研究生算法课笔记

上课日期：2016 年 11 月 14 日　　第（2）节
组长：张　艺　1601111298
组员：陈潇漪　1601111276

● 内容概要

本节课主要分为以下几部分内容：

第一，对课后同学提出的问题进行了答疑；

第二，继续 Adaboost 的学习，解决了相关问题；

第三，介绍新的机器学习算法 Stacking；

第四，对于遗留的 XGBoost 问题进行了答疑。

● 课堂内容

一、　答疑

（1）Q：validation 集可以用来测试吗？

A：一般讲，训练数据分三份：

训练集(train set)：用来训练及确定模型参数。

验证集(validation set)：选择模型及调整参数，不可在其上面进行测试。

测试集(test set)：测试已经训练好的模型的拟合能力。

（2）Q: 多分类问题的返回值是什么？

A:（1）返回类别整数

如果对每个类别都有一个整数编号，如 1，2，3…，直接返回类型的相应编号即可。

（2）返回 one-hot 向量

如需返回一个 one-hot 向量，预测类型为 i，那么在该向量中第 i 个位置的数是 1，其余位置均未零，向量维数是总的类别个数。

（3）Q: 投票方式有哪些？

A:（1）hard voting:

每个子分类器都有一个分类结果，如果是某一类则为 1，不是该类则为 0（当然也可以是其它数，如-1），它的分类结果是一个整数，最后取投票数最多的结果作为最终分类。

（2）soft voting：

soft voting 的子分类器并不确定地指出它的分类结果是哪一个，而是给出样

例属于每个类的概率，它的分类结果是一个实数，表明样本属于该类的可信度，将每个类的所有概率相加取平均作为属于该类的最终概率，最后取概率最大的作为样本类别。

二、 Adaboost 补充问题

（1）Q：使用 Adaboost 算法时，需要每个样本有一个权重，如果子分类器是决策树，而决策树不支持样本权重，那这种情况下怎么办？原来的算法能否不改？

A：可以不改。仍采用不支持权重的算法，只需采样时按权重采样。例如一个样本权重是另外一个样本权重的 2 倍，则它有两倍于另外这个样本的权重被采中，多采样几次，样本的分布基本就是按权重来分配的，再按照不改的决策树算法去做，也是可以的。

（2）Q: Adaboost 每一轮需要做归一化，让最后样本总权重等于 1，如果不做归一化，最后总权重是变大了还是变小了？

A: 证明：

当不做归一化时，分对的样本权重是

$$(1-\varepsilon)/\sqrt{\frac{1-\varepsilon_t}{\varepsilon_t}} = (1-\varepsilon)*\sqrt{\frac{\varepsilon_t}{1-\varepsilon_t}} = \sqrt{\varepsilon_t(1-\varepsilon_t)}$$

∴ 分错的样本权重应和分对的样本权重一样

∴ 分错的样本权重也是$\sqrt{\varepsilon_t(1-\varepsilon_t)}$

∴ 总权重是 $2\sqrt{\varepsilon_t(1-\varepsilon_t)}$

又∵ $\varepsilon_t(1-\varepsilon_t) = -\left(\varepsilon_t-\frac{1}{2}\right)^2+\frac{1}{4}<\frac{1}{4}$

∴ $2\sqrt{\varepsilon_t(1-\varepsilon_t)} < 2*\frac{1}{2} = 1$

∴ 不做归一化后权重降低了。

（3）Q：Adaboost 在特征维度进行采样可以吗？

A：可以。无论是在数据维度采样，还是在特征维度采样，都可以增加子分类器多样性。但这种采样会降低子分类器准确度，对随机森林来讲，如果每个分类器正确率都比较低，则最终的投票结果也不会很好，但是对 Adaboost 来说，在保证分类结果比随机猜的结果好的前提下，子分类器性能弱一点没有关系，甚

