大数据导论
Introduction to Big Data

第9讲: 集成学习

叶允明

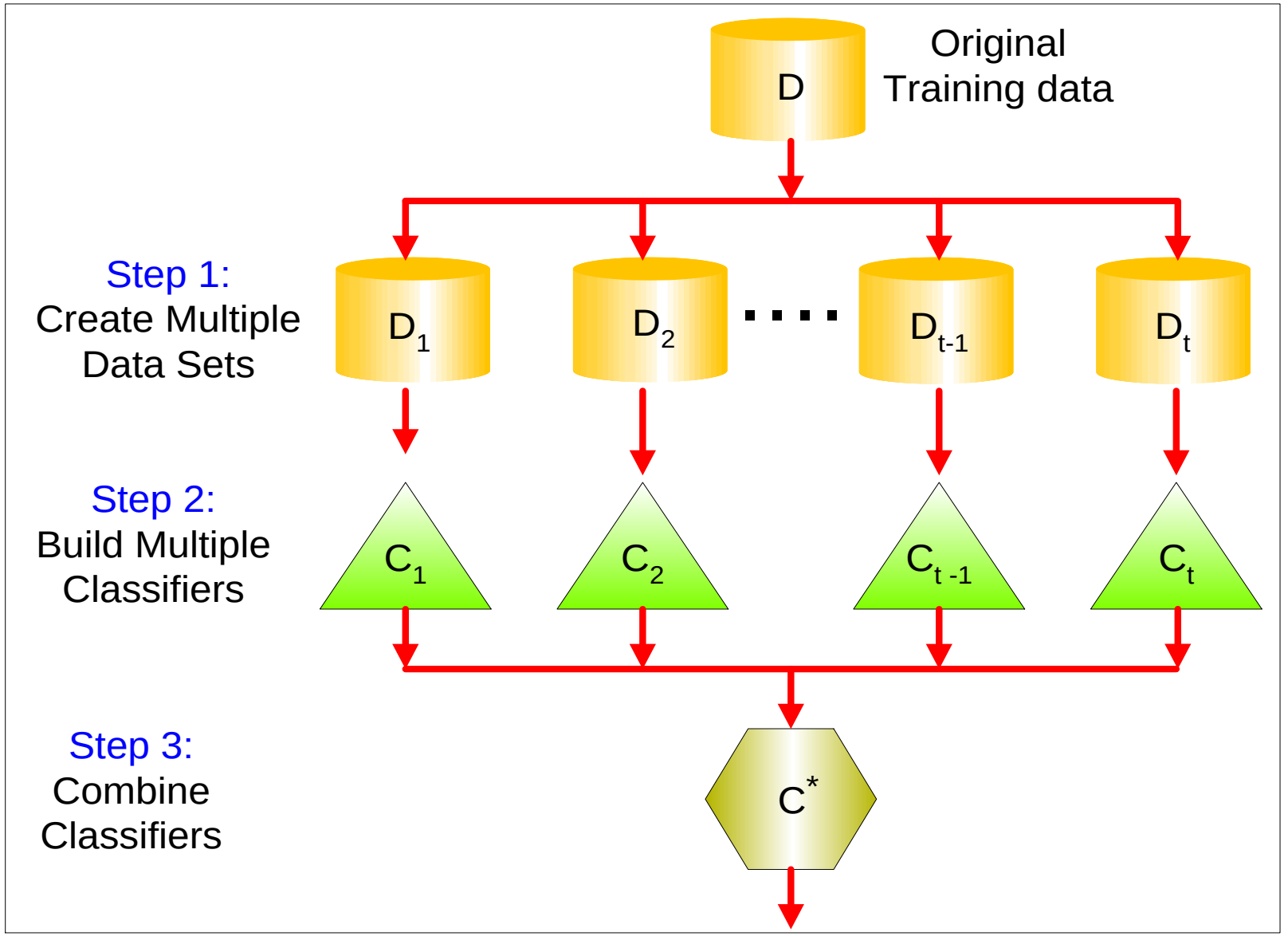
计算机科学与技术学院

哈尔滨工业大学（深圳）

基于集成学习的分类方法

- 集成学习（ensemble learning）
- 从训练数据构建分类器集合
- 综合多个分类器的预测结果来预测新实例的分类预测结果

总体思路


D
Original
Training data
Step 1:
Create Multiple
Data Sets
D1
D2
Dt-1
Dt
Step 2:
Build Multiple
Classifiers
C1
C2
Ct -1
Ct
Step 3:
Combine
Classifiers
C*

为什么会起作用?

- 假设有25个基础分类器
  - 每个分类器的错误率是$\varepsilon = 0.35$
  - 假设每个分类器之间是相互独立的
  - 那么这个集成分类器预测错误的概率是:

$$\sum_{i=13}^{25} \binom{25}{i} \varepsilon^i (1-\varepsilon)^{25-i} = 0.06$$

集成学习的关键问题

- 强度 （strength）
- 相关性（correlation）
- Tradeoff between strength and correlation

集成方法的例子

- 如何生成一个集成分类器?
  - 装袋（Bagging）
  - 堆叠（Stacking）
  - 提升（Boosting）

装袋（Bagging）

- 有放回抽样

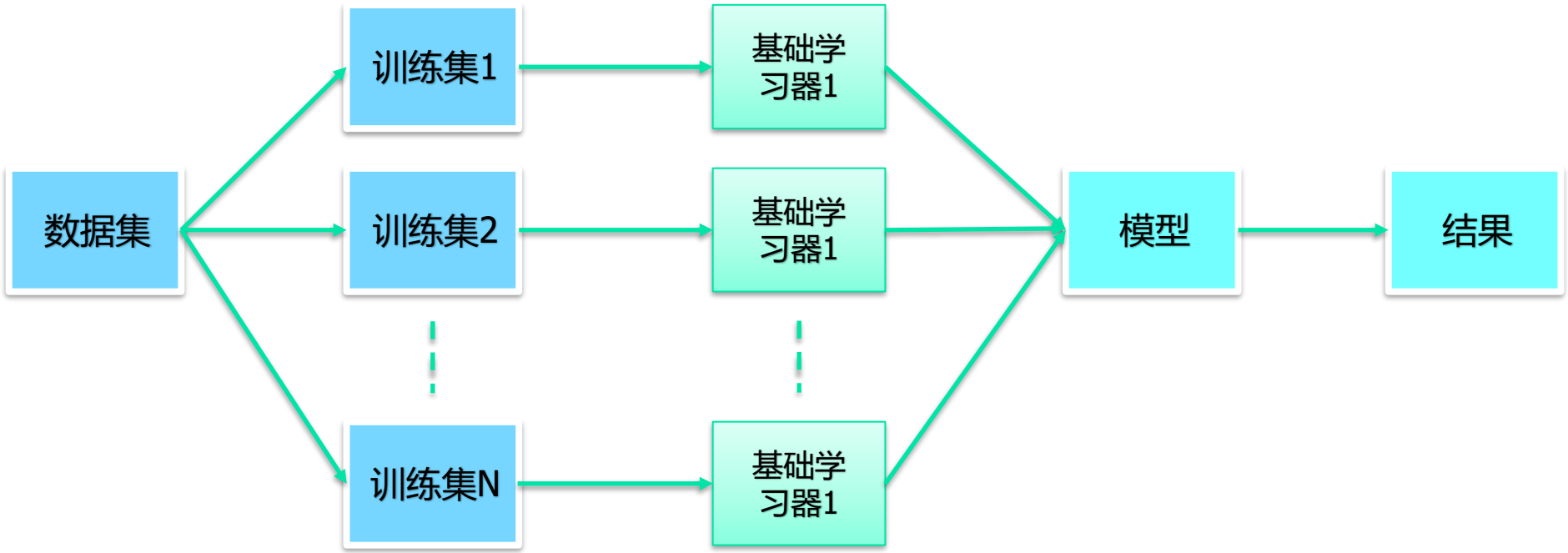
Original Data	1	2	3	4	5	6	7	8	9	10
Bagging (Round 1)	7	8	10	8	2	5	10	10	5	9
Bagging (Round 2)	1	4	9	1	2	3	2	7	3	2
Bagging (Round 3)	1	8	5	10	5	5	9	6	3	7

