Gradient Boosting

- 梯度提升算法
  - 分类回归树（CART）
  - Adaboost算法概述
  - GBDT优化思想
  - GBDT解决分类与回归问题

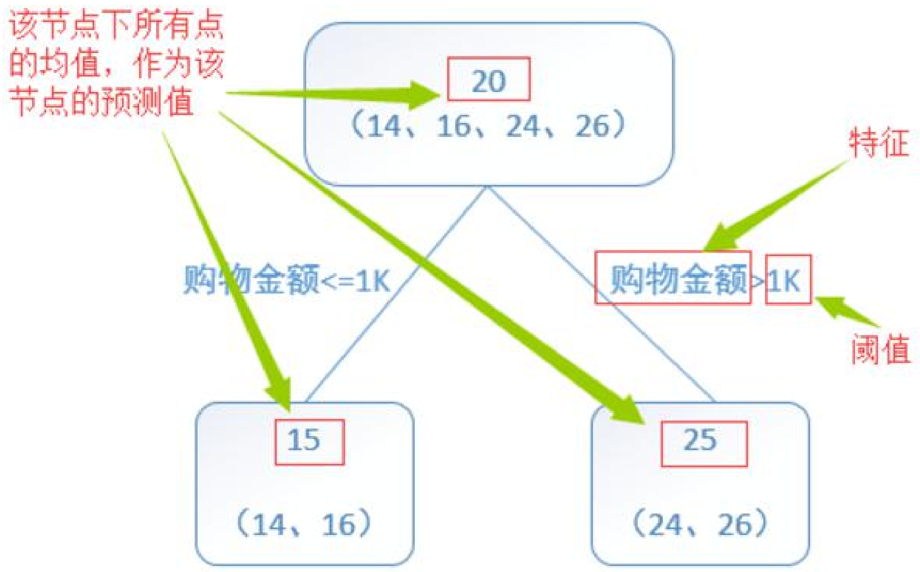
感谢 Alexey Natekin大神提供的部分素材

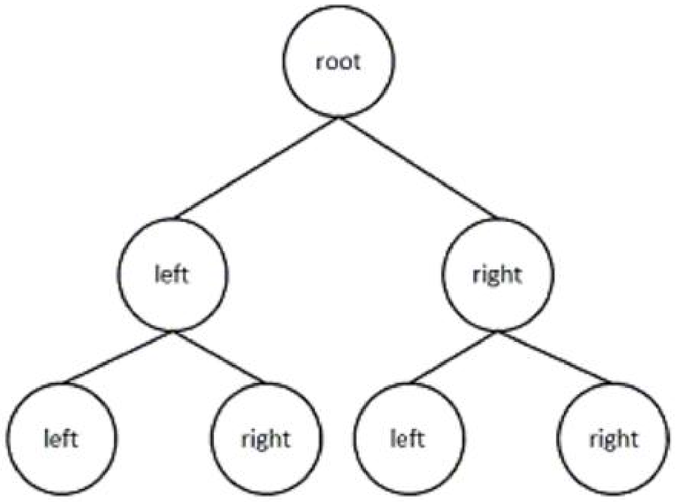
Gradient Boosting

分类回归树

- 数据集：$\left\{\left(X^{(1)}, y^{(1)}\right), \left(X^{(2)}, y^{(2)}\right), \cdots, \left(X^{(m)}, y^{(m)}\right)\right\}$

