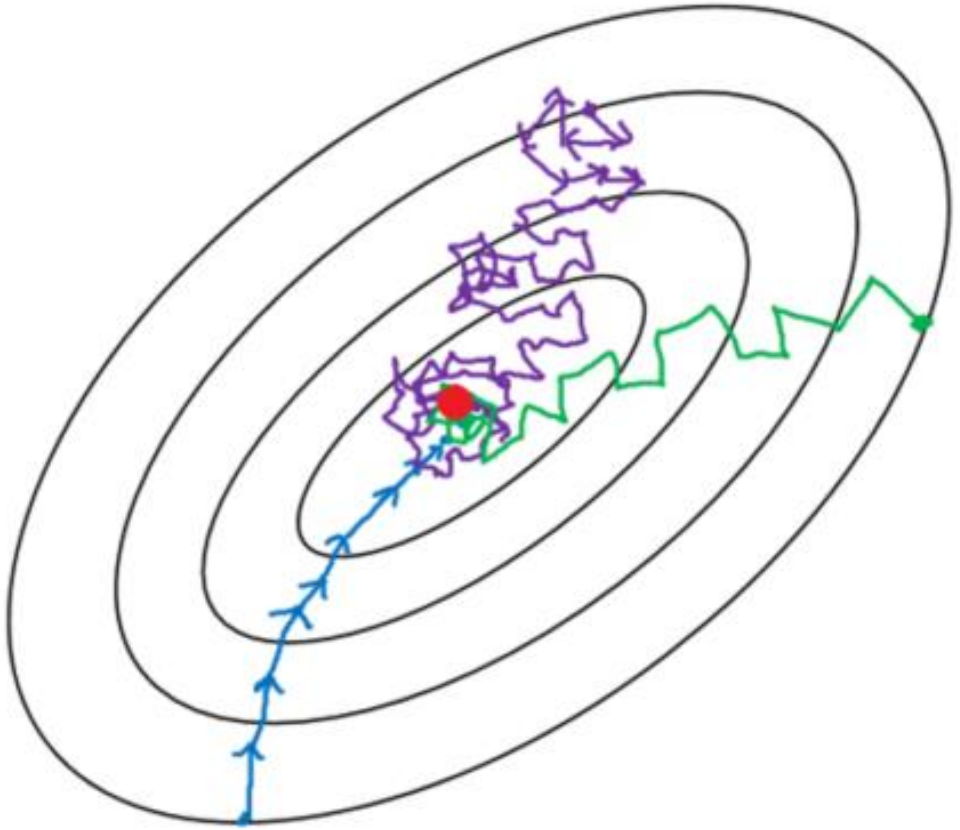
$$\begin{aligned} \mathbf{g}(\overline{\mathbf{w}}) &= 2\mathbf{X}^T(\mathbf{X}\overline{\mathbf{w}} - \mathbf{y}) \\ &= 2\sum_{i=1}^{N} \mathbf{x}^{(i)} \left(\mathbf{w}^T\mathbf{x}^{(i)} - y^{(i)}\right) \end{aligned}$$

$$\overline{\mathbf{w}} \leftarrow \overline{\mathbf{w}} - \alpha \mathbf{g}(\overline{\mathbf{w}})$$

- 随机梯度下降法（实际中很有用）

$$\left\{ i = 1 : N,\ 2\mathbf{x}^{(i)} \left(\mathbf{w}^T\mathbf{x}^{(i)} - y^{(i)}\right) \right\}$$

三种梯度下降方式效果示意



Batch gradient descent
Mini-batch gradient Descent
Stochastic gradient descent

Preparing your data for training with DataLoaders

The `Dataset` retrieves our dataset's features and labels one sample at a time. While training a model, we typically want to pass samples in "minibatches", reshuffle the data at every epoch to reduce model overfitting, and use Python's `multiprocessing` to speed up data retrieval.

`DataLoader` is an iterable that abstracts this complexity for us in an easy API.

