提升算法AdaBoost

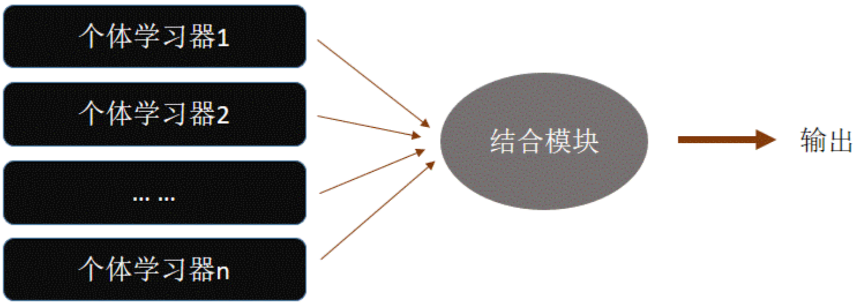
2016-11-20

引 子

强可学习	弱可学习，(基\组件)学习器
一个概念如果存在一个多项式的学习算法能够学习它，并且正确率很高，那么，这个概念是强可学习的；	一个概念如果存在一个多项式的学习算法能够学习它，并且学习的正确率仅比随机猜测略好，那么，这个概念是弱可学习的；

Schapire证明强可学习与弱可学习是等价的

集成学习



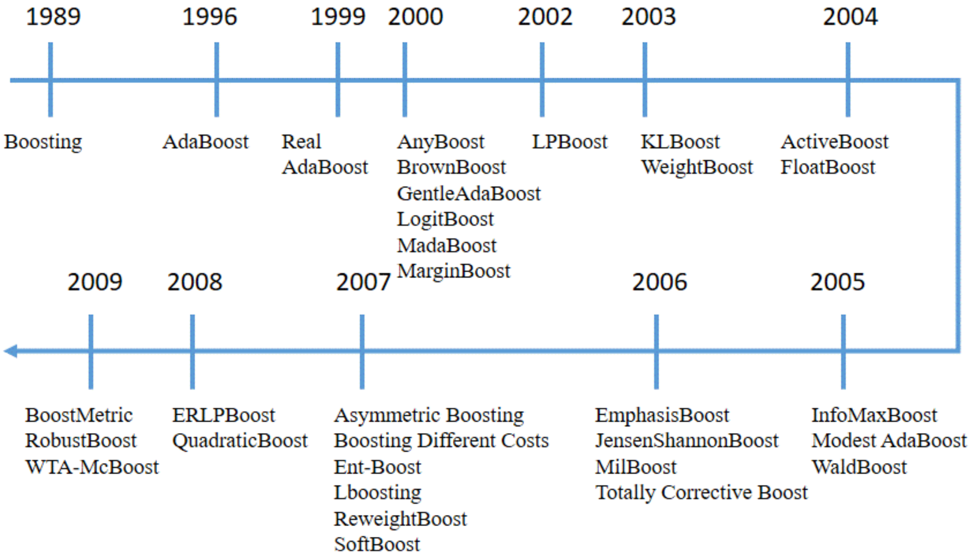
集成学习示意图（*来自：周志华机器学习*）

集成学习

- 个体学习器间存在强依赖关系、必须串行生成序列化方法；Boosting
- 个体学习器间不存在强依赖关系、可同时生成并行化方法；Bagging和Random Forest

Boosting算法发展史

Boosting算法是基于PAC学习理论（probably approximately correct）而建立的一套集成学习算法(ensemble learning)



算法过程

训练数据集

$$T = \{(x_1, y_1), (x_2, y_2), \cdots, (x_N, y_N)\}$$

其中

实例 $x_i \in \mathcal{X} \subseteq \mathbb{R}^n$　　$\mathcal{X}$ 是实例空间

标记 $y_i \in \mathcal{Y} = \{-1, +1\}$　　$\mathcal{Y}$ 是标记集合

1. 初始化训练数据的权值分布

$$D_1 = (w_{11}, \cdots, w_{1i}, \cdots, w_{1N}), \quad w_{1i} = \frac{1}{N}, \quad i = 1, 2, \cdots, N$$