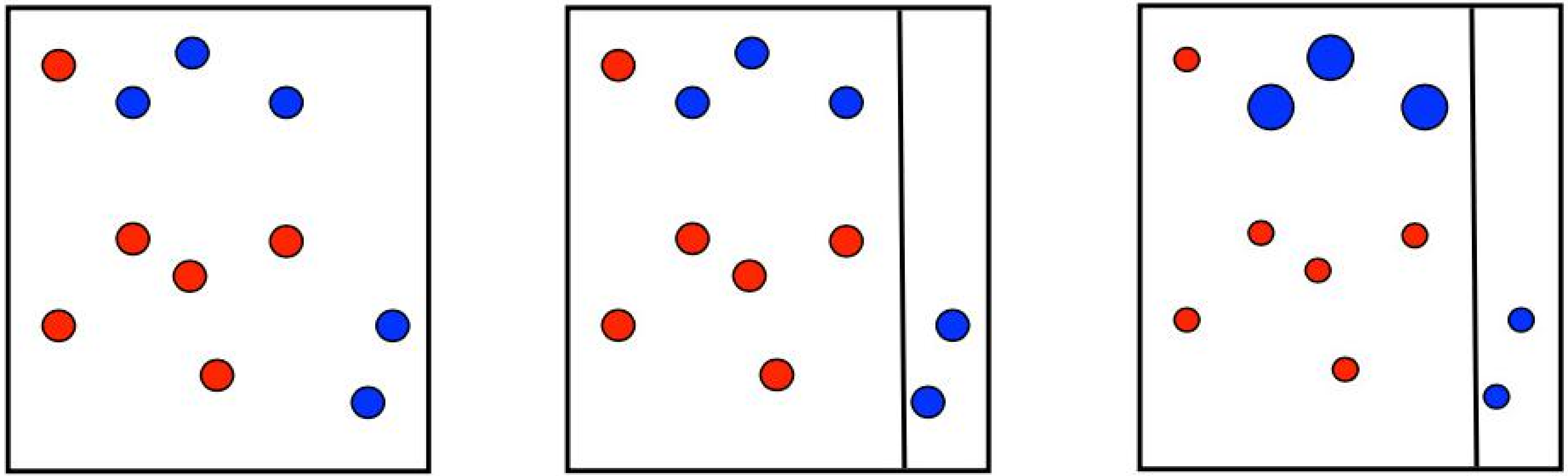
- 衡量的标准：$s^2 \cdot m = \left(y^{(1)} - \bar{y}\right)^2 + \left(y^{(2)} - \bar{y}\right)^2 + \cdots + \left(y^{(m)} - \bar{y}\right)^2$