至可能效果更好。

（4）Q：Adaboost 数据采样跟随机森林比，采样的比例上有什么不同呢？

讨论：

（1） 随机森林按 100%的比例有放回的采样；

（2） 如果 Adaboost 的数据采样比例太低，它在采样样本集上的准确率高，却可能在整个样本集的准确率小于 50%，导致算法不收敛。

（3） 如果保证准确率高于 50%的前提下，若 Adaboost 采样比例很高（80%以上），会有什么问题？

A：Adaboost 数据采样跟随机森林比要低一点。

随机森林树较高，因为如果基于数据扰动产生多样性的话，前提是模型要对数据扰动敏感；如果模型对数据扰动不敏感，那么很可能数据采样之后训练出来的模型是差不多的。因为采样时即使有的样本出现多一点，有的样本出现少一点，但是总的统计规律是不变的。那么统计规律不变的情况下，如果树比较矮，则 bias 比较高，对数据扰动不敏感。

最极端的例子：一个决策树桩，在数据里面加一些噪音，可能树桩根本不动。如果随机森林使用决策树桩，很可能采样后得到的是同一棵树。

Adaboost 树的高度较低，所以采样比例应该低一些。如果树的高度本身比较矮，而采样比例较高，那么很可能采样作用不明显。

三、Stacking:

(一) 关于 Stacking 的分类

把多个模型合在一起的方法都可以叫 Blending，Blending 分类如下：

（1）uniform blending/weighting

采用“一人一票制”，通过将子分类器的结果相加之后的结果作为对目标的预测值，这个投票的结果实际上反应了少数服从多数的原则。e.g., Random forest

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

（2）linear blending

通过给子分类器不同的票数，即将子分类器的结果赋予不同的权重，用线性组合后的结果作为最终结果。eg: Adaboost

$$G(X) = sign(\sum_{t=1}^{T} \alpha_t \cdot g_t(x))$$

（3）conditional blending

在不同的条件下，选择不同的子分类器，最后将结果合起来。eg: Decision Tree，节点根据取值不同要么是子树 A 要么是子树 B，每个子树可以看做一个单独的分类器，最后根据取值不同合在一起。

$$G(X) = sign(\sum_{t=1}^{T} q_t(x) \cdot g_t(x))$$

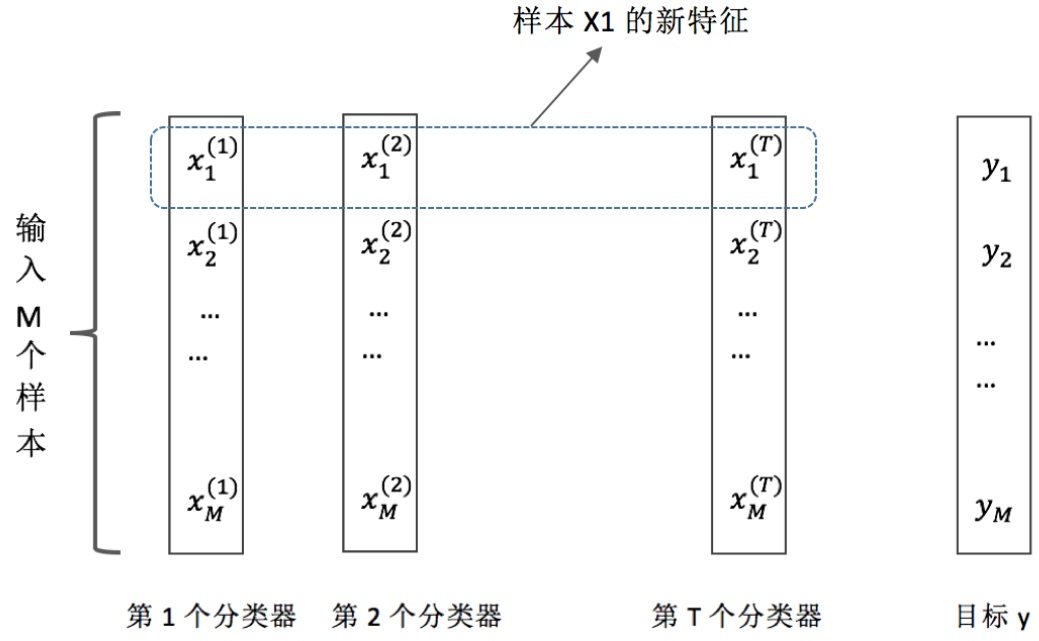
(4) any blending

Stacking

（二）stacking 算法

样本 x 本身有 n 维（$x_1, x_2, \ldots, x_n$），共有 M 个样本，有 T 个子分类器 $g_1(x), g_2(x), \ldots, g_T(x)$，对 M 个样本中的每个样本都按照第一个分类器算一遍，得到 M 组数；再按照第二个分类器算一遍（不一定是分类器，也可能是回归的），以此类推，得到第 T 个 M 组数。目标值 y 不变，再用另外一个分类器（或回归器）在第二层去拟合 y。