2. 对 $m = 1, 2, \cdots, M$

   a) 使用具有权重分布D_m的训练数据集学习，得到基本分类器

$$G_m(x) : (X) \to \{-1, +1\}$$

   b) 计算G_m在训练数据集上的分类误差率

$$e_m = P(G_m(x_i) \neq y_i) = \sum_{i=1}^{N} w_{mi} I(G_m(x_i) \neq y_i)$$

   c) 计算G_m的系数

$$a_m = \frac{1}{2} ln \frac{1 - e_m}{e_m}$$

   d) 更新训练数据集的权值分布

$$D_{m+1} = (w_{m+1,1}, \cdots, w_{m+1,i}, \cdots, w_{m+1,N})$$

$$w_{m+1,i} = \frac{w_{mi}}{Z_m} exp(-a_m y_i G_m(x_i)), \quad i = 1, 2, \cdots, N$$

其中，Z_m是规范化因子

$$Z_m = \sum_{i=1}^{N} w_{mi} exp(-a_m y_i G_m(x_i))$$

3. 构建基本分类器的线性组合

$$f(x) = \sum_{m=1}^{M} a_m G_m(x)$$

得到最终分类器

$$G(x) = sign(f(x)) = sign\left(\sum_{m=1}^{M} a_m G_m(x)\right)$$

算法过程实例

给定下列训练样本

X	0	1	2	3	4	5	6	7	8	9
Y	1	1	1	-1	-1	-1	1	1	1	-1

1. 初始化训练数据的权值分布

$$D_1 = (w_{11}, \cdots, w_{1i}, \cdots, w_{1N}),$$
$$w_{1i} = 0.1, \quad i = 1, 2, \cdots, N$$

2. 对 $m = 1$

   a) 在权值分布为D_1训练数据上，阈值v取2.5时分类误差率最低，故基本分类器为

$$G_1(x) = \begin{cases} 1, & x < 2.5 \\ -1, & x > 2.5 \end{cases}$$

b) 计算G_1在训练数据集上的误差率

$$e_1 = P(G_1(x_i) \neq y_i) = 0.3$$

c) 计算G_1的系数

$$a_1 = \frac{1}{2} ln \frac{1-e_1}{e_1} = 0.4236$$

d) 更新训练数据集的权值分布

$$D_2 = (w_{2,1}, \cdots, w_{2,i}, \cdots, w_{2,10})$$

$$w_{2,i} = \frac{w_{1i}}{Z_1} exp(-a_1 y_i G_1(x_i)), \quad i = 1, 2, \cdots, 10$$

$$D_2 = (0.07143, 0.07143, 0.07143, 0.07143, 0.07143, 0.07143, 0.16667, 0.16667, 0.16667, 0.07143)$$

3. 构建基本分类器的线性组合

$$f_1(x) = 0.4236 G_1(x) \qquad sign(f(1))$$ 分错3个点

精度不满足

2. 对 $m = 2$

a) 在权值分布为D_2训练数据上，阈值v取8.5时分类误差率最低，故基本分类器为

$$G_2(x) = \begin{cases} 1, & x < 8.5 \\ -1, & x > 8.5 \end{cases}$$

b) 计算G_2在训练数据集上的误差率

$$e_2 = P(G_2(x_i) \neq y_i) = 0.2143$$

c) 计算G_2的系数

$$a_2 = \frac{1}{2} ln \frac{1-e_2}{e_2} = 0.6496$$

d) 更新训练数据集的权值分布

$$D_3 = (w_{3,1}, \cdots, w_{3,i}, \cdots, w_{3,10})$$

$$w_{3,i} = \frac{w_{2i}}{Z_2} exp(-a_2 y_i G_2(x_i)), \quad i = 1, 2, \cdots, 10$$

