Adaptive Boosting

一、Motivation of Boosting

我们启发式的算法基于小学老师教同学辨认苹果的这一个事实：

1. Tony: Red!

   Teacher: Almost Right! Let's look at the rest! (Then enlarge the *MISTAKES*!)

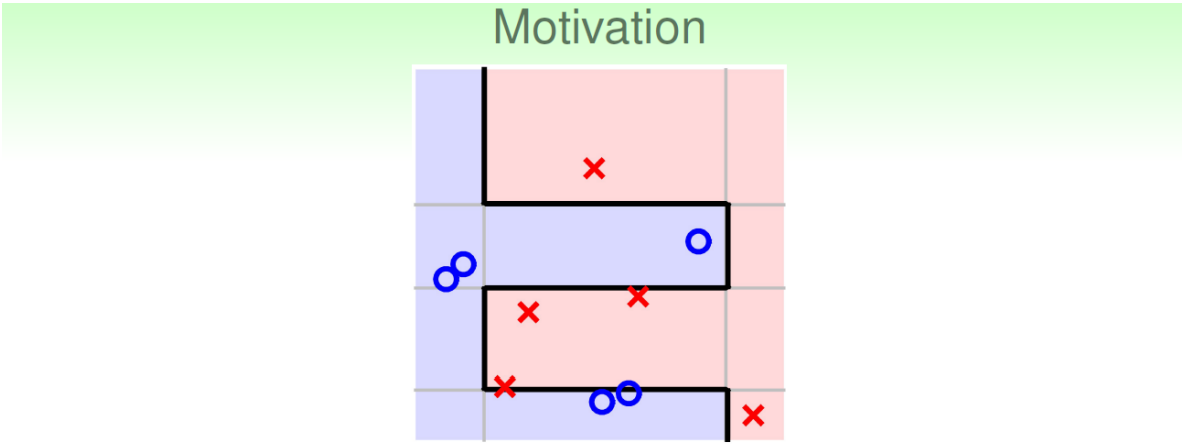
2. Jack: Circular!

   Teacher: Almost Right! Let's look at the rest! (Then enlarge the *MISTAKES*!) And Then merge the 2 principles!

3. Jessica: Stem!

   Teacher: Almost Right! Let's look at the rest! (Then enlarge the *MISTAKES*!) And Then merge the 3 principles!

最后得出的其实是综合每个同学的结果的出来的模型选择，其中老师的作用是让这些错误的类别被放大以至于学生可以学习的更好。这一点在perceptron里面也可以体现出来，总结如下。



- students: simple hypotheses g_t (like vertical/horizontal lines)
- (Class): sophisticated hypothesis G (like black curve)
- Teacher: a tactic learning algorithm that **directs the students to focus on key examples**

接下来我们希望数学证明这个想法的可行性。

二、Diversity by re-weighting

1. 之气在bagging中我们是采用uniformly的方式也就是说每个g_t预测的权重都是1，现在我们希望对于这些权重进行一些改进，就好比老师对于那些错误的放大一般。我们对于错误进行一个权重衡量，看bootstrapping之后所有的点出现的次数作为实际衡量中的err权重

Bootstrapping as Re-weighting Process

$$\mathcal{D} = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), (\mathbf{x}_3, y_3), (\mathbf{x}_4, y_4)\}$$

$$\xrightarrow{\text{bootstrap}} \quad \tilde{\mathcal{D}}_t = \{(\mathbf{x}_1, y_1), (\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), (\mathbf{x}_4, y_4)\}$$

weighted E_{in} on $\mathcal{D}$

$$E_{\text{in}}^{\mathbf{u}}(h) = \frac{1}{4}\sum_{n=1}^{4} u_n^{(t)} \cdot [\![y_n \neq h(\mathbf{x}_n)]\!]$$

$(\mathbf{x}_1, y_1)$, $u_1 = 2$
$(\mathbf{x}_2, y_2)$, $u_2 = 1$
$(\mathbf{x}_3, y_3)$, $u_3 = 0$
$(\mathbf{x}_4, y_4)$, $u_4 = 1$

E_{in} on $\tilde{D}_t$

$$E_{\text{in}}^{0/1}(h) = \frac{1}{4}\sum_{(\mathbf{x},y)\in\tilde{D}_t} [\![y \neq h(\mathbf{x})]\!]$$

$(\mathbf{x}_1, y_1)$, $(\mathbf{x}_1, y_1)$
$(\mathbf{x}_2, y_2)$

$(\mathbf{x}_4, y_4)$

each diverse g_t in bagging:
by minimizing bootstrap-weighted error

2. 接下来考虑的是如何 re-weighting这些权重呢，因为我们一开始对于数据集boostrap应该和re-weighting一起进行所以我们应该想一下其他的方法。

$$g_t \leftarrow \underset{h\in\mathcal{H}}{\text{argmin}} \left(\sum_{n=1}^{N} u_n^{(t)} [\![y_n \neq h(\mathbf{x}_n)]\!]\right)$$

$$g_{t+1} \leftarrow \underset{h\in\mathcal{H}}{\text{argmin}} \left(\sum_{n=1}^{N} u_n^{(t+1)} [\![y_n \neq h(\mathbf{x}_n)]\!]\right)$$

if g_t 'not good' for $\mathbf{u}^{(t+1)}$ $\Longrightarrow$ g_t-like hypotheses not returned as g_{t+1}
$\Longrightarrow$ g_{t+1} diverse from g_t