Gradient Boosting

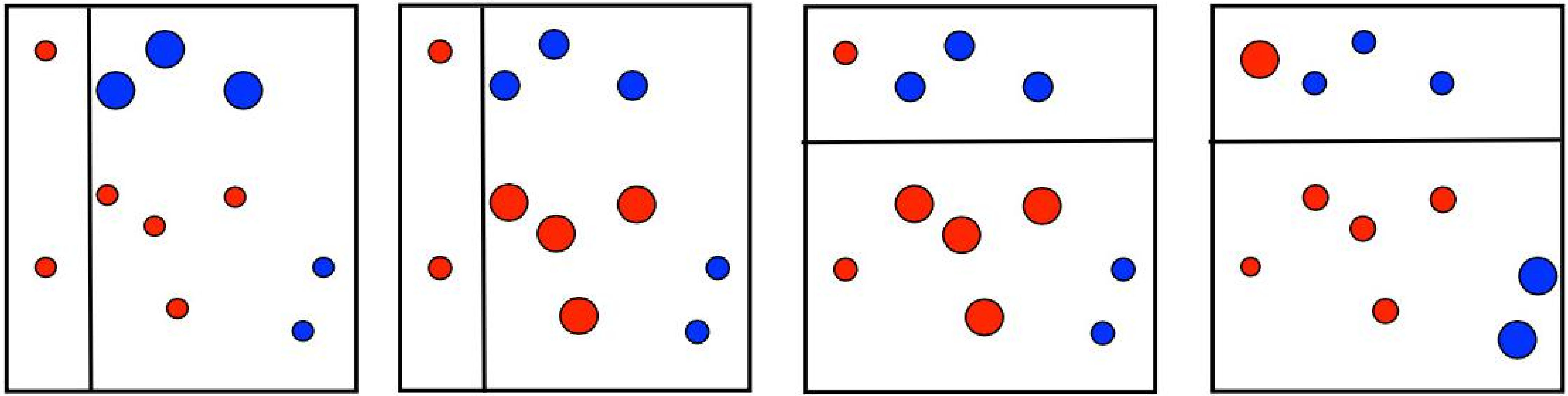
Adaboost算法概述



数据集　　　　第一次划分　　　　更新权重

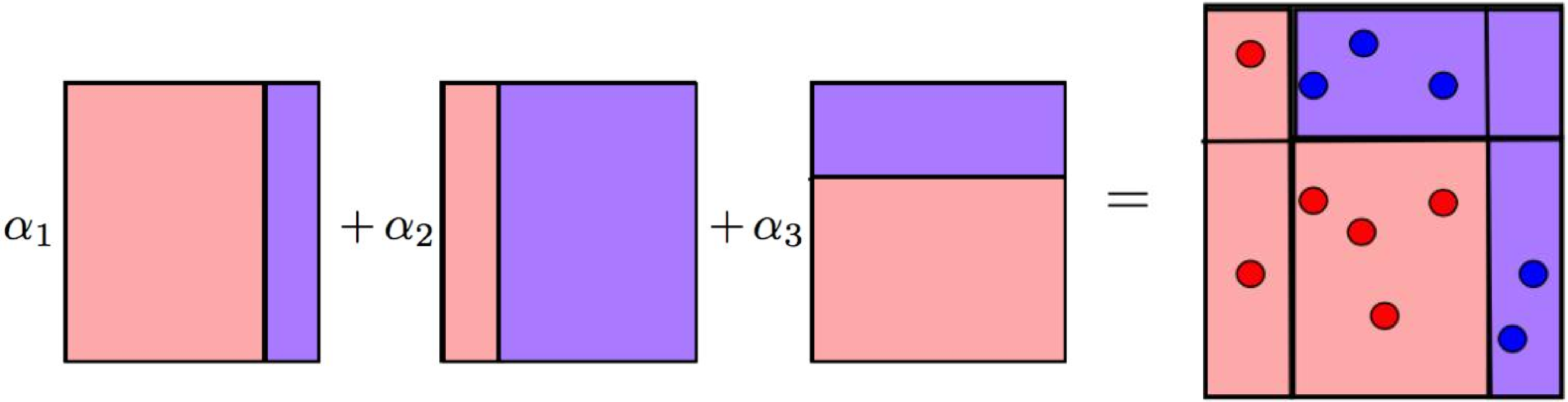
Gradient Boosting

- Adaboost迭代过程



Gradient Boosting

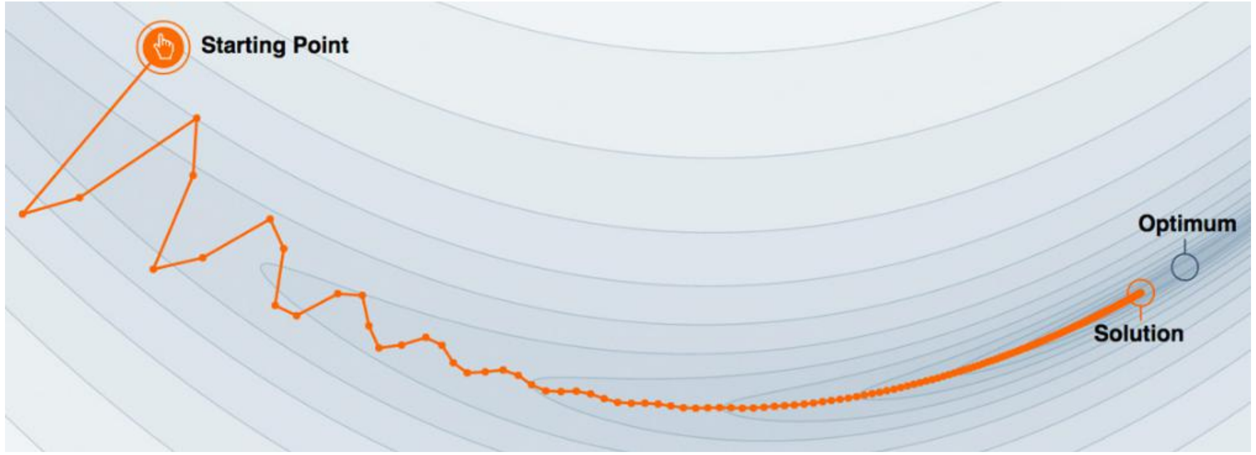
- Adaboost最终结果



Gradient Boosting

GB算法

- 优化的目标：$\arg\min\limits_{f(x)} \mathbb{E}_{x,y}[L(y, f(x))]$
- 其实目标还是去找最合适的参数：$\hat{\theta} = \arg\min\limits_{\theta} \mathbb{E}_{x,y}[L(y, f(x,\theta))]$
- 结果依旧是需要迭代得出：$\hat{\theta} = \sum_{i=1}^{M} \hat{\theta}_i$



Gradient Boosting

GB算法

损失函数的选择：

Absolute loss (more robust to outliers)

$$L(y, F) = |y - F|$$

Huber loss (more robust to outliers)

$$L(y, F) = \begin{cases} \frac{1}{2}(y - F)^2 & |y - F| \le \delta \\ \delta(|y - F| - \delta/2) & |y - F| > \delta \end{cases}$$

y_i	0.5	1.2	2	5*
$F(x_i)$	0.6	1.4	1.5	1.7
Square loss	0.005	0.02	0.125	5.445
Absolute loss	0.1	0.2	0.5	3.3
Huber loss($\delta = 0.5$)	0.005	0.02	0.125	1.525

Gradient Boosting

梯度的思想

- 找到最合适的参数： $(\rho_t, \theta_t) = \arg\min_{\rho,\theta} \mathbb{E}_{x,y}[L(y, \hat{f}(x) + \rho \cdot h(x, \theta))]$

- 残差的计算： $r_{it} = -\left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}\right]_{f(x)=\hat{f}(x)}$

- 参数迭代： $\theta_t = \arg\min_{\theta} \sum_{i=1}^{n} (r_{it} - h(x_i, \theta))^2$

- 步长的选择： $\rho_t = \arg\min_{\rho} \sum_{i=1}^{n} L(y_i, \hat{f}(x_i) + \rho \cdot h(x_i, \theta_t))$