- 为每个自助样本建立分类器

- 每个样本被抽取的概率是相等的

堆叠法（stacking）基本原理

- 异构集成学习方法
  - 组成堆叠法的基础学习器一般互不相同



堆叠法的关键问题

- 保证不同个体学习数据的独立性
  - 对数据集进行随机划分
- 保证不同个体的多样性
  - 对不同的数据集采取不同的模型训练
- 集成策略的选择
  - 将多个模型的输出作为一个模型的输入

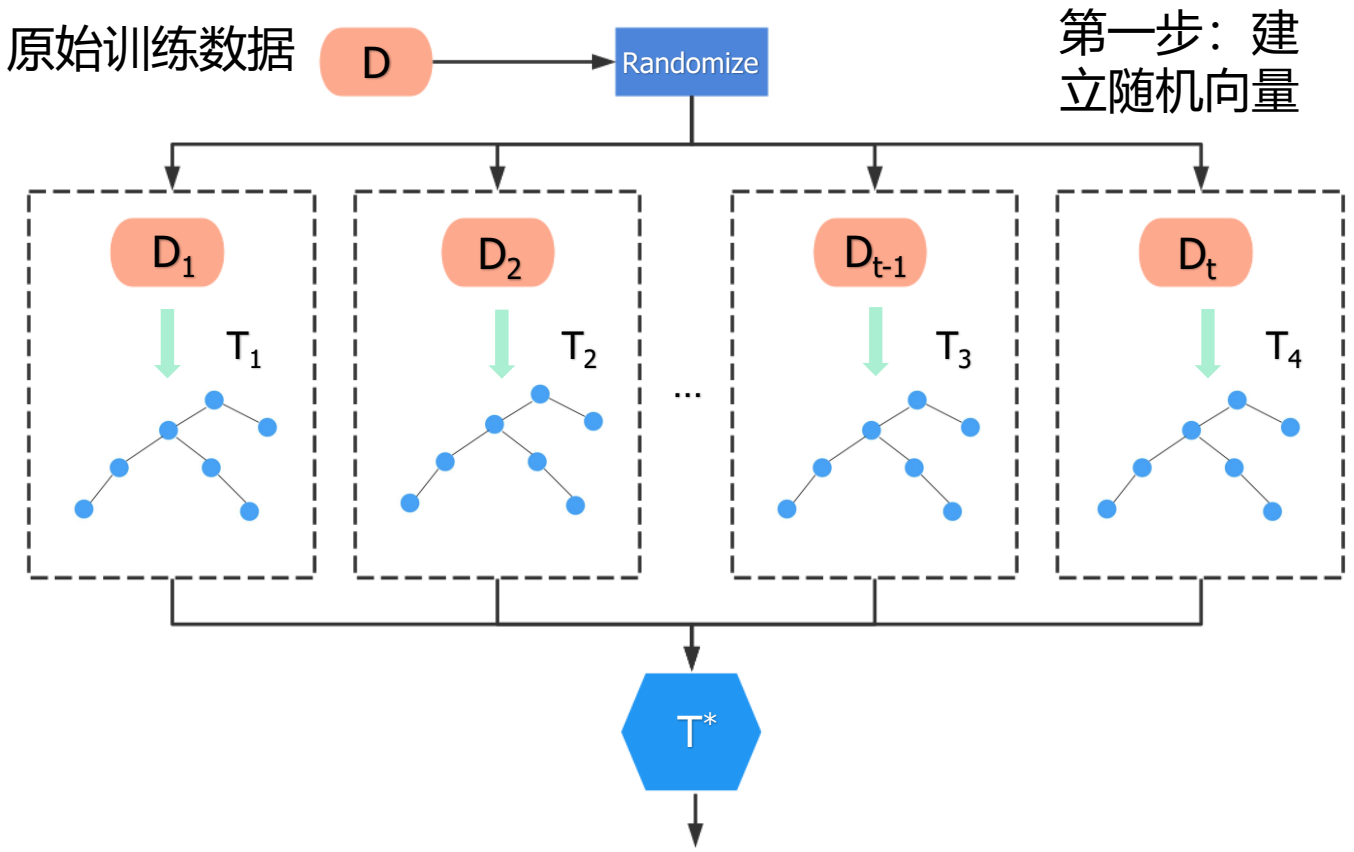
随机森林

- 专门为决策树分类器设计的集成方法
- 随机森林长出许多分类树 (名字由来)
- 未修剪的决策树集合
- 每个基础分类器分类一个“新的”向量
- 森林选择表决投票最多的分类结果

随机森林

- 随机性的两个来源介绍：“装袋”和“随机输入向量”
  - 每棵树都是使用训练数据的bootstrapping样本来生长的
  - 在每个节点上，最佳的分裂是从m_{try}个变量中选择，而不是所有变量

随机森林



第二步：使用随机向量建立多个随机数

第三步：联合所有决策树

提升（Boosting）

- 更注重之前错误分类的实例以自适应地改变训练数据分布的迭代过程
  - 最初，所有N个实例都会分配相同的权重
  - 跟装袋不同的是，这些权重会在每一轮提升结束之后发生变化
  - 最终的分类器是基于弱分类器的权重表决的

Original Data	1	2	3	4	5	6	7	8	9	10
Boosting (Round 1)	7	3	2	8	7	9	4	10	6	3
Boosting (Round 2)	5	4	9	4	2	5	1	7	4	2
Boosting (Round 3)	4	4	8	10	4	5	4	6	3	4