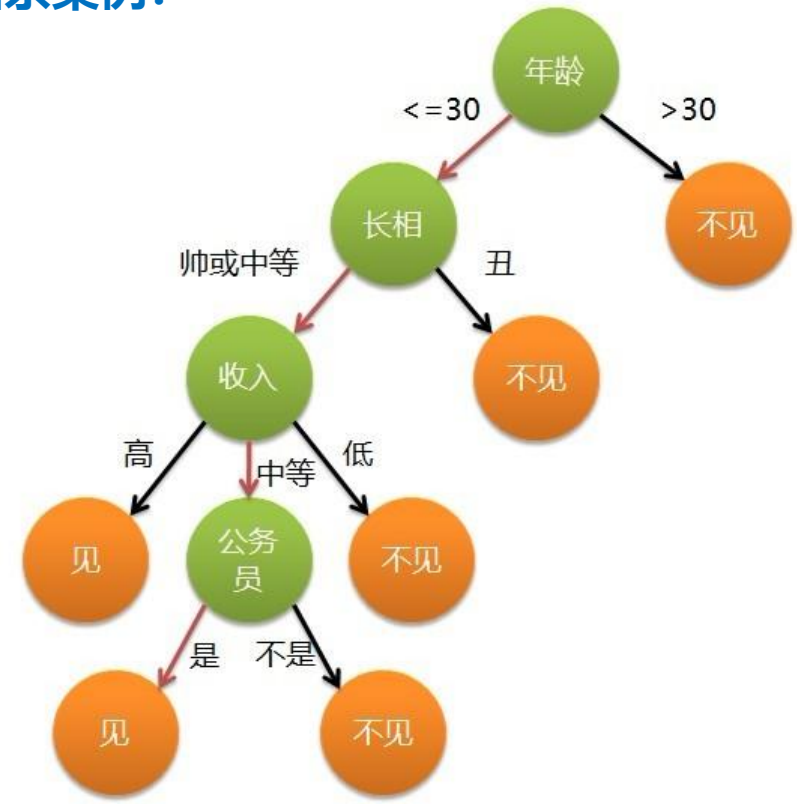
```
from torch.utils.data import DataLoader

train_dataloader = DataLoader(training_data, batch_size=64, shuffle=True)
test_dataloader = DataLoader(test_data, batch_size=64, shuffle=True)
```

决策树

Decision (if else) Tree

某相亲案例：




```
if 年龄 <=30:
    if 长相 == '帅' or 长相 == '中等':
        if 收入 == '高':
            return '见'
        elif 收入 == '低':
            return '不见'
        else:
            if 职业 == '公务员':
                return '见'
            else:
                return '不见'
    else:
        return '不见'
else:
    return '不见'
```

常见决策树类型：ID3、C4.5、CART

ID3、C4.5

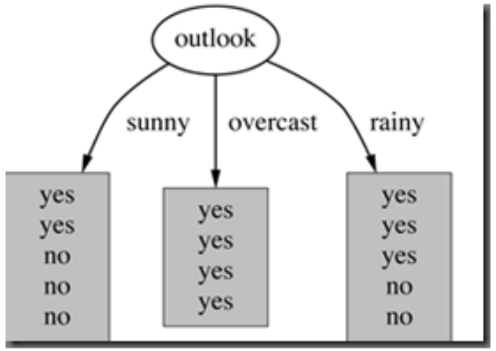
表1 天气预报数据集例子

Outlook	Temperature	Humidity	Windy	Play?
sunny	hot	high	false	no
sunny	hot	high	true	no
overcast	hot	high	false	yes
rain	mild	high	false	yes
rain	cool	normal	false	yes
rain	cool	normal	true	no
overcast	cool	normal	true	yes
sunny	mild	high	false	no
sunny	cool	normal	false	yes
rain	mild	normal	false	yes
sunny	mild	normal	true	yes
overcast	mild	high	true	yes
overcast	hot	normal	false	yes
rain	mild	high	true	no

如何建立（训练）决策树?

信息熵 $Ent(D) = -\sum_{k=1}^{K} p_k \cdot log_2 p_k$

$$Ent(D) = -\frac{9}{14} log_2 \frac{9}{14} - \frac{5}{14} log_2 \frac{5}{14} = 0.940286$$



$$H(D^{sunny}) = -\frac{2}{5} log_2 \frac{2}{5} - \frac{3}{5} log_2 \frac{3}{5} = 0.970951$$

$$H(D^{overcast}) = -\frac{4}{4} log_2 \frac{4}{4} = 0$$

$$H(D^{rain}) = -\frac{2}{5} log_2 \frac{2}{5} - \frac{3}{5} log_2 \frac{3}{5} = 0.970951$$

信息增益

$$Ent(D) = -\frac{9}{14}log_2\frac{9}{14} - \frac{5}{14}log_2\frac{5}{14} = 0.940286$$

$$\begin{aligned} Ent(D') &= \frac{5}{14}H(D^{sunny}) + \frac{4}{14}H(D^{overcast}) + \frac{5}{14}H(D^{rain}) \\ &= \frac{5}{14} \times 0.970951 + 0 + \frac{5}{14} \times 0.970951 \\ &= 0.693536 \end{aligned}$$

$$Gain(D^{outlook}) = Ent(D) - Ent(D') = 0.24675$$

计算特征 Temperature 对应的信息增益

决策树

Hot: no no yes yes
Mild: yes no yes yes yes no
Cool: yes no yes yes

$$H(D^{hot}) = -\frac{2}{4}\log_2(\frac{2}{4}) - \frac{2}{4}\log_2(\frac{2}{4}) = 1.0$$

$$H(D^{mild}) = -\frac{4}{6}\log_2(\frac{4}{6}) - \frac{2}{6}\log_2(\frac{2}{6}) = 0.9183$$