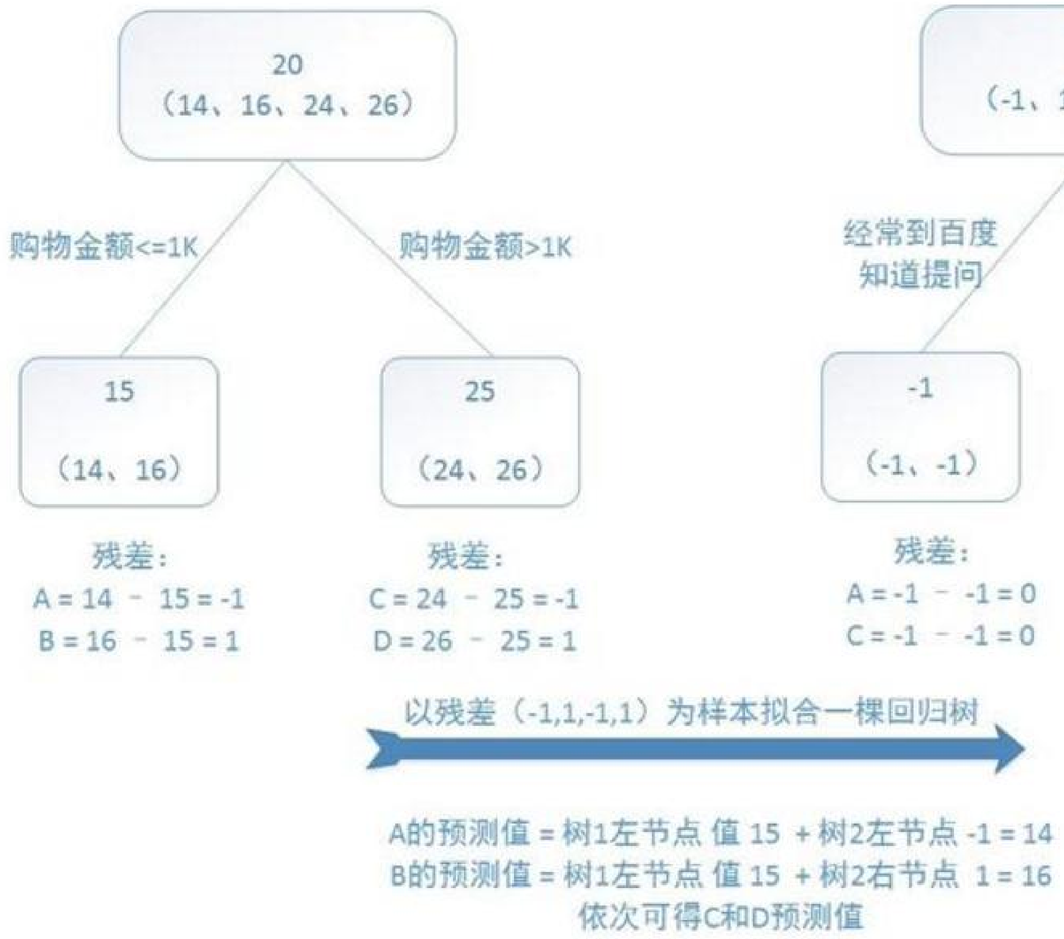
Gradient Boosting

- GBDT的工作流程 $\{(x_i, y_i)\}_{i=1,...,n}$

  - （1）初始化第一个方程：$\hat{f}(x) = \hat{f}_0, \hat{f}_0 = \gamma, \gamma \in \mathbb{R}\ \hat{f}_0 = \arg\min_{\gamma} \sum_{i=1}^{n} L(y_i, \gamma)$
  - （2）每次迭代都要计算残差：$r_{it} = -\left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}\right]_{f(x)=\hat{f}(x)}, \quad \text{for } i = 1, \ldots, n$
  - （3）以残差为目标构建回归方程：$\{(x_i, r_{it})\}_{i=1,...,n}$
  - （4）找到当前最合适的组合：$\rho_t = \arg\min_{\rho} \sum_{i=1}^{n} L(y_i, \hat{f}(x_i) + \rho \cdot h(x_i, \theta))$
  - （5）经过M次迭代后的结果：$\sum_{i=0}^{M} \hat{f}_i(x)$

Gradient Boosting

回归任务实例



Gradient Boosting

- 分类任务实例
  - 分类与回归任务均使用CART
  - 原理和多分类的逻辑回归类似
  - 对于一个3分类的任务：$p_1 = exp(f_1(x))/\sum_{k=1}^{3} exp(f_k(x))$
  - 这里我们可以训练3颗树（分别表示是不是第一类，是不是第二类。。。）