$$D_3 = (0.0455, 0.0455, 0.0455, 0.1667, 0.1667, 0.1667, 0.1060, 0.1060, 0.1060, 0.0455)$$

3. 构建基本分类器的线性组合

$$f_2(x) = 0.4236G_1(x) + 0.6496G_2(x)$$

精度不满足

$sign(f(2))$ 分错3个点

2. 对 $m = 3$

   a) 在权值分布为D_3训练数据上，阈值v取5.5时分类误差率最低，故基本分类器为

   $$G_3(x) = \begin{cases} 1, & x < 5.5 \\ -1, & x > 5.5 \end{cases}$$

   b) 计算G_3在训练数据集上的误差率

   $$e_3 = P(G_3(x_i) \neq y_i) = 0.1820$$

   c) 计算G_3的系数

   $$a_3 = \frac{1}{2} ln \frac{1 - e_3}{e_3} = 07514$$

   d) 更新训练数据集的权值分布

   $$D_4 = (w_{4,1}, \cdots, w_{4,i}, \cdots, w_{4,10})$$

$$w_{4,i} = \frac{w_{3i}}{Z_3} exp(-a_3 y_i G_3(x_i)), \quad i = 1, 2, \cdots, 10$$

$$D_4 = (0.125, 0.125, 0.125, 0.102, 0.102, 0.102, 0.065, 0.065, 0.065, 0.125)$$

3. 构建基本分类器的线性组合

$$f_3(x) = 0.4236G_1(x) + 0.6496G_2(x) + 0.7514G_3(x)$$

$sign(f(3))$ 分错0个点

精度满足

得到最终分类器

$$\begin{aligned} G(x) &= sign(f(x)) \\ &= sign\Big(0.4236G_1(x) + 0.6496G_2(x) + 0.7514G_3(x)\Big) \end{aligned}$$

AdaBoost变形

<Discrete> AdaBoost	每一个弱分类的输出结果是1或-1，计算误差率后，通过一个对数函数将0-1的误差率值映射到实数域，最后的分类器是所有映射函数的和。
Real AdaBoost	对每个特征空间进行取值划分，然后计算每个子空间上正负样本的权重，通过一个对数函数计算每一个弱分类器的输出，再选择最小的的弱分类器作为该轮迭代选出的弱分类器，最后的分类器是所有映射函数的和。
Gentle AdaBoost	相比Real AdaBoost而言，使用牛顿法来减少对离群点的权重，提高了集成的可靠性；在每次迭代时，基于最小二乘去做一个加权回归，最后所有回归函数的和作为最终的分类器。
Modest AdaBoost	在每次迭代时，在正确和不正确的分类器上使用“反向分布”策略，减少已经很好正确分类的分类器的权重，最后的分类器是所有映射函数的和。
XGBoost	全称: eXtreme Gradient Boosting; 能够自动利用CPU的多线程进行并行，同时在算法上加以改进提高了精度。

Algorithm 4 (Discrete) AdaBoost algorithm for binary classification

Input: Dataset $Z = \{z_1, z_2, \ldots, z_N\}$, with $z_i = (\mathbf{x}_i, y_i)$, where $\mathbf{x}_i \in \mathcal{X}$ and $y_i \in \{-1, +1\}$.
M, the maximum number of classifiers.
Output: A classifier $H : \mathcal{X} \to \{-1, +1\}$.

Initialize the weights $w_i^{(1)} = 1/N, i \in \{1, \ldots, N\}$, and set $m = 1$.
while $m \leq M$ **do**
  Run weak learner on Z, using weights $w_i^{(m)}$, yielding classifier $H_m : \mathcal{X} \to \{-1, +1\}$.
  Compute $\text{err}_m = \sum_{i=1}^{N} w_i^{(m)}\, h\,(-y_i\, H_m(\mathbf{x}_i))$, the weighted error of H_m.
  Compute $\alpha_m = \frac{1}{2} \log\left(\frac{1-\text{err}_m}{\text{err}_m}\right)$. {/* Weight of weak classifier. */}
  For each sample $i = 1, \ldots, N$, update the weight $v_i^{(m)} = w_i^{(m)} \exp(-\alpha_m\, y_i\, H_m(\mathbf{x}_i))$.
  Renormalize the weights: compute $S_m = \sum_{j=1}^{N} v_j$ and, for $i = 1, \ldots, N$, $w_i^{(m+1)} = v_i^{(m)} \big/ S_m$.
  Increment the iteration counter: $m \leftarrow m + 1$
end while
Final classifier: $H(\mathbf{x}) = \text{sign}\left(\sum_{j=1}^{M} \alpha_j\, H_j(\mathbf{x})\right)$.