Adaboost (Freund and Schapire, 1997)

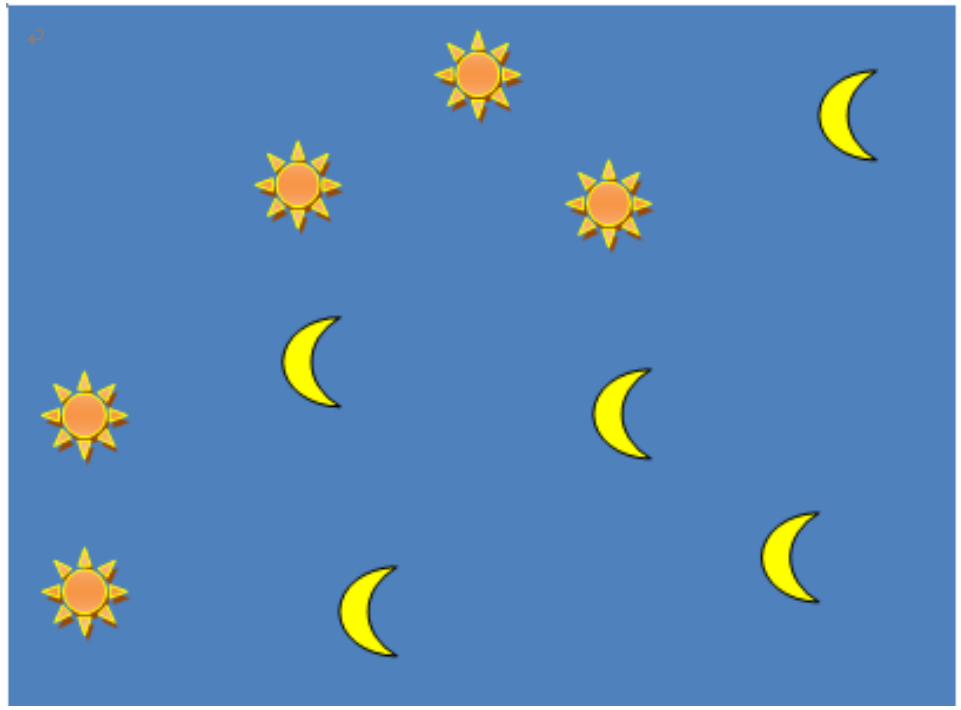
- 给定数据集D，包含d个类标记的元组 ($\mathbf{X_1}$, y_1), ..., ($\mathbf{X_d}$, y_d)
- 初始情况下，对每个训练元组赋予相等的权重1/d
- 执行k轮算法产生k个基分类器 。在第 i 轮：
  - 从D中元组进行有放回抽样，形成相同大小的训练集D_i
  - 每个元组被选中的机会由它的权重决定
  - 从训练集D_i中导出分类器M_i
  - 使用D_i 作为验证集计算M_i 的错误率
  - 如果元组被不正确地分类，那么它的权重增加；否则权重减少
- 错误率：err($\mathbf{X_j}$) 是元组$\mathbf{X_j}$的误分类误差；分类器M_i 的错误率是每个元组错误率的加权和:

$$error(M_i) = \sum_{j}^{d} w_j \times err(\mathbf{X_j})$$

- 分类器M_i的表决权重为：

$$\log \frac{1-error(M_i)}{error(M_i)}$$

Adaboost:例子



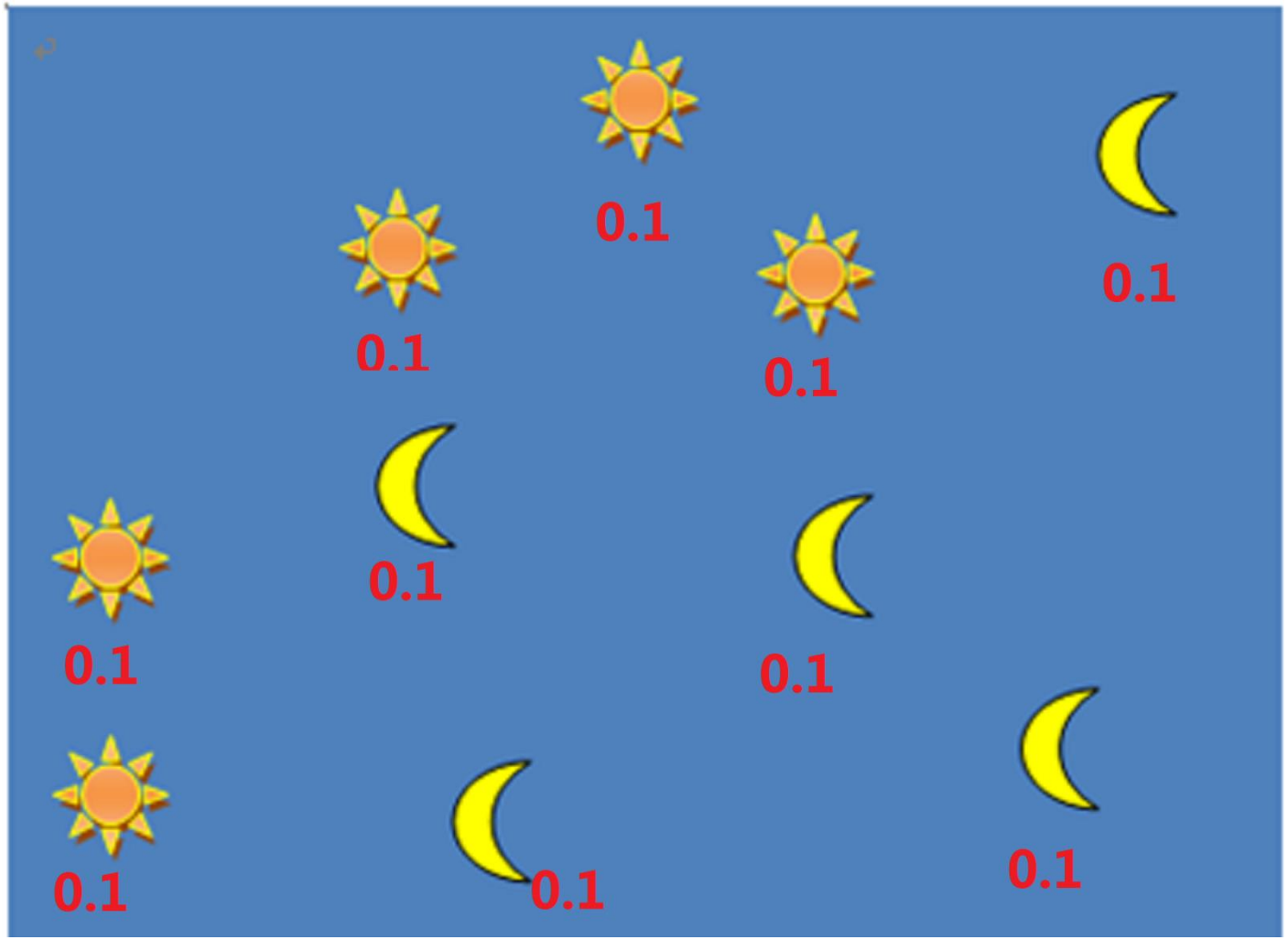
和 分别代表两个类别

在此过程中，我们使用水平或垂直线作为分类器进行分类。

Adaboost:例子

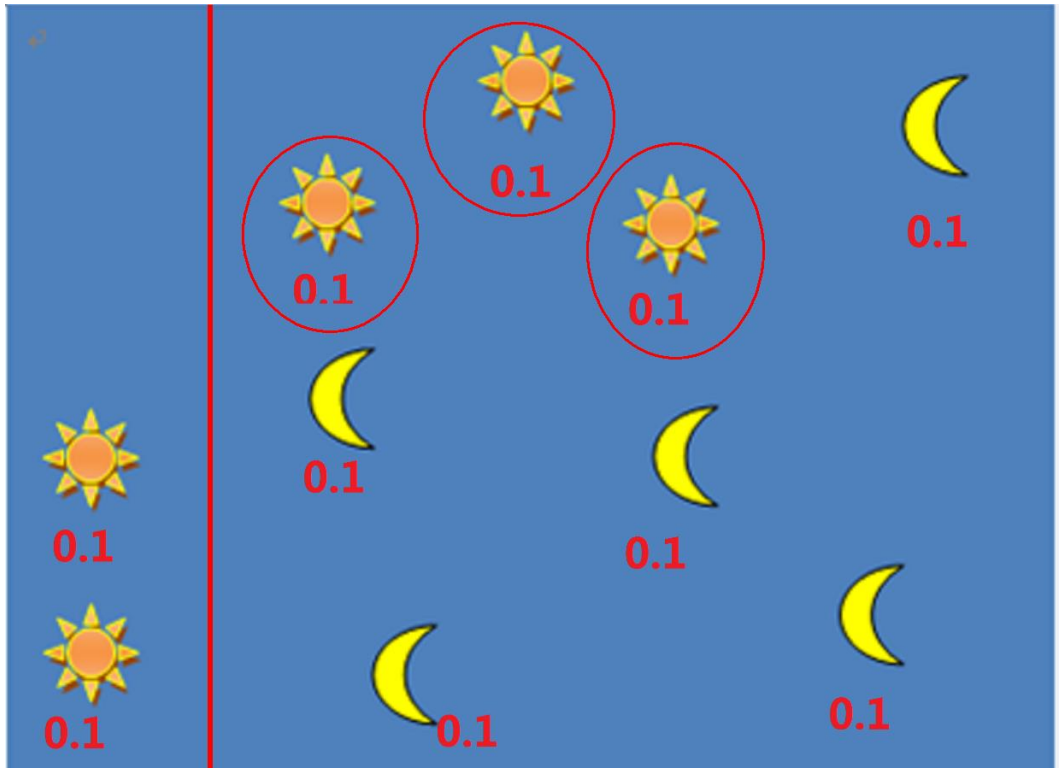
初始化元组的权重：

$$w_{init} = \frac{1}{D} = \frac{1}{10} = 0.1$$



Adaboost:例子

第一轮：M1



M1的错误率：