Gradient Boosting

- 分类任务实例

如果当前样本属于第二类：

$$f_{11}(x) = 0 - f_1(x)$$
$$f_{22}(x) = 1 - f2(x)$$
$$f_{33}(x) = 0 - f_3(x)$$

下一轮的输入：$(x, f_{11}(x))$ $(x, f_{22}(x))$ $(x, f_{33}(x))$

当新来一个样本的时候：$f_1(x), f_2(x), f_3(x)$

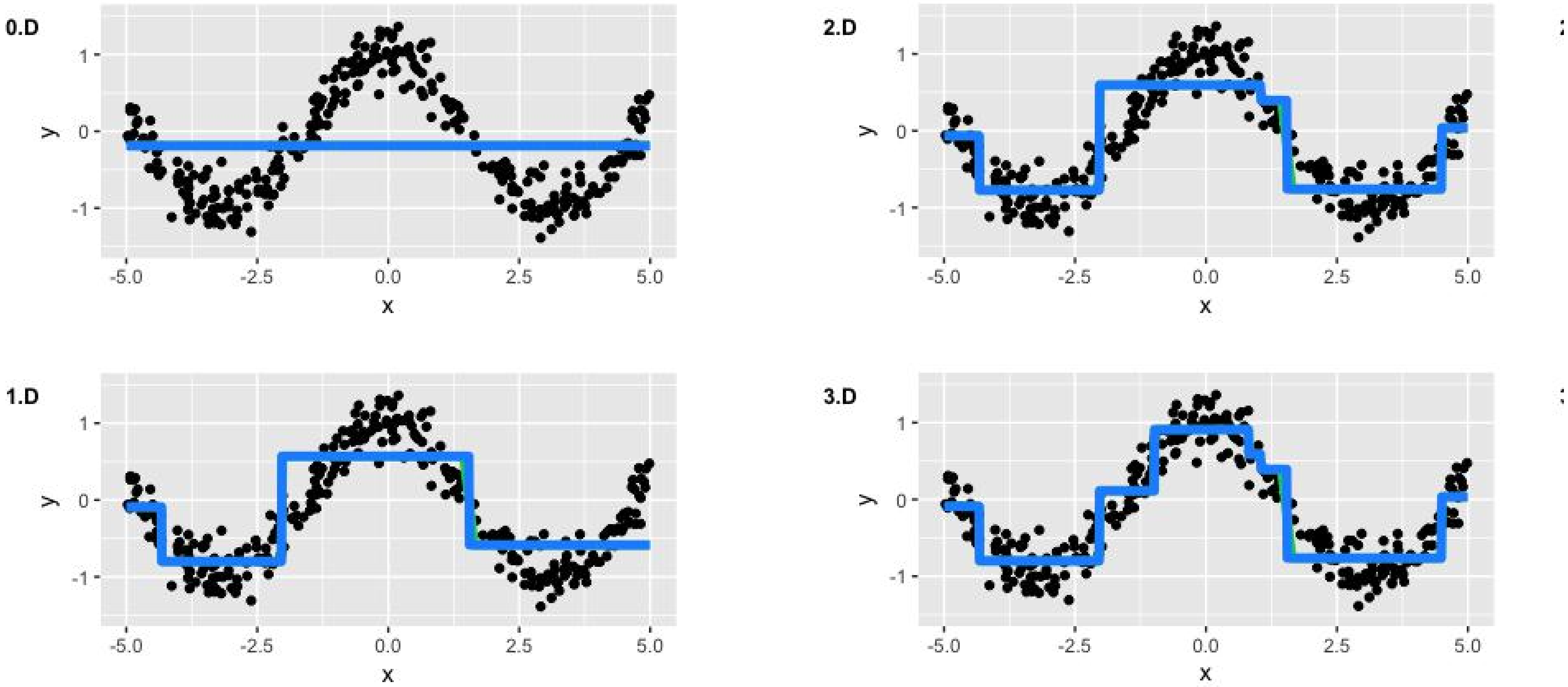
Gradient Boosting

分类任务实例

样本编号	花萼长度(cm)	花萼宽度(cm)	花瓣长度(cm)	花瓣宽度	花的种类
1	5.1	3.5	1.4	0.2	山鸢尾
2	4.9	3.0	1.4	0.2	山鸢尾
3	7.0	3.2	4.7	1.4	杂色鸢尾
4	6.4	3.2	4.5	1.5	杂色鸢尾
5	6.3	3.3	6.0	2.5	维吉尼亚鸢尾
6	5.8	2.7	5.1	1.9	维吉尼亚鸢尾