我们迭代的想如果g_t对于$u^{(t+1)}$不太好我们就应该应该返回一个更好的g_{t+1}

3. 我们回忆一下啊抛硬币的过程，这个时候下面的式子应该是1/2，也就是说我们希望随机性是最大的，这样我们的diversity才能最大。那么我们试图打到下面式子的条件！

idea: construct $\mathbf{u}^{(t+1)}$ to make g_t random-like

$$\frac{\sum_{n=1}^{N} u_n^{(t+1)} [\![y_n \neq g_t(\mathbf{x}_n)]\!]}{\sum_{n=1}^{N} u_n^{(t+1)}} = \frac{1}{2}$$

4. 过程如下：

'Optimal' Re-weighting

want: $\frac{\sum_{n=1}^{N} u_n^{(t+1)} [\![y_n \neq g_t(\mathbf{x}_n)]\!]}{\sum_{n=1}^{N} u_n^{(t+1)}} = \frac{\blacksquare_{t+1}}{\blacksquare_{t+1} + \bullet_{t+1}} = \frac{1}{2}$, where

$$\blacksquare_{t+1} = \sum_{n=1}^{N} u_n^{(t+1)} [\![y_n \neq g_t(\mathbf{x}_n)]\!], \bullet_{t+1} = \sum_{n=1}^{N} u_n^{(t+1)} [\![y_n = g_t(\mathbf{x}_n)]\!]$$

- need: $\underbrace{(\text{total } u_n^{(t+1)} \text{ of incorrect})}_{\blacksquare_{t+1}} = \underbrace{(\text{total } u_n^{(t+1)} \text{ of correct})}_{\bullet_{t+1}}$
- one possibility by **re-scaling (multiplying) weights**, if

(total $u_n^{(t)}$ of incorrect) = 1126 ; (weighted incorrect rate) = $\frac{1126}{7337}$	(total $u_n^{(t)}$ of correct) = 6211 ; (weighted correct rate) = $\frac{6211}{7337}$
incorrect: $u_n^{(t+1)} \leftarrow u_n^{(t)} \cdot 6211$	correct: $u_n^{(t+1)} \leftarrow u_n^{(t)} \cdot 1126$

'optimal' re-weighting under weighted incorrect rate ϵ_t:
multiply incorrect $\propto (1 - \epsilon_t)$; multiply correct $\propto \epsilon_t$

简言之就是每一次re-scaling，通过乘以对立面的系数来达到这一点。于是我们实现了diversity。

三、 Adaptive Boosting

1. 接下来我们讲一下这个算法的实现过程：利用Scaling Factor来实现re-weighting

Scaling Factor

'optimal' re-weighting: let $\epsilon_t = \frac{\sum_{n=1}^{N} u_n^{(t)} [\![y_n \neq g_t(\mathbf{x}_n)]\!]}{\sum_{n=1}^{N} u_n^{(t)}}$,

multiply incorrect $\propto (1 - \epsilon_t)$; multiply correct $\propto \epsilon_t$

define scaling factor $\blacklozenge_t = \sqrt{\frac{1-\epsilon_t}{\epsilon_t}}$

$$\begin{aligned} \text{incorrect} &\leftarrow \text{incorrect} \cdot \blacklozenge_t \\ \text{correct} &\leftarrow \text{correct} / \blacklozenge_t \end{aligned}$$

- **equivalent** to optimal re-weighting
- $\blacklozenge_t \geq 1$ iff $\epsilon_t \leq \frac{1}{2}$
  —physical meaning: **scale up incorrect**; **scale down correct**
  —like what Teacher does

2. 一个算法的初步想法大概如下：

A Preliminary Algorithm

$\mathbf{u}^{(1)} = ?$
for $t = 1, 2, \ldots, T$

1. obtain g_t by $\mathcal{A}(\mathcal{D}, \mathbf{u}^{(t)})$,
   where $\mathcal{A}$ tries to minimize $\mathbf{u}^{(t)}$-weighted 0/1 error
2. update $\mathbf{u}^{(t)}$ to $\mathbf{u}^{(t+1)}$ by $\blacklozenge_t = \sqrt{\frac{1-\epsilon_t}{\epsilon_t}}$,
   where ϵ_t = weighted error (incorrect) rate of g_t

return $G(\mathbf{x}) = ?$

- want g_1 'best' for E_{in}: $u_n^{(1)} = \frac{1}{N}$
- $G(\mathbf{x})$:
  - uniform? but g_2 very bad for E_{in} (**why? :-**)
  - linear, non-linear? **as you wish**

next: a special algorithm to aggregate
linearly on the fly with theoretical guarantee