$$error(M_1) = \sum_{j=1}^{10} w_j \times err(X_j)$$

$$= 0.1 + 0.1 + 0.1$$

$$= 0.3$$

M1的表决权重：

$$W_1 = \log \frac{1 - error(M_1)}{error(M_1)}$$

$$= 0.477$$

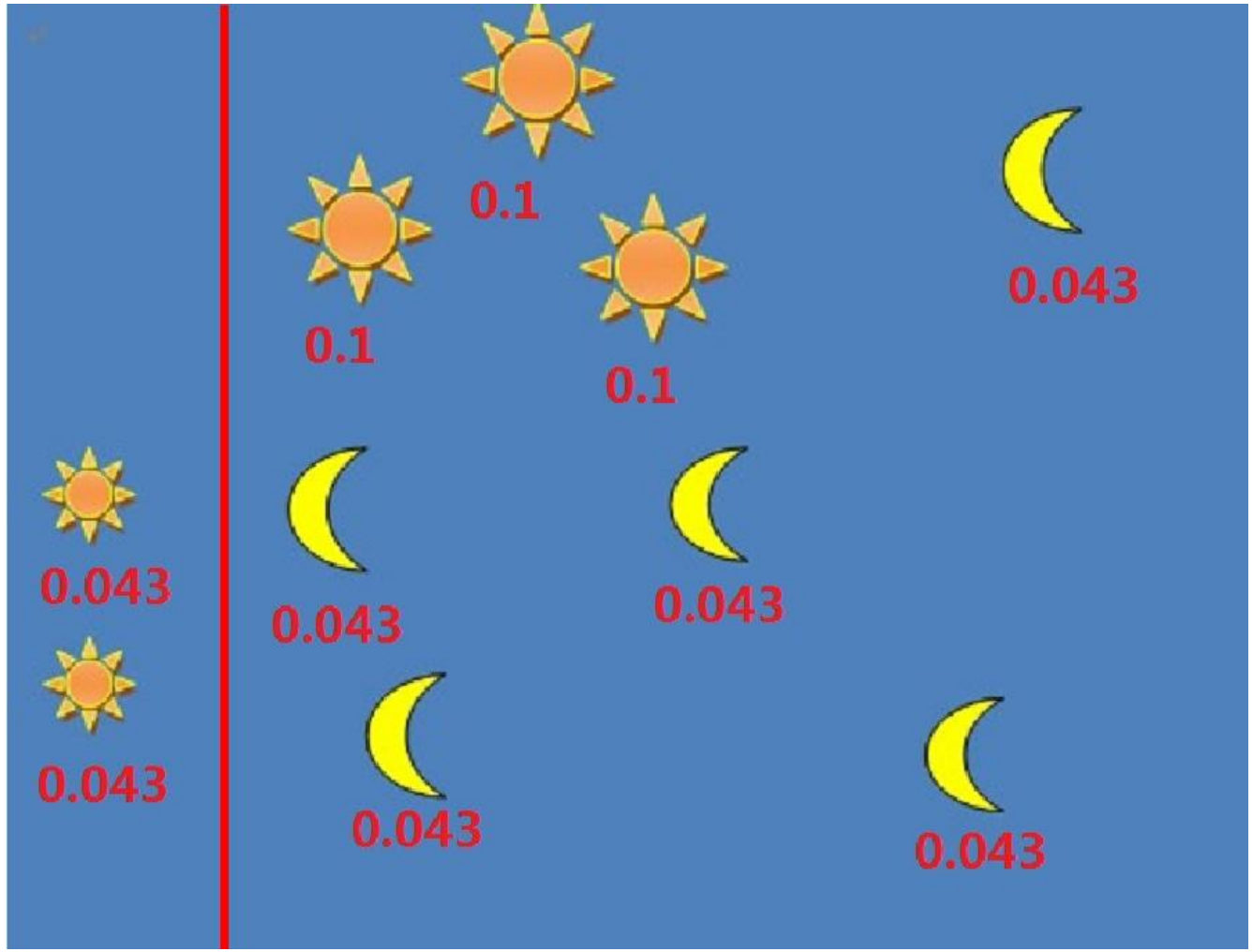
Adaboost:例子

第一轮：M1

更新被正确分类的元组的权重：

$$0.1 * \frac{error(M_1)}{1 - error(M_1)}$$

$$= 0.043$$



Adaboost:例子

归一化:

旧的权重之和:

0.1*10=1.0

新的权重之和:

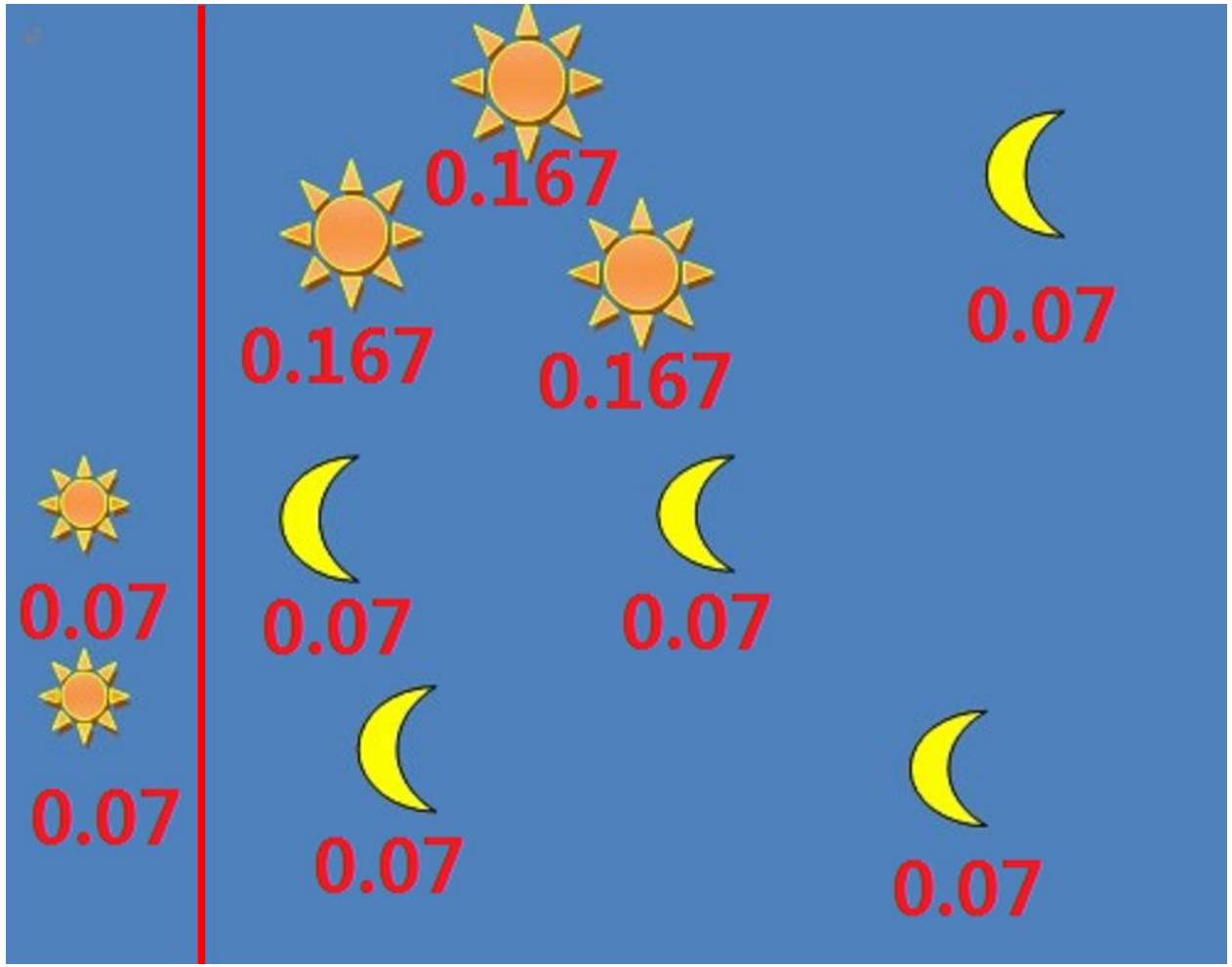
0.043*7+0.1*3=0.6

错误地分类:

$$0.1*\frac{1}{0.6}=0.167$$

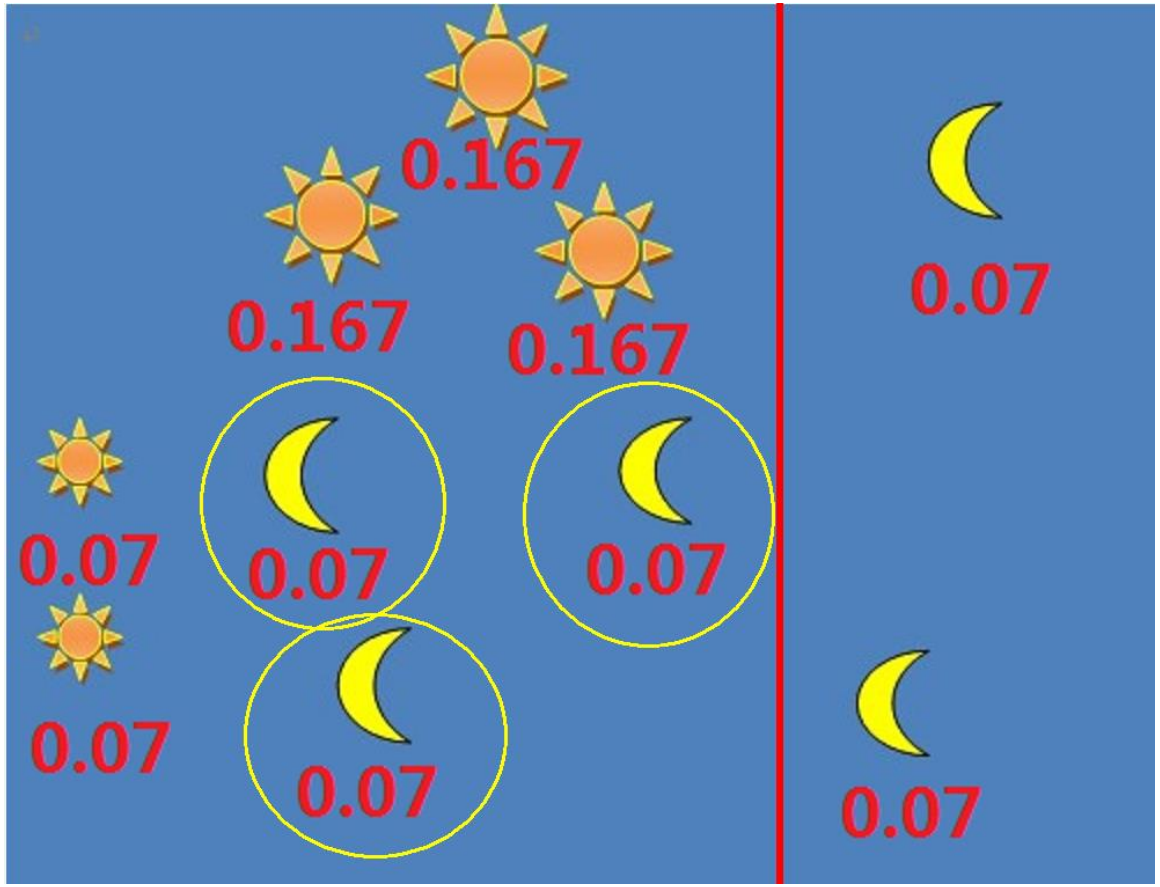
正确地分类:

$$0.043*\frac{1}{0.6}=0.07$$



Adaboost:例子

第二轮：M2



M2的错误率：

$$error(M_2) = \sum_{j=1}^{10} w_j \times err(X_j)$$
$$= 0.07 + 0.07 + 0.07$$
$$= 0.21$$

