

Blending and Bagging

主要内容：如何将不同的hypothesis和features结合起来，让模型更好。

1.Motivation of Aggregation

You can ...

- **select** the most trust-worthy friend from their **usual performance**
—**validation!**
- **mix** the predictions from all your friends **uniformly**
—let them **vote!**
- **mix** the predictions from all your friends **non-uniformly**
—let them vote, but **give some more ballots**
- **combine** the predictions **conditionally**
—if [**t satisfies some condition**] give some ballots to friend t
- ...

第一种方法对应的模型：

$$G(x) = g_{t_*}(x) \text{ with } t_* = \operatorname{argmin}_{t \in \{1, 2, \dots, T\}} E_{\text{val}}(g_t^-)$$

第二种方法对应的模型：

$$G(x) = \operatorname{sign}\left(\sum_{t=1}^T 1 \cdot g_t(x)\right)$$

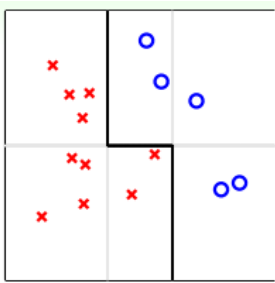
第三种方法对应的模型：

$$G(x) = \operatorname{sign}\left(\sum_{t=1}^T \alpha_t \cdot g_t(x)\right) \text{ with } \alpha_t \geq 0$$

第四种方法对应的模型：

$$G(x) = \operatorname{sign}\left(\sum_{t=1}^T q_t(x) \cdot g_t(x)\right) \text{ with } q_t(x) \geq 0$$

- **select** the most trust-worthy friend from their **usual performance**
 $G(\mathbf{x}) = g_{t_*}(\mathbf{x})$ with $t_* = \operatorname{argmin}_{t \in \{1, 2, \dots, T\}} E_{\text{val}}(g_t^-)$
- **mix** the predictions from all your friends **uniformly**
 $G(\mathbf{x}) = \operatorname{sign}\left(\sum_{t=1}^T 1 \cdot g_t(\mathbf{x})\right)$
- **mix** the predictions from all your friends **non-uniformly**
 $G(\mathbf{x}) = \operatorname{sign}\left(\sum_{t=1}^T \alpha_t \cdot g_t(\mathbf{x})\right)$ with $\alpha_t \geq 0$
 - include **select**: $\alpha_t = \llbracket E_{\text{val}}(g_t^-) \text{ smallest} \rrbracket$
 - include **uniformly**: $\alpha_t = 1$
- **combine** the predictions **conditionally**
 $G(\mathbf{x}) = \operatorname{sign}\left(\sum_{t=1}^T q_t(\mathbf{x}) \cdot g_t(\mathbf{x})\right)$ with $q_t(\mathbf{x}) \geq 0$
 - include **non-uniformly**: $q_t(\mathbf{x}) = \alpha_t$



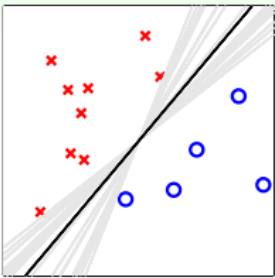
如果要求只能用一条水平的线或者垂直的线进行分类（即上述第一种方法：validation），那不论怎么选取直线，都达不到最佳的分类效果。

如果可以使用aggregate，比如一条水平线和两条垂直线组合而成的图中折线形式，就可以将所有的点完全分开，得到了最优化的预测模型。

- mix **different weak hypotheses** uniformly
— $G(\mathbf{x})$ 'strong'
- aggregation
⇒ **feature transform (?)**

aggregation提高了预测模型的power，起到了特征转换的效果。

使用PLA算法，可以得到很多满足条件的分类线



aggregation也起到了正则化的效果，让预测模型更具有代表性。

- mix **different random-PLA hypotheses** uniformly
— $G(\mathbf{x})$ 'moderate'
- aggregation
⇒ **regularization (?)**

aggregation的两个优势：feature transform和regularization。

feature transform和regularization是对立的。

如果进行feature transform，那么regularization的效果通常很差，反之亦然。也就是说，单一模型通常只能倾向于feature transform和regularization之一，在两者之间做个权衡。

aggregation能将feature transform和regularization各自的优势结合起来，从而得到不错的预测模型。

2.Uniform Blending

uniform blending，应用于classification分类问题，做法是将每一个可能的矩赋予权重1，进行投票，得到的 $G(\mathbf{x})$ 表示为：

$$g(\mathbf{x}) = \text{sign}\left(\sum_{t=1}^T 1 \cdot g_t(\mathbf{x})\right)$$

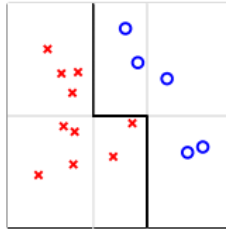
这种方法对应三种情况：

- 1.每个候选的矩 g_t 都完全一样，这跟选其中任意一个 g_t 效果相同
- 2.每个候选的矩 g_t 都有一些差别，通过投票的形式使多数意见修正少数意见，从而得到很好的模型