我们考虑一下$u^{(1)}$的初始化，就希望最简单——$\frac{1}{N}$，至于G(x)的方法其实都可以用，我们的adaptive boosting用的是non-uniformly的α_t。

3. AdaBoosting的算法实现：

Linear Aggregation on the Fly

$\mathbf{u}^{(1)} = [\frac{1}{N}, \frac{1}{N}, \cdots, \frac{1}{N}]$
for $t = 1, 2, \ldots, T$

1. obtain g_t by $\mathcal{A}(\mathcal{D}, \mathbf{u}^{(t)})$, where ...
2. update $\mathbf{u}^{(t)}$ to $\mathbf{u}^{(t+1)}$ by $\blacklozenge_t = \sqrt{\frac{1-\epsilon_t}{\epsilon_t}}$, where ...
3. compute $\alpha_t = \ln(\blacklozenge_t)$

return $G(\mathbf{x}) = \text{sign}\left(\sum_{t=1}^{T} \alpha_t g_t(\mathbf{x})\right)$

- wish: large α_t for 'good' $g_t \Longleftarrow \alpha_t = \text{monotonic}(\blacklozenge_t)$
- will take $\alpha_t = \ln(\blacklozenge_t)$
  - $\epsilon_t = \frac{1}{2} \Longrightarrow \blacklozenge_t = 1 \Longrightarrow \alpha_t = 0$ (bad g_t zero weight)
  - $\epsilon_t = 0 \Longrightarrow \blacklozenge_t = \infty \Longrightarrow \alpha_t = \infty$ (super g_t superior weight)

Adaptive Boosting = weak base learning algorithm $\mathcal{A}$ (Student)
+ optimal re-weighting factor $\blacklozenge_t$ (Teacher)
+ 'magic' linear aggregation α_t (Class)

总结一下AdaBoosting做的事情：

- 学生：基本的算法 base learning algorithm
- 老师：Scaling Factor放大错误来更好的学习
- 班级: 用on the fly 的方式顺便学习了 α_t:

$$\alpha_t = \ln(\sqrt{\frac{1-\epsilon_t}{\epsilon_t}}) \tag{1}$$

最后学习到的 $G(x) = sign(\sum_{t=1}^{T} \alpha_t g_t(x))$

Adaptive Boosting (AdaBoost) Algorithm

$\mathbf{u}^{(1)} = [\frac{1}{N}, \frac{1}{N}, \cdots, \frac{1}{N}]$
for $t = 1, 2, \ldots, T$

1. obtain g_t by $\mathcal{A}(\mathcal{D}, \mathbf{u}^{(t)})$,
   where $\mathcal{A}$ tries to minimize $\mathbf{u}^{(t)}$-weighted 0/1 error
2. update $\mathbf{u}^{(t)}$ to $\mathbf{u}^{(t+1)}$ by

$$[\![y_n \neq g_t(\mathbf{x}_n)]\!] \text{ (incorrect examples)}: \quad u_n^{(t+1)} \leftarrow u_n^{(t)} \cdot \blacklozenge_t$$
$$[\![y_n = g_t(\mathbf{x}_n)]\!] \text{ (correct examples)}: \quad u_n^{(t+1)} \leftarrow u_n^{(t)} / \blacklozenge_t$$

   where $\blacklozenge_t = \sqrt{\frac{1-\epsilon_t}{\epsilon_t}}$ and $\epsilon_t = \frac{\sum_{n=1}^{N} u_n^{(t)} [\![y_n \neq g_t(\mathbf{x}_n)]\!]}{\sum_{n=1}^{N} u_n^{(t)}}$

3. compute $\alpha_t = \ln(\blacklozenge_t)$

return $G(\mathbf{x}) = \text{sign}\left(\sum_{t=1}^{T} \alpha_t g_t(\mathbf{x})\right)$

当然A的选择也至关重要，可以有的选择很多。

四、Theoretical Guarantee

最后是根据VC bound的出来的一些理论上的保证

Theoretical Guarantee of AdaBoost

- From VC bound

$$E_{\text{out}}(G) \leq E_{\text{in}}(G) + O\left(\sqrt{\underbrace{O(d_{\text{VC}}(\mathcal{H}) \cdot T \log T)}_{d_{\text{VC}} \text{ of all possible } G} \cdot \frac{\log N}{N}}\right)$$

- **first term can be small**:
  $E_{\text{in}}(G) = 0$ after $T = O(\log N)$ iterations if $\epsilon_t \leq \epsilon < \frac{1}{2}$ always
- **second term can be small**:
  overall d_{VC} grows "slowly" with T

boosting view of AdaBoost:
if $\mathcal{A}$ is weak but always **slightly better than random** ($\epsilon_t \leq \epsilon < \frac{1}{2}$),
then (AdaBoost+$\mathcal{A}$) can be strong ($E_{\text{in}} = 0$ and E_{out} small)