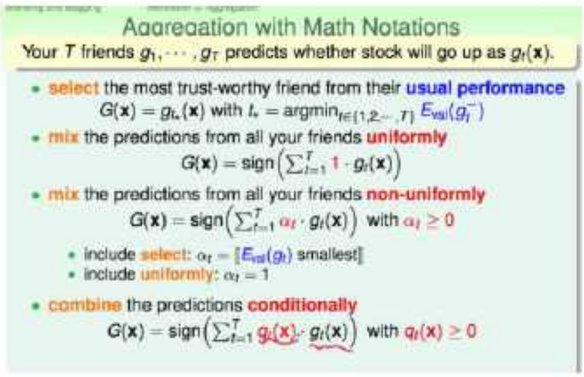
发件人：**Yulin Wu** yw4923@nyu.edu
主题：
日期：2020年9月9日 下午5:10
收件人：

Aggregation

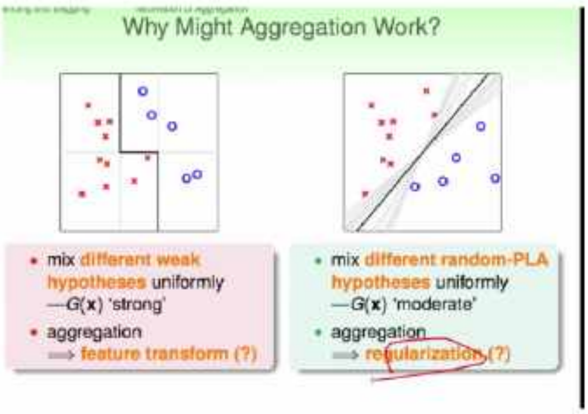
2020年8月28日 星期五
上午11:55

15个朋友给了你15个意见(model g_t), 你如何去运用你朋友的意见呢?



第一种情形:
选出一个最强的(model selection)。但若你的朋友都是弱弱的，就gg

而aggregation真正想做的是:
三个臭皮匠，胜过一个诸葛亮

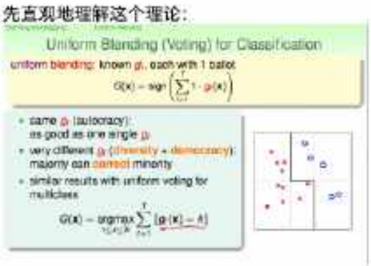


Aggregation includes: blending and bagging

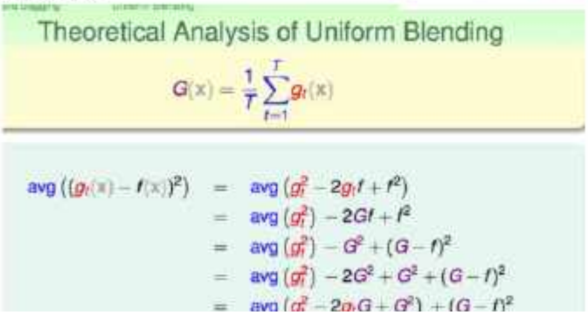
Uniform Blending:

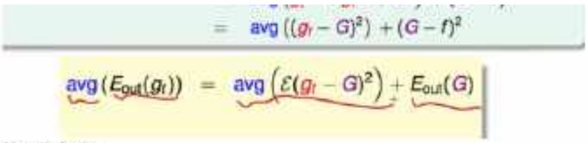
一个很直观的理论是:
g_t同质化严重，blending后的结果as good as one g; 差异大的g blending后才能有更好的效果

先直观地理解这个理论:

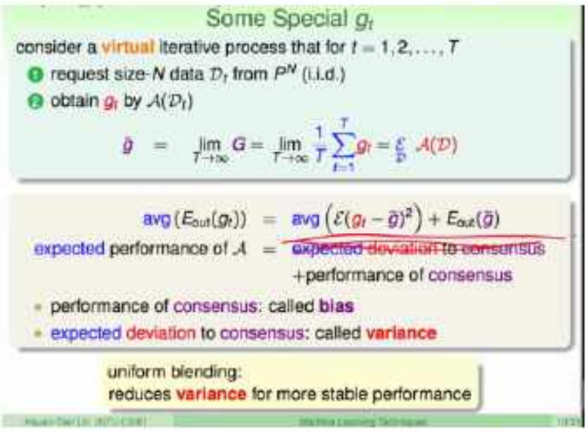


严谨地证明:

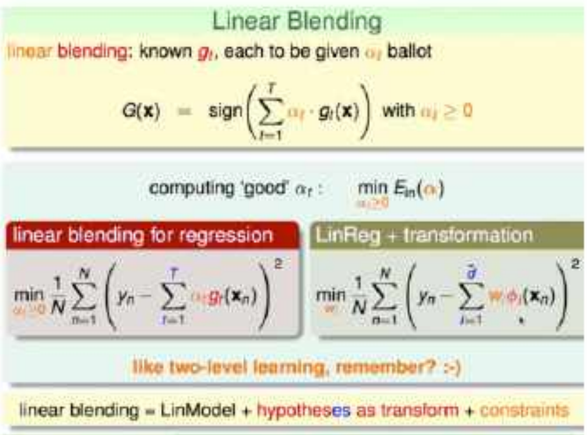




等式的左边:
对每一个x来看，g_t(x) - f(x) 的平方代表了g_t这个model在预测x上的error，可以用E_out(g_t)来代替。avg(E_out(g_t))就是，所有model预测x的偏差的平均。这里的意义是，如果你每次只选择一个g，那么你能期待的表现是avg(E_out(g_t))。
等式的右边:
(G-f)的平方代表了G这一个model在预测x上的error，而G是谁呢？就是uniform Blending后的产物，3个臭皮匠结合后的产物。(G-f)的平方就是3个臭皮匠结合后的预测。
等式说明:
blending后的预测（3个臭皮匠的结合）比每次只选择一个g来预测（臭皮匠单独预测）要好！而且要好avg((g_t - G)^2)那么多。
而且！avg((g_t - G)^2) 代表了g之间的方差。

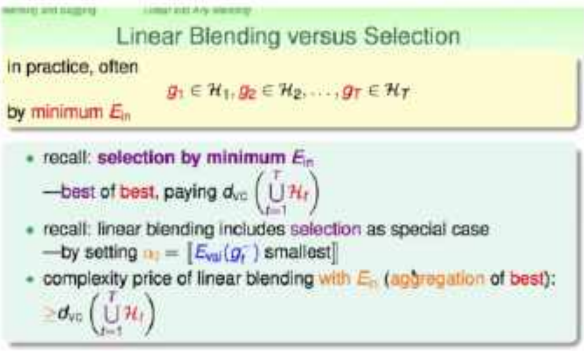


Linear Blending



其实这里的$\alpha \geq 0$的限制可以去掉。

ok,now, how to 训练这个model呢?
首先，关于如何解这个目标函数，要么用linear regression的解析解，要么用gradien descent.
然后，这个blending的input和output如何构造呢？还可以用原本训练集的数据吗？



1. D分为D_train & D_val, use D_train to train $g_1^-, g_2^-, ..., g_T^-$ 这一步可以看作feature transform.

If use g = [g_1^-, g_2^-, ...,g_T^-] directly as input, y_train as output to train the blending model（如何将朋友的意见加权平均）,what would happen?
注：model selection is a special blending, how does it work? It selects the best model based on validation error instead of in-sample error. Essentially, [g_1^-(Dval), ..., g_T^-(Dval)] as input, and y_val as output, 演算法就是手动选出最接近的那个g^-.

Similarly, if I use g as input and y_train as output, I will definitely choose the best model or construct a even more complicated model. (ex: 1*model_1 + 2 * model_2).
So I can't do this way.

Now I throw Dtrain out. 不再使用训练集了，只剩下验证集。

2. 模仿model selection，用g将验证集预测出来，so I have $z_{val} = [g_1^-(Dval), g_2^-(Dval), ..., g_T^-(Dval)]$
As input, and y_val as output. 在验证集上训练s.t. $\min_\alpha \sum (y_{val_n} - \alpha^T z_{val,n})$^2

3. Finally, obviously, 选出最优的hypothesis后(ex: the second model is uniform blending, and the first model is XGB, SVM, LASSO), 再用所有的D fit一次， 选出最优的w。

最后，如何构造g的diversity

learning g_t for uniform aggregation: diversity important

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

next: diversity by data randomness without g^-

Bagging:

Revisit of Bias-Variance

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

$$\text{consensus } \bar{g} = \text{expected } g_t \text{ from } \mathcal{D}_t \sim P^N$$

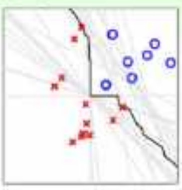
最理想的情况是:
1. 有无限多个g
2. 每个g都有新的数据来训练

但是，太理想了，现实中要妥协
1. finite but large g
2. use bootstrapping (resample N examples with replacement) 去生成新的数据

virtual aggregation	bootstrap aggregation
consider a **virtual** iterative process that for $t = 1, 2, \ldots, T$	consider a **physical** iterative process that for $t = 1, 2, \ldots, T$
1. request size-N data $\mathcal{D}_t$ from P^N (i.i.d.)	1. request size-N' data $\tilde{\mathcal{D}}_t$ from bootstrapping
2. obtain g_t by $\mathcal{A}(\mathcal{D}_t)$	2. obtain g_t by $\mathcal{A}(\tilde{\mathcal{D}}_t)$
$G = \text{Uniform}(\{g_t\})$	$G = \text{Uniform}(\{g_t\})$

Bagging 的效果:

Bagging Pocket in Action



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

- very diverse g_t from bagging
- proper non-linear boundary after aggregating binary classifiers