第一层是 base learner，第二层是 meta learner，将训练集输入到 T 个 base learner，产生这些输出再作为输入进入 meta learner，最后产生最终的结果。



这种从原始输入数据到第一层分类器输出结果的转换过程可以看做是 feature transformation（特征转换）。原来每个样本是 n 维特征（$x_1, x_2, \ldots, x_n$），经过转换，每一个子分类器/一级模型 base learner 可以产生一个特征，T 个 base

learner 产生了 t 维新的特征，x 的 n 维原始特征（$x_1, x_2 , \dots , x_n$）转化为新的 T 维特征（$x^1, x^2, \dots , x^T$）。

优点：可以把原来的 random forest、Adaboost 等方法都可以看做 stacking 的一种特殊情况。则随机森林相当于第二级 meta learner 将第一级 base learner 的结果直接相加；而 Adaboost 第二层的 meta learner 是将前面的结果进行线性组合加权，其加权方法的特殊之处在于第二级的 meta learner 不需要重新学权重系数，而是根据第一级输出的准确率自动生成，保证在一定意义上最优。一般情况的二层 meta learner 是需要学习的。

若把多个模型 stack 在一起，将其一层层训练至多层，其**缺点**在于：缺乏效率，因为一层层训练太慢；容易过拟合，因为一层层累加的时候模型的复杂度很高，会把训练集都记下来。

（三）Meta learner 训练方法

(x) 错误做法：有一个训练集，用每一个 base learner 在上面训练之后得到一个模型，这个模型可将训练集的内容都转换为 t 维模型。再拿 meta learner 训练，将结果进行组合。

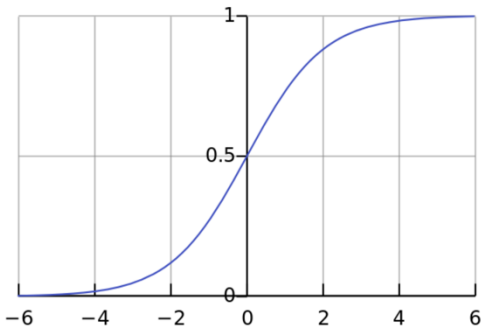
(√) 正确做法：Base learner 训练数据集之后，找另外一个没见过的数据集再进行 feature transformation，然后 meta learner 在这个新的数据集上进行组合，保证在新的数据集上准确率较高的确实是泛化能力较强的，而不是把原来的训练集记下来的。

四、XGBoost 问题解释

（一）sigmoid 函数

（1） 定义

sigmoid 函数是一种 S 型函数，也叫挤压（squash）函数，定义$S(x) = \frac{1}{1+e^{-x}}$，图像如下：



（2）作用：

把负无穷到正无穷的数，压缩到 0 到 1 之间。

（3）优势

1. 可以无限多次求导；
2. 导数形式非常简单。

（二）XGBoost 步长解释

真实值	第一次预测值	残差（第二次拟合的目标值）
50	45	5
106	97	9
13	22	-9
2	5	-3

如果未修改步长，第一次预测值如上图所示，第二次就会按残差来拟合，如果修改了步长使第一次预测值成倍减小，这样残差会大很多倍，第二次则会按新的残差来拟合。

真实值	第一次预测值	残差（第二次拟合的目标值）
50	4.5	45.5
106	9.7	96.3
13	2.2	10.8
2	0.5	1.5

Adaboost 乃至 boosting，在保证子分类器的分类效果比随机猜好的情况下，子分类器弱一些没有关系。用步长的方法，人为地让 base learner 更弱，降低同样幅度的残差则需要更多的子分类器，增加了子分类器多样性，最终效果也会更好。