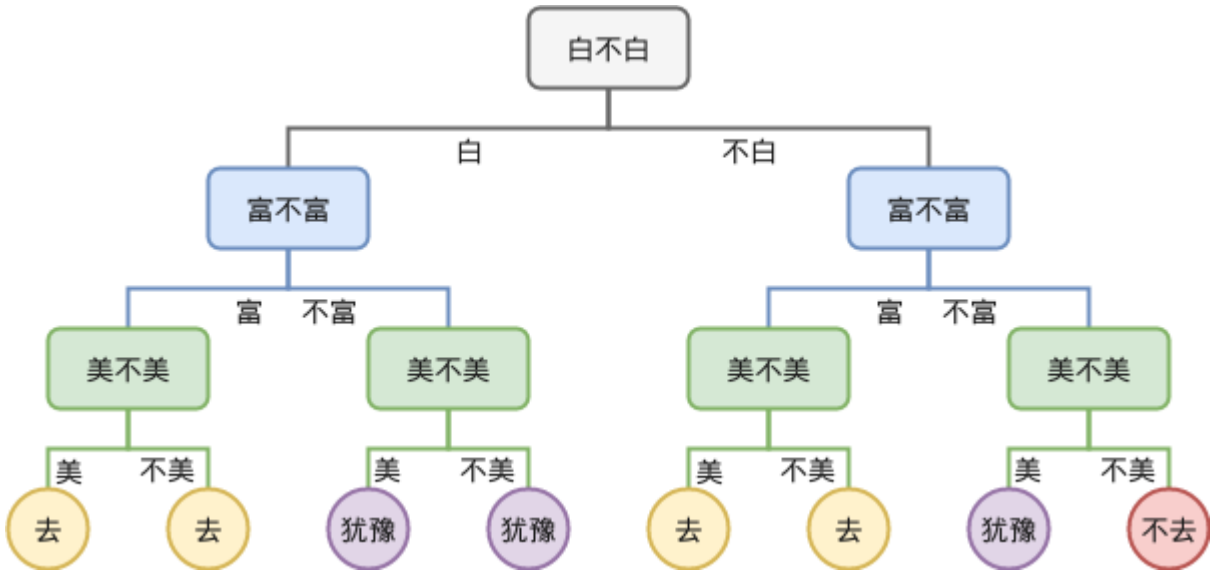
$$H(D^{cool}) = -\frac{3}{4}\log_2(\frac{3}{4}) - \frac{1}{4}\log_2(\frac{1}{4}) = 0.8112$$

$$Ent(D') = \frac{4}{14} \times 1.0 + \frac{6}{14} \times 0.9183 + \frac{4}{14} \times 0.8112 = 0.9110$$

$$Gain(D^{Temperature}) = Ent(D) - Ent(D') = 0.0293$$

决策树

CART (Classification And Regression Tree)



有房者	婚姻状况	年收入	拖欠贷款者
是	单身	125K	否
否	已婚	100K	否
否	单身	70K	否
是	已婚	120K	否
否	离异	95K	是
否	已婚	60K	否
是	离异	220K	否
否	单身	85K	是
否	已婚	75K	否
否	单身	90K	是

https://blog.csdn.net/u010089444/article/details/53241218

基尼（Gini）指数

$$Ent(D) = -\sum_{k=1}^{K} p_k \cdot log_2 p_k$$

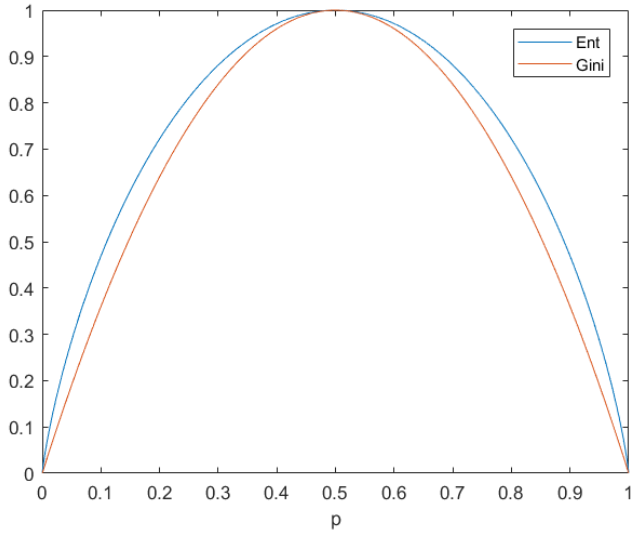
$$gini(T_i) = 1 - \sum_{j=1}^{n} p_j^2$$