Algorithm 5 Real AdaBoost

Input: Dataset $Z = \{z_1, z_2, \dots, z_N\}$, with $z_i = (\mathbf{x_i}, y_i)$, where $\mathbf{x}_i \in \mathscr{X}$ and $y_i \in \{-1, +1\}$.
M, the maximum number of classifiers.
Output: A classifier $H : \mathscr{X} \rightarrow \{-1, +1\}$.

Initialize the weights $w_i = 1/N, i \in \{1, \dots, N\}$.
for $m = 1$ to M **do**
  Fit the class probability estimate $p_m(\mathbf{x}) = \hat{P}_w(y = 1|\mathbf{x})$, using w_i.
  Set $\mathrm{H}_m = \frac{1}{2} \log\left((1 - p_m(\mathbf{x}))p_m(\mathbf{x})\right) \in \mathscr{R}$.
  Update the weights: $w_i \leftarrow w_i \exp(-y_i H_m(\mathbf{x_i}))$
  Renormalize to weights.
end for
Final classifier: $H(\mathbf{x}) = \operatorname{sign}\left(\sum_{j=1}^{M} \alpha_j \; H_j(\mathbf{x})\right)$.

Algorithm 7 Gentle AdaBoost

Input: Dataset $Z = \{z_1, z_2, \dots, z_N\}$, with $z_i = (\mathbf{x_i}, y_i)$, where $\mathbf{x}_i \in \mathscr{X}$ and $y_i \in \{-1, +1\}$.
M, the maximum number of classifiers.
Output: A classifier $H : \mathscr{X} \rightarrow \{-1, +1\}$.

Initialize the weights $w_i = 1/N, i \in \{1, \dots, N\}$.
for $m = 1$ to M **do**
  Train $H_m(\mathbf{x})$ by weighted least-squares of y_i to $\mathbf{x_i}$, with weights w_i.
  Update $H(\mathbf{x}) \leftarrow H(\mathbf{x}) + H_m(\mathbf{x})$.
  Update $w_i \leftarrow w_i \exp(-y_i H_m(\mathbf{x_i}))$ and renormalize to $\sum_i w_i = 1$.
end for
Final classifier: $H(\mathbf{x}) = \operatorname{sign}\left(\sum_{j=1}^{M} \alpha_j \; H_j(\mathbf{x})\right)$.

Algorithm 8 Modest AdaBoost

Input: Dataset $Z = \{z_1, z_2, \dots, z_N\}$, with $z_i = (\mathbf{x_i}, y_i)$, where $\mathbf{x}_i \in \mathscr{X}$ and $y_i \in \{-1, +1\}$.
M, the maximum number of classifiers.
Output: A classifier $H : \mathscr{X} \rightarrow \{-1, +1\}$.