3. 多分类问题，选择投票数最多的那一类。

- same g_t (autocracy):
as good as one single g_t
- very different g_t (diversity + democracy):
majority can correct minority
- similar results with uniform voting for multiclass

$$G(\mathbf{x}) = \operatorname{argmax}_{1 \leq k \leq K} \sum_{t=1}^T \mathbb{I}[g_t(\mathbf{x}) = k]$$



uniform blending应用于regression，将所有的 g_t 求平均值：

$$G(x) = \frac{1}{T} \sum_{t=1}^T g_t(x)$$

uniform blending for regression对应两种情况：

1. 每个候选的 g_t 都完全一样，这跟选其中任意一个 g_t 效果相同
2. 每个候选的 g_t 都有一些差别，有的 $g_t > f(x)$ ，有的 $g_t < f(x)$ ，此时求平均值的操作可能会消去这种大于和小于的影响，从而得到更好的回归模型。因此，一般来说，求平均值的操作更加稳定，更加准确。

- same g_t (autocracy):
as good as one single g_t
- very different g_t (diversity + democracy):
some $g_t(\mathbf{x}) > f(\mathbf{x})$, some $g_t(\mathbf{x}) < f(\mathbf{x})$
 \Rightarrow average could be more accurate than individual

计算 g_t 的平均值可能比单一的 g_t 更稳定，更准确：

$$\begin{aligned} \operatorname{avg}((g_t(\mathbf{x}) - f(\mathbf{x}))^2) &= \operatorname{avg}(g_t^2 - 2g_t f + f^2) \\ &= \operatorname{avg}(g_t^2) - 2Gf + f^2 \\ &= \operatorname{avg}(g_t^2) - G^2 + (G - f)^2 \\ &= \operatorname{avg}(g_t^2) - 2G^2 + G^2 + (G - f)^2 \\ &= \operatorname{avg}(g_t^2 - 2g_t G + G^2) + (G - f)^2 \\ &= \operatorname{avg}((g_t - G)^2) + (G - f)^2 \end{aligned}$$

$$\begin{aligned} \operatorname{avg}(E_{\text{out}}(g_t)) &= \operatorname{avg}(\mathcal{E}(g_t - G)^2) + E_{\text{out}}(G) \\ &\geq E_{\text{out}}(G) \end{aligned}$$

$\operatorname{avg}(E_{\text{out}}(g_t)) \geq E_{\text{out}}(G)$ ，从而证明了从平均上来说，计算 g_t 的平均值 $G(t)$ 要比单一的 g_t 更接近目标函数 f ，regression效果更好。

G 是数目为 T 的 g_t 的平均值。令包含 N 个数据的样本 D 独立同分布于 P^N ，每次从新的 D_t 中学习得到新的 g_t ，在对 g_t 求平均得到 G ，当做无限多次，即 T 趋向于无穷大的时候：

consider a virtual iterative process that for $t = 1, 2, \dots, T$

- ① request size- N data D_t from P^N (i.i.d.)
- ② obtain g_t by $\mathcal{A}(D_t)$

$$\bar{g} = \lim_{T \rightarrow \infty} G = \lim_{T \rightarrow \infty} \frac{1}{T} \sum_{t=1}^T g_t = \mathcal{E}_{\mathcal{D}} \mathcal{A}(\mathcal{D})$$

当 T 趋于无穷大的时候， $G = \bar{g}$ ，则有如下等式成立：

$$\text{avg}(E_{\text{out}}(g_t)) = \text{avg}(\mathcal{E}(g_t - \bar{g})^2) + E_{\text{out}}(\bar{g})$$

expected performance of \mathcal{A} = expected deviation to consensus
+ performance of consensus

- performance of consensus: called **bias**
- expected deviation to consensus: called **variance**

一个演算法的平均表现可以被拆成两项，一个是所有 g_t 的共识，一个是不同 g_t 之间的差距是多少，即 bias 和 variance。
uniform blending 的操作时求平均的过程，削弱了上式第一项 variance 的值，从而演算法的表现就更好了，能得到更加稳定的表现。