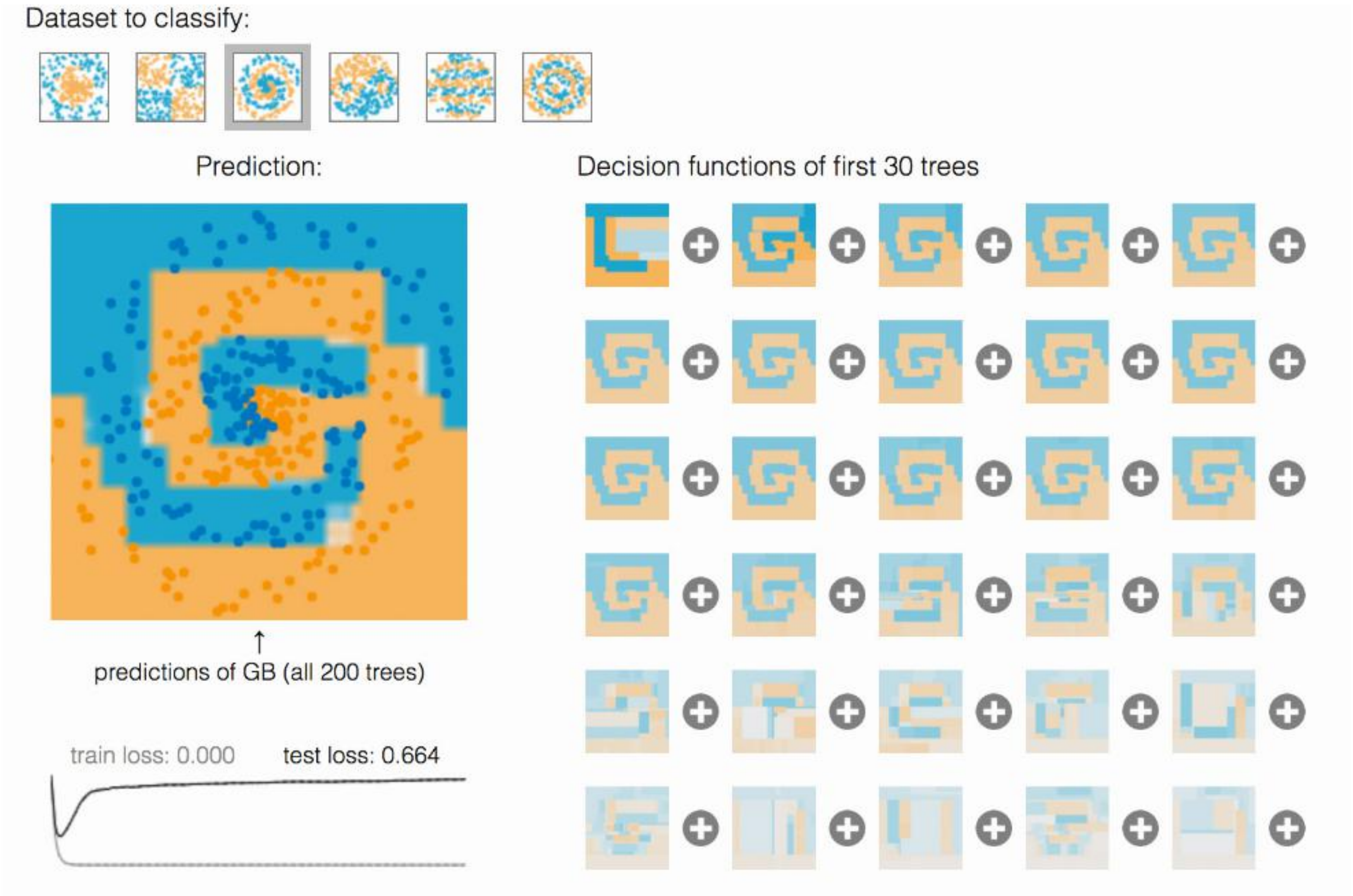
Gradient Boosting

- GBDT迭代效果



Gradient Boosting

- 可视化展示



http://arogozhnikov.github.io/2016/06/24/gradient_boosting_explained.html

http://arogozhnikov.github.io/2016/07/05/gradient_boosting_playground.html