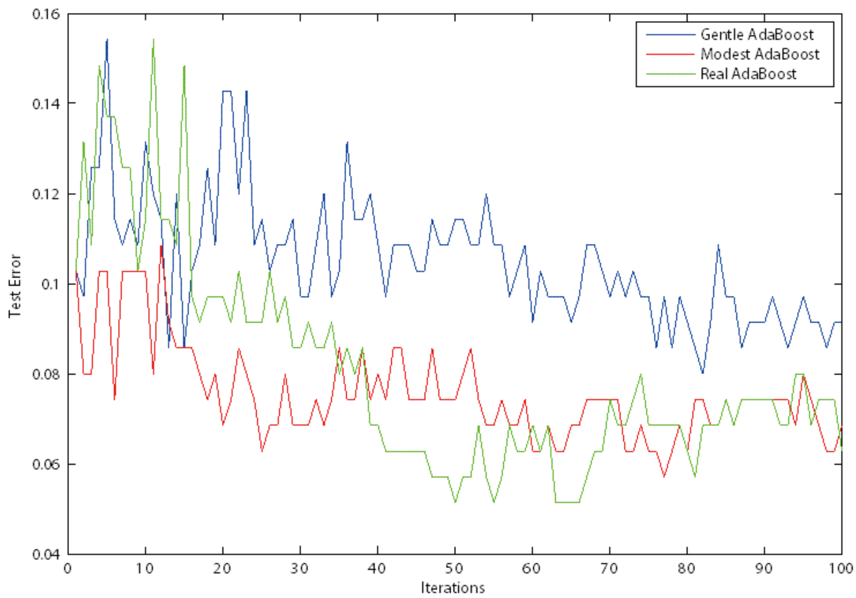
Initialize the weights $w_i = 1/N, i \in \{1, \dots, N\}$.
for $m = 1$ to M and while $H_m \neq 0$ **do**
  Train $H_m(\mathbf{x})$ by weighted least-squares of y_i to $\mathbf{x_i}$, with weights w_i.
  Compute "inverted" distribution $\overline{w}_i = (1 - w_i)$ and renormalize to $\sum_i \overline{w}_i = 1$.
  Compute $P_m^{+1} = P_w(y = +1, H_m(\mathbf{x}))$, $\overline{P_m^{+1}} = P_{\overline{w}}(y = +1, H_m(\mathbf{x}))$.
  Compute $P_m^{-1} = P_w(y = -1, H_m(\mathbf{x}))$, $\overline{P_m^{-1}} = P_{\overline{w}}(y = -1, H_m(\mathbf{x}))$,
  Set $H_m(\mathbf{x}) = \left(P_m^{+1}\left(1 - P_m^{+1}\right) - P_m^{-1}\left(1 - P_m^{-1}\right)\right)$
  Update $w_i \leftarrow w_i \exp(-y_i H_m(\mathbf{x_i}))$ and renormalize to $\sum_i w_i = 1$.
end for
Final classifier: $H(\mathbf{x}) = \operatorname{sign}\left(\sum_{j=1}^{M} \alpha_j \; H_j(\mathbf{x})\right)$.

数据集

Dataset	P	N	*Type of data / Classification problem*
Heart	45	267	SPECTF Heart Data Set
Pima.te	8	332	Diabetes in Pima Indian Women
Haberman	4	306	Haberman's Survival Data Set
Mammographic_masses	6	829	Mammographic Mass Data Set
Ionosphere	34	351	Radar data–signals returned from the ionosphere

数据来源UCI Machine Learning Repository

实验结果



数据集为UCI中的Ionosphere

实验结果

Dataset	*Real AdaBoost*	*Modest AdaBoost*	*Gentle AdaBoost*	*SVM*	XGBoost	KNN
Heart	0.20790	0.22172	0.18346	0.20608	0.203704	0.25073
Pima.te	0.28005	0.22882	0.26908	0.28910	0.164179	0.25938
Haberman	0.34088	0.27123	0.37649	0.28139	0.258065	0.29746
Mammographic_masses	0.19701	0.16042	0.20624	0.19419	0.180723	0.2461
Ionosphere	0.06690	0.07229	0.08747	0.11099	0.014085	0.14258

Adaboost迭代次数均为200；

采用5折交叉验证，最后误差取5个误差值的均值。

总 结

特点：

I. 每次迭代改变的是样本的分布，而不是重复采样（reweight)；
II. 样本分布的改变取决于样本是否被正确分类，总是分类正确的样本权值低， 总是分类错误的样本权值高（通常是边界附近的样本）；
III. 最终的结果是弱分类器的加权组合，权值表示该弱分类器的性能。

优点：

I. AdaBoost是一种有很高精度的分类器
II. 可以使用各种方法构建子分类器，AdaBoost算法提供的是框架
III. 当使用简单分类器时，计算出的结果是可以理解的。而且弱分类器构造极其简单
IV. 不用担心overfitting !

附 注

工具参考：
GML_AdaBoost_Matlab_Toolbox_0.3
http://graphics.cs.msu.ru/en/science/research/machinelearning/adaboosttoolbox
XGBoost Python Package
http://xgboost.readthedocs.io/en/latest/python/index.html

实验环境：
Win10 + Matlab R2010a + Python 2.7.11

参考：

1. Zhang, C., Zhang, C., & HC/Technik/Sonstiges. (2012). *Ensemble Machine Learning*. Springer US. *Section 2: Boosting Algorithms: A Review of Methods, Theory, and Applications*
2. 李航. 统计学习方法，第8章，北京：清华大学出版社， 2012.
3. 周志华.机器学习, 第8章，北京：清华大学出版社，2016.