3. Linear and Any Blending

linear blending, 每个 g_t 赋予的权重 α_t 不同, $\alpha_t \geq 0$ 。

我们最终得到的预测结果等于所有 g_t 的线性组合:

linear blending: known g_t , each to be given α_t ballot

$$G(\mathbf{x}) = \text{sign}\left(\sum_{t=1}^T \alpha_t \cdot g_t(\mathbf{x})\right) \text{ with } \alpha_t \geq 0$$

computing 'good' α_t : $\min_{\alpha_t \geq 0} E_{\text{in}}(\alpha)$

linear blending for regression

$$\min_{\alpha_t \geq 0} \frac{1}{N} \sum_{n=1}^N \left(y_n - \sum_{t=1}^T \alpha_t g_t(\mathbf{x}_n) \right)^2$$

LinReg + transformation

$$\min_{w_i} \frac{1}{N} \sum_{n=1}^N \left(y_n - \sum_{i=1}^{\tilde{d}} w_i \phi_i(\mathbf{x}_n) \right)^2$$

like two-level learning, remember? :-)

linear blending = LinModel + hypotheses as transform + constraints

利用误差最小化的思想，找出最佳的 α_t ，使 $E_{\text{in}}(\alpha)$ 取最小值。

linear blending = LinModel + hypotheses as transform + constraints:

$$\min_{\alpha_t \geq 0} \frac{1}{N} \sum_{n=1}^N \text{err} \left(y_n, \sum_{t=1}^T \alpha_t g_t(\mathbf{x}_n) \right)$$

先计算 $g_t(x_n)$ ，再进行 linear regression 得到 α_t 值。

$\alpha_t < 0$ 并不会影响分类效果，只需要将正类看成负类，负类当成正类即可。

可以把 $\alpha_t \geq 0$ 这个条件舍去，这样 linear blending 就可以使用常规方法求解。

linear blending for binary classification

$$\text{if } \alpha_t < 0 \implies \alpha_t g_t(\mathbf{x}) = |\alpha_t| (-g_t(\mathbf{x}))$$

- negative α_t for $g_t \equiv$ positive $|\alpha_t|$ for $-g_t$
- if you have a stock up/down classifier with 99% error, tell me! :-)

in practice, often

linear blending = LinModel + hypotheses as transform ~~+ constraints~~

Linear Blending 中使用的 g_t 是通过模型选择而得到的，利用 validation，从 D_{train} 中得到 $g_1^-, g_2^-, \dots, g_T^-$ 。

将 D_{train} 中每个数据点经过各个矩的计算得到的值，代入到相应的 linear blending 计算公式中，迭代优化得到对应 α 值。

利用所有样本数据，得到新的 g_t 代替 g_t^- ，则 $G(t)$ 就是 g_t 的线性组合而不是 g_t^- ，系数是 α_t 。

Given $g_1^-, g_2^-, \dots, g_T^-$ from $\mathcal{D}_{\text{train}}$, transform (\mathbf{x}_n, y_n) in \mathcal{D}_{val} to $(\mathbf{z}_n = \Phi^-(\mathbf{x}_n), y_n)$, where $\Phi^-(\mathbf{x}) = (g_1^-(\mathbf{x}), \dots, g_T^-(\mathbf{x}))$

Linear Blending

- 1 compute α
= $\text{LinearModel}(\{(\mathbf{z}_n, y_n)\})$
- 2 return $G_{\text{LINB}}(\mathbf{x}) = \text{LinearHypothesis}_\alpha(\Phi(\mathbf{x}))$,

Any Blending (Stacking)

- 1 compute \tilde{g}
= $\text{AnyModel}(\{(\mathbf{z}_n, y_n)\})$
- 2 return $G_{\text{ANYB}}(\mathbf{x}) = \tilde{g}(\Phi(\mathbf{x}))$,

where $\Phi(\mathbf{x}) = (g_1(\mathbf{x}), \dots, g_T(\mathbf{x}))$

any blending:

- **powerful**, achieves conditional blending
- but **danger of overfitting**, as always :-)

linear blending中, $G(t)$ 是 $g(t)$ 的线性组合; any blending中, $G(t)$ 可以是 $g(t)$ 的任何函数形式(非线性),这种形式的blending也叫做 Stacking。

any blending的优点是模型复杂度提高, 更容易获得更好的预测模型; 缺点是复杂模型也容易带来过拟合的危险。

在使用any blending的过程中要时刻注意避免过拟合发生, 通过采用regularization的方法, 让模型具有更好的泛化能力。

4. Bagging(Bootstrap Aggregation)

blending的做法就是将已经得到的矩 g_t 进行aggregate的操作。具体的aggregation形式包括: uniform, non-uniform和conditional。

blending: aggregate **after getting g_t** ;
learning: aggregate **as well as getting g_t**

aggregation type	blending	learning
uniform	voting/averaging	?
non-uniform	linear	?
conditional	stacking	?

