基于 Stacking 框架的学习机制研究*
A Researches on Learning Mechanism Based on Stacking Framework

韦艳艳，李陶深
Wei Yanyan, Li Taoshen
（广西大学计算机与电子信息学院，广西南宁 530004）
(Coll. of Comp. & Elec. Info., Guangxi Univ., Nanning, Guangxi, 530004, China)

摘要：分析 Stacking 框架的基本原理，T^1 空间的数据表示和 Stacking 的缺陷，认为基于 Stacking 框架的学习能够有效地提高学习效果，但是在分类器个数增大时可能会导致元层训练数据规模增加。提出对底层分类器输出的后验概率用加权平均的方法构造元训练样本，减少二次建模的时间开销。该方法能够弥补由于对平均后验概率进行简单平均而丧失的模型输出特征，纠正分类偏差。

关键词：学习机制 Stacking 元学习 分类 分类器组合
中图法分类号：TP391.6

Abstract: One of disadvantages in Stacking method of the classifier is that the size of meta training examples increases when the number of base classifiers goes up. An approach to overcome this disadvantage is presented, in which the weighted average distribution of the posterior probability of the classifiers is used to form the meta training set. Experiment result shows that this approach can improve the output character lost caused by the average distribution of the posterior probability, and correct the mistakes made by base classifiers.

Key words: learning mechanism, stacking, meta learning, classification, classifiers combination

机器学习领域的分类学习模型组合集成问题已经成为适应分布式计算以及提高分类准确率而需要解决的重要课题[1]. 在一组分类模型生成以后，就产生如何进行组合的问题. 分类模型组合的最终目的是使组合后的模型具有比单一模型更高的精度. 由于训练过程中所使用的数据集合不相同，以及学习方法不一样，因此归纳得到的各个分类模型的分类效果有差别，有时具有一定的互补性. 为了达到对分类器进行有效组合的目的，需要使用一些数据特征信息加以调整.

多分类器的组合方式有两类：一类是级联式，代表算法是 Boosting[1]；另一类是并联组合方法，最简单也最为常用的组合方法是多数表决（Majority Voting）[2]，它让每一个底层模型对样本 X 的分类进行表决，样本 X 的最终类就是表决结果票数最多的类. 并联式的另一种组合框架就是 Wolpert 于 1992 年提出的叠加法（Stacking）[3]，也称为 Stacked Generalization. 它是利用前一层模型的输出结果作为下一层的学习输入信息，使得前一次的学习能够充分用于后面的归纳过程，从而发现并且纠正所使用的学习算法中的系统偏差，提高学习的精度. Stacking 具有较强的可扩展性，组合的层次可以从一层向多层向上伸延. Stacking 方法的缺点之一是分类器个数增大时会导致元训练数据规模增加. 针对这一情况，本文提出对底层分类器输出的后验概率用加权平均的方法构造元训练样本，它能在一定程度上弥补由于对平均后验概率进行简单平均而丧失的模型输出特征，纠正分类偏差.

2004-05-17 收稿。
*广西留学回国人员科学基金项目（桂科回 0342001）和广西教育厅科技项目（桂教科研[2001]401 号）联合资助。

1 Stacking 框架

1.1 基本原理

Stacking 是一种叠加组合的通用框架，该框架将组合问题看作是一个归纳过程，它的输入空间是（或近似于）学习模型的输出结果. 具体描述如下：设 T^0 空间上的数据集合表示为 $T^0 = \{(X_m, Y_m,), m = 1, \cdots, M\}$，$F$ 是由 T^0 构造的函数（模型），$F = \{\overline{f}_i(X), i = 1, \cdots, N\}$ 中的每一个元素都近似 $f(X)$. 即：

$$T^1 = \{((\overline{f}_1(X_i), \overline{f}_2(X_i), \cdots, \overline{f}_N(X)), y_i), i =$$

$1,\cdots,N\}$,

模型的近似输出表示为$\overline{f}_n(X_i)$,它下面方法生成:

(1)T^0空间上的数据划分成v个部分;

(2)对于每一个数据划分v:

1)用学习算法对v进行训练得到$\{\overline{f}_n^{-v}\}$;

2)在v上测试$\{\overline{f}_n^{-v}\}$中的每一个模型;

3)对v中每一个样本的预测(也就是新的输入空间)和对应的输出组合在一起,作为新的样本加入T^1当中去.