M2的表决权重：

$$W_2 = \log \frac{1 - error(M_2)}{error(M_2)}$$
$$= 0.575$$

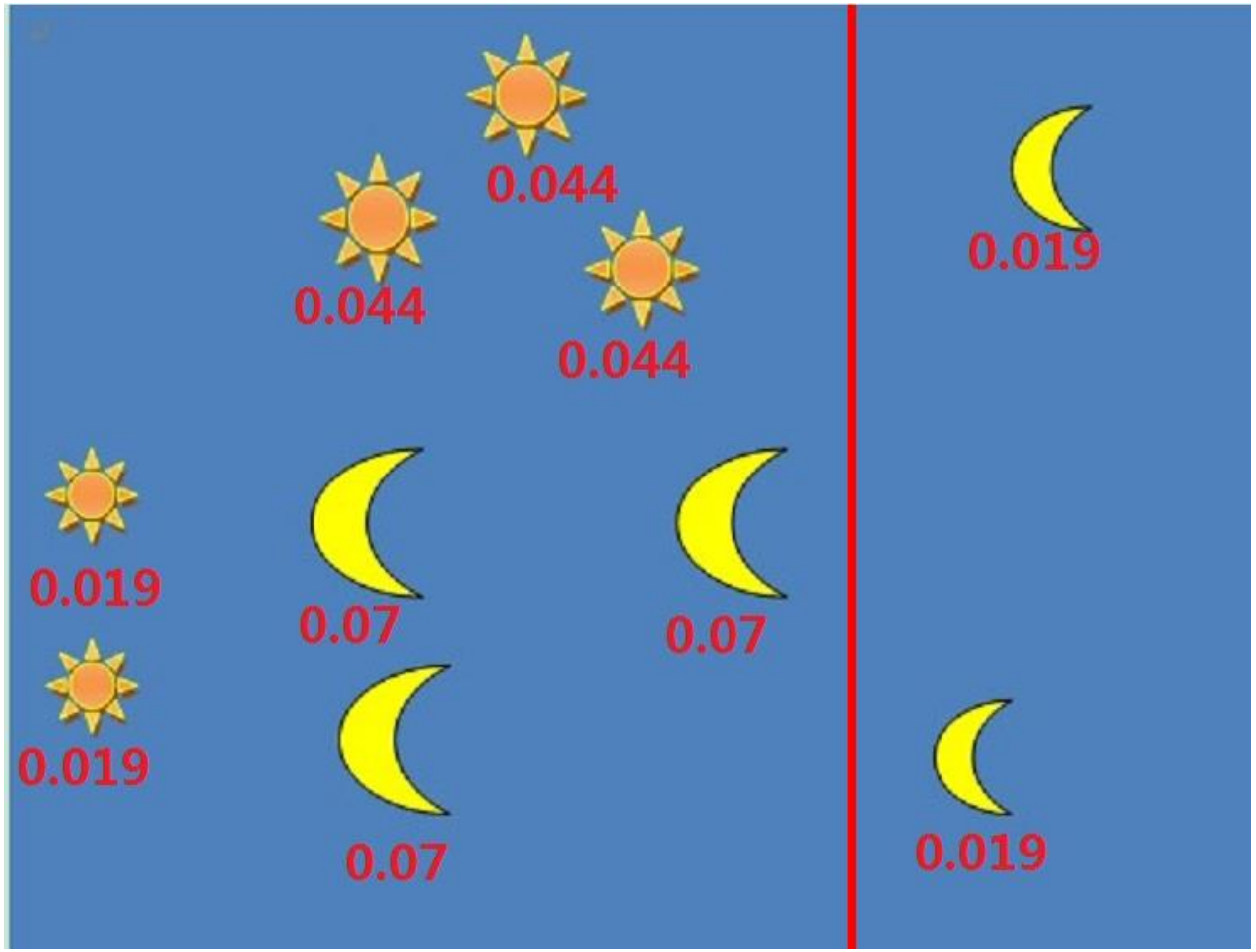
Adaboost:例子

第二轮：M2

更新被正确分类的元组的权重：

$$0.07 * \frac{error(M_2)}{1-error(M_2)} = 0.019$$

$$0.167 * \frac{error(M_2)}{1-error(M_2)} = 0.044$$



Adaboost:例子

归一化：

旧的权重之和：
0.167*3+0.07*4=0.99

新的权重之和：
0.019*4+0.044*3+0.07*3=0.418

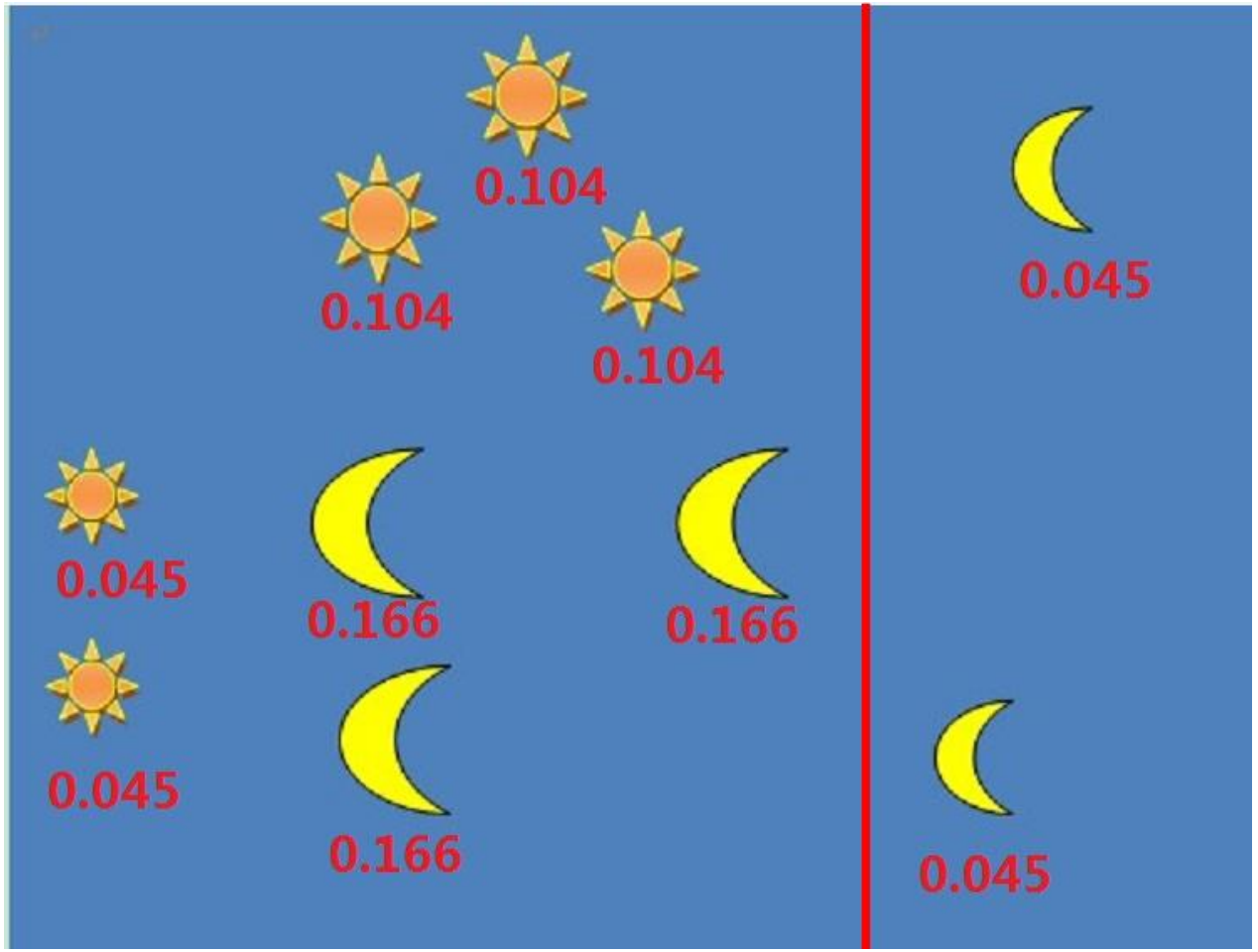
错误地分类：

$$0.07*\frac{0.99}{0.418}=0.166$$

正确地分类：

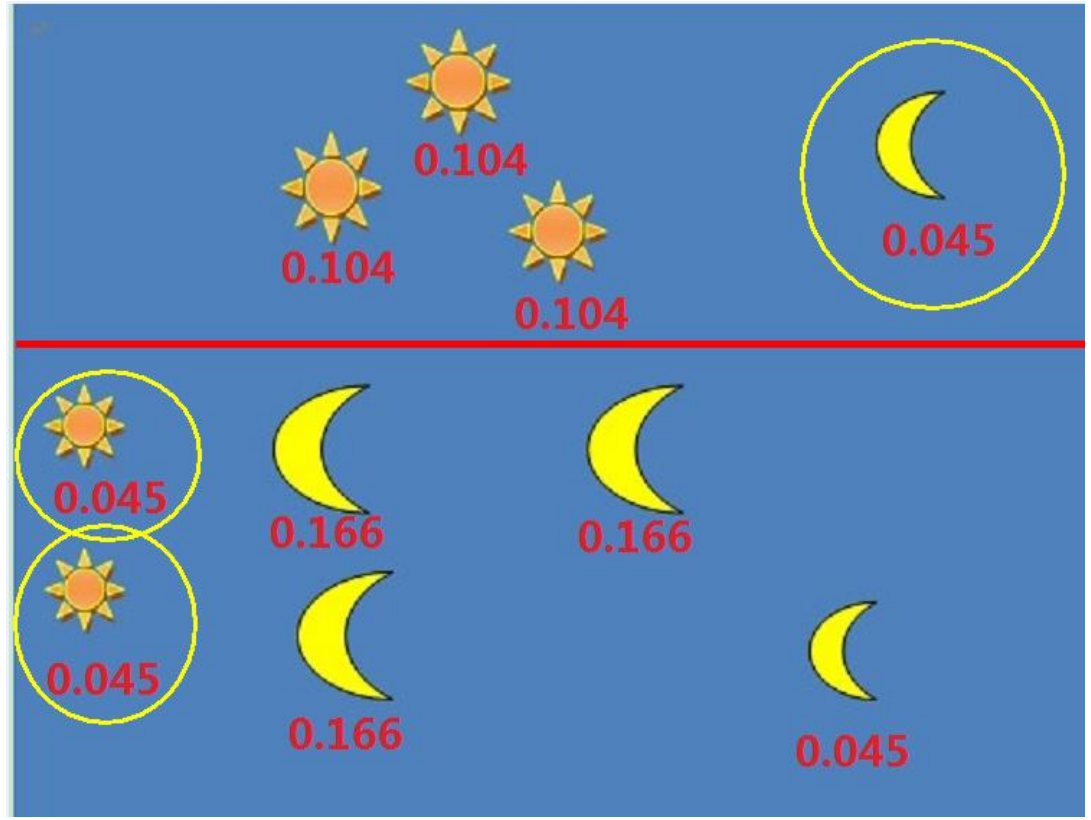
$$0.019*\frac{0.99}{0.418}=0.045$$

$$0.044*\frac{0.99}{0.418}=0.104$$



Adaboost:例子

第三轮：M3



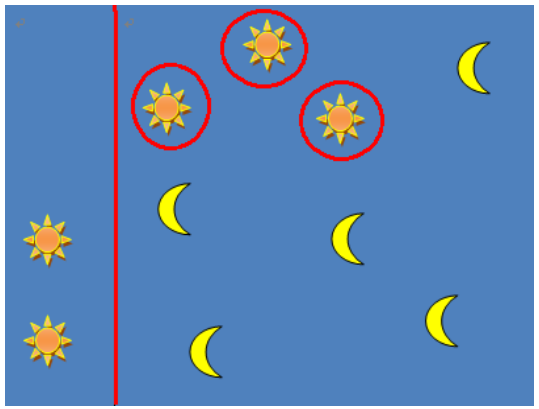
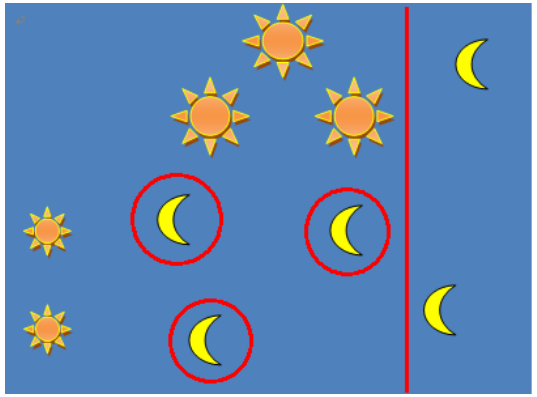
M3的错误率：

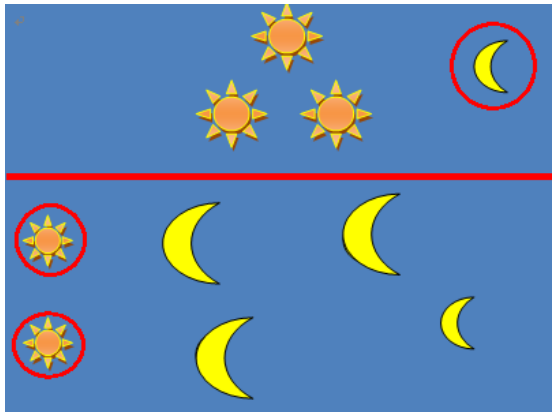
$$error(M_3) = \sum_{j=1}^{10} w_j \times err(X_j)$$
$$= 0.045 + 0.045 + 0.045$$
$$= 0.135$$

M3的表决权重：

$$W_3 = \log \frac{1 - error(M_3)}{error(M_3)} = 0.807$$

Adaboost:例子

M_{final}=sign [0.477  +0.575 

+0.807 ]

Gradient Boosting

Notations

A training set $\mathcal{D} = \{(x_i, y_i)\}_1^N$. A loss function L. The model F.

F is an additive model

$$F(x; w) = \sum_{k=0}^{K} \alpha_k h_k(x; w_k) = \sum_{k=0}^{K} f_k(x; w_k) \tag{3}$$

Define:

$$F_k = \sum_{i=0}^{k} f_i \tag{4}$$

$\{h_k(x; w_k)\}_1^K$, $\{\alpha_k\}_1^K$, $\{w_k\}_1^K$: weak learners and their weights, parameters.

Gradient Boosting

Goal

Overall Loss Function

$$\mathcal{L} = \underbrace{\sum_{i=1}^{N} L(y_i, F(x_i; w))}_{Training\ \ loss} + \underbrace{\sum_{k=1}^{K} \Omega(f_k)}_{Regularization} \tag{5}$$

Goal

$$F^* = \arg\min_F \mathcal{L} \tag{6}$$

This is a NP hard problem

Gradient Boosting

Learn Greedily

Iteration 0

Choose a f_0, usually a constant

Iteration k

$$f_k = \arg\min_{f_k} \mathcal{L}(f_k) \tag{7}$$

$$= \arg\min_{f_k} \sum_{i=1}^{N} L(y_i, F_{k-1}(x_i; w) + f_k(x_i)) + \Omega(f_k) \tag{8}$$

Finally

$$F^* = \sum_{k=1}^{K} f_k$$