learning g_t for uniform aggregation: **diversity** important

- **diversity** by different models: $g_1 \in \mathcal{H}_1, g_2 \in \mathcal{H}_2, \dots, g_T \in \mathcal{H}_T$
- **diversity** by different parameters: GD with $\eta = 0.001, 0.01, \dots, 10$
- **diversity** by algorithmic randomness:
random PLA with different random seeds
- **diversity** by data randomness:
within-cross-validation hypotheses $g_{\bar{v}}$

可以选取不同模型 H ; 可以设置不同的参数, 例如 η 、迭代次数 n 等; 可以由算法的随机性得到, 例如PLA、随机种子等; 可以选择不同的数据样本等。这些方法都可能得到不同的 g_t 。

expected performance of \mathcal{A} = expected deviation to consensus
+ performance of consensus
consensus \bar{g} = expected g_t from $\mathcal{D}_t \sim P^N$

\bar{g} 是在矩个数 T 趋向于无穷大的时候, 不同的 g_t 计算平均得到的值。

这里我们为了得到 \bar{g} , 做两个近似条件:

1. 有限的 T ;
2. 由已有数据集 D 构造出 $D_t \sim P^N$, 独立同分布

第一个条件没有问题, 第二个近似条件的做法就是bootstrapping。

bootstrapping是统计学的一个工具, 思想就是从已有数据集 D 中模拟出其他类似的样本 D_t 。

- **consensus** more stable than direct $\mathcal{A}(\mathcal{D})$, but comes from many more \mathcal{D}_t than the \mathcal{D} on hand
- want: approximate \bar{g} by
 - finite (large) T
 - approximate $g_t = \mathcal{A}(\mathcal{D}_t)$ from $\mathcal{D}_t \sim P^N$ using only \mathcal{D}

bootstrapping: a statistical tool that re-samples from \mathcal{D} to 'simulate' \mathcal{D}_t

bootstrapping的做法是，假设有N笔资料，先从中选出一个样本，再放回去，再选择一个样本，再放回去，共重复N次。

这样我们就得到了一个新的N笔资料，这个新的 $\tilde{\mathcal{D}}_t$ 中可能包含原D里的重复样本点，也可能没有原D里的某些样本， $\tilde{\mathcal{D}}_t$ 与D类似但又不完全相同。

抽取-放回的操作不一定非要是N，次数可以任意设定。

利用bootstrap进行aggregation的操作就被称为bagging。

virtual aggregation

consider a **virtual** iterative process that for $t = 1, 2, \dots, T$

- 1 request size- N data \mathcal{D}_t from P^N (i.i.d.)
 - 2 obtain g_t by $\mathcal{A}(\mathcal{D}_t)$
- $G = \text{Uniform}(\{g_t\})$

bootstrap aggregation

consider a **physical** iterative process that for $t = 1, 2, \dots, T$

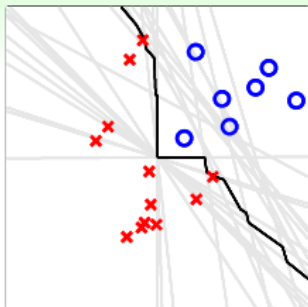
- 1 request size- N data $\tilde{\mathcal{D}}_t$ from **bootstrapping**
 - 2 obtain g_t by $\mathcal{A}(\tilde{\mathcal{D}}_t)$
- $G = \text{Uniform}(\{g_t\})$

eg: Bagging Pocket算法应用。

先通过bootstrapping得到25个不同样本集，再使用pocket算法得到25个不同的 g_t ，每个pocket算法迭代1000次。

再利用blending，将所有的 g_t 融合起来，得到最终的分类线。

虽然bootstrapping会得到差别很大的分类线（灰线），但是经过blending后，得到的分类线效果是不错的，则bagging通常能得到最佳的分类模型。



$$T_{\text{POCKET}} = 1000; T_{\text{BAG}} = 25$$

只有当演算法对数据样本分布比较敏感的情况下，才有比较好的表现。

5.summary

blending和bagging都属于aggregation，是将不同的 g_t 合并起来，利用集体的智慧得到更加优化的 $G(t)$ 。

Blending通常分为三种情况：Uniform Blending，Linear Blending和Any Blending。

其中，uniform blending采样最简单的“一人一票”的方法，linear blending和any blending都采用标准的two-level learning方法，类似于特征转换的操作，来得到不同 g_t 的线性组合或非线性组合。

利用bagging (bootstrap aggregation)，从已有数据集D中模拟出其他类似的样本 \mathcal{D}_t ，而得到不同的 g_t ，再合并起来，优化预测模型。