二叉树

$$\begin{aligned} Ent(D) &= -\sum_{k=1}^{K} p_k \cdot \log_2 p_k \\ &= -p\log_2(p) - (1-p)\log_2(1-p) \end{aligned}$$

$$gini(T_i) = 1 - p^2 - (1-p)^2$$

```
x = 0.0001:0.0001:0.9999;
y1 = -x.*log2(x)-(1-x).*log2(1-x);
y2 = 2.*(1-x.^2-(1-x).^2);
plot(x,y1);
hold on
plot(x,y2);
xlabel('p')
legend('Ent','Gini')
hold off
```



二叉树处理连续特征

有房者	婚姻状况	年收入	拖欠贷款者
是	单身	125K	否
否	已婚	100K	否
否	单身	70K	否
是	已婚	120K	否
否	离异	95K	是
否	已婚	60K	否
是	离异	220K	否
否	单身	85K	是
否	已婚	75K	否
否	单身	90K	是

	有房	无房
否	3	4
是	0	3

$Gini(t_1)=1-(3/3)^2-(0/3)^2=0$

$Gini(t_2)=1-(4/7)^2-(3/7)^2=0.4849$

$Gini=0.3\times0+0.7\times0.4898=0.343$

	60	70	75	85	90	95	100	120	125	220
	65	72	80	87	92	97	110	122	172	
	≤ >	≤ >	≤ >	≤ >	≤ >	≤ >	≤ >	≤ >	≤ >	≤ >
是	0 3	0 3	0 3	1 2	2 1	3 0	3 0	3 0	3 0	3 0
否	1 6	2 5	3 4	3 4	3 4	3 4	4 3	5 2	6 1	
Gini	0.400	0.375	0.343	0.417	0.400	0.300	0.343	0.375	0.400	

回归问题（自学）

x	1	2	3	4	5	6	7	8	9	10
y	5.56	5.7	5.91	6.4	6.8	7.05	8.9	8.7	9	9.05

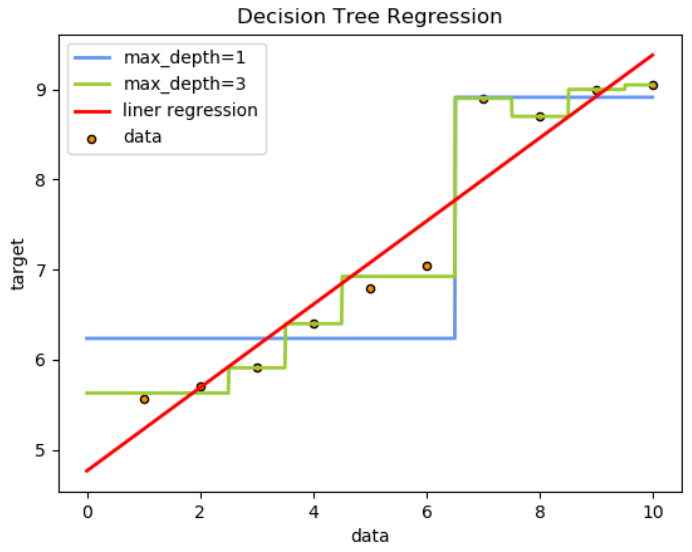
例如，取 $s=1.5$。此时 $R_1=\{1\}, R_2=\{2,3,4,5,6,7,8,9,10\}$，这两个区域的输出值分别为：
$c_1=5.56, c_2=\frac{1}{9}(5.7+5.91+6.4+6.8+7.05+8.9+8.7+9+9.05)=7.50$。得到下表：

s	1.5	2.5	3.5	4.5	5.5	6.5	7.5	8.5	9.5
c_1	5.56	5.63	5.72	5.89	6.07	6.24	6.62	6.88	7.11
c_2	7.5	7.73	7.99	8.25	8.54	8.91	8.92	9.03	9.05

把c_1, c_2的值代入到上式，如：$m(1.5)=0+15.72=15.72$。同理，可获得下表：

s	1.5	2.5	3.5	4.5	5.5	6.5	7.5	8.5	9.5
m(s)	15.72	12.07	8.36	5.78	3.91	1.93	8.01	11.73	15.74

显然取 $s=6.5$时，$m(s)$最小。因此，第一个划分变量$j=x, s=6.5$



最大深度：递归终止条件

https://blog.csdn.net/weixin_40604987/article/details/79296427