Gradient Boosting Machine

J. Friedman(1999). Greedy Function Approximation: A Gradient Boosting Machine

$\Omega(f_k) = 0$ in [Friedman(1999)].

1 choose an initial f_0, let $F_0 = f_0$

2 for $k = 1, 2, \ldots, K$

2.1 $\tilde{y}_i = -\frac{\partial L(y_i, F_{k-1}(x_i))}{\partial F_{k-1}(x_i)}$, $i = 1, 2, \ldots, N$

$\{\tilde{y}_i\}_1^N$: pseudo responses, a measurement of residuals, namely $\{y_i - F_{k-1}(x_i)\}_1^N$

2.2 $w^* = \arg\min_w \sum_{i=1}^N [\tilde{y}_i - h_k(x_i; w)]^2$

Train h_k to fit $\{(x_i, \tilde{y}_i)\}_1^N$ using square error loss

2.3 $\rho^* = \arg\min_\rho \sum_{i=1}^N L(y_i, F_{k-1}(x_i) + \rho h_k(x_i; w^*))$

Perform a line search, so that $F_{k-1} + \rho^* h_k$ reduces L most

2.4 let $f_k = \rho^* h_k(x; w^*)$, $F_k = F_{k-1} + f_k$

3 output F_K

XGBoost

Tianqi Chen, Carlos Guestrin(2016). **XGBoost: A Scalable Tree Boosting System**. KDD'16.

- Objective: $\sum_{i=1}^{n} l(y_i, \hat{y}_i) + \sum_k \Omega(f_k), f_k \in \mathcal{F}$
- We can not use methods such as SGD, to find f(since they are trees, instead of just numerical vectors)
- Solution: Additive Trainging(Boosting)
  - Start from constant prediction, add a new function each time

$$
\begin{aligned}
\hat{y}_i^{(0)} &= 0 \\
\hat{y}_i^{(1)} &= f_1(x_i) = \hat{y}_i^{(0)} + f_1(x_i) \\
\hat{y}_i^{(2)} &= f_1(x_i) + f_2(x_i) = \hat{y}_i^{(1)} + f_2(x_i) \\
&\cdots \\
\hat{y}_i^{(t)} &= \sum_{k=1}^{t} f_k(x_i) = \hat{y}_i^{(t-1)} + f_t(x_i)
\end{aligned}
$$

New function

Model at training round t

Keep functions added in previous round

LightGBM

- Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, Tie-Yan Liu. "LightGBM: A Highly Efficient Gradient Boosting Decision Tree". Advances in Neural Information Processing Systems 30 (NIPS 2017), pp. 3149-3157.

致谢

- 部分图表、文字来自教材、互联网等，仅供公益性的学习参考，在此表示感谢！如有版权要求请联系：yym@hit.edu.cn，谢谢！