(3)返回T^1.

由此可见,Stacking就是根据T^1中的偏置信息来进行二次建模,也称为元学习,它把F的元素组合起来,在F中找出最接近$f(X)$的函数(模型).可以看出,Stacking可以采取递进的方式多次建模来实现对分类信息的集成,具有较强的可扩展性.

1.2 T^1空间的数据表示

Wolpert给出的T^1空间是将底层分类器对训练数据的所属类作出预测,并将该预测结果同数据的真实分类一起组合成为新的训练数据.文献[4]将T^0中的一部分数据划分出来作为验证集(Validate Set),不参与训练底层分类器,专门用作底层分类器的预测样本,以便能够有效地避免底层分类器对数据的"过分拟合".文献[4]证明了用样本属于某个类的概率$p_0,p_1,\cdots,p_n$(n表示类标签的个数,$\sum p_i=1$)作为T^1空间的特征数据,训练效果要优于以预测类作为T^1空间的方式.T^1空间如表1和表2所示.

表1 训练集

Attributes			Class
x_{11}	$\cdots$	x_{1m}	$C1$
x_{21}	$\cdots$	x_{2m}	$C2$
$\cdots$	$\cdots$	$\cdots$	$\cdots$
x_{n1}	$\cdots$	x_{nm}	$C1$

表2 T^1空间的学习信息

Attributes				Class
$C1$	$C2$	$\cdots$	Cn	
T	T	$\cdots$	T	$C1$
F	T	$\cdots$	F	$C2$
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
T	T	$\cdots$	F	$C1$

1.3 Stacking的缺陷

尽管能够有效地对底层分类器进行二次建模,但是Stacking仍不可避免地存在以下缺陷:

(1)建模结果不易理解.由于T^1空间表示的是有关底层分类器的特征数据,因此新的模型描述的是各个底层分类器与正确分类之间的关系;而T^0空间的建模结果是原始数据与正确分类之间的关系.显然,后一种关系更容易被用户理解.

(2)T^1空间上的特征信息的规模不易控制.当底层分类器的个数增大并且问题域中包含的类别个数较大时,T^1空间的规模将会大大增加,继而使得新模型的建模时间增加.

2 降低T^1空间的属性规模

如果底层分类器的数目很多,势必造成T^1空间的规模增大.根据文献[5]得到的T^1空间可有$n_l\times n_c$个属性,其中n_l表示问题域中的类个数,n_c表示底层分类器的个数.这样就会造成元层训练的时间开销增大.一种可行的方法[6]是将元层训练集的属性由原来的各个底层分类器的预测类改成问题域中所含的类标签。这样,元层训练集的属性个数就与问题域的类别个数相等,而与底层分类器的数目无关.在这种情况下,可以用底层分类器对数据样本所属类的后验概率作为元训练集的属性值.概率向量(p_{k0},$p_{k1},\cdots,p_{kn}$)表示的就是底层分类器k对某个样本输出类$c_0,c_1,\cdots,c_n$的后验概率.由于M个底层分类器会产生k个概率向量,因此,样本X属于类标签i的概率p_i可以采用平均法表示成:

$$p_i=(\sum_{k=0}^{M}p_{ki})/M,i=1,2,\cdots,n.$$

用该方法生成的元层训练样本,可以大大减小二次建模的时间开销,文献[6]在使用UCI机器学习数据库中最大的4个数据集进行实验的结果验证了这一点.由于采用平均概率,使得T^1空间特征信息的部分丧失,导致新模型的分类准确率要低于标准Stacking的准确率,但从时间开销的角度来看,仍有相当有吸引力.

3 加权平均的后验概率方法

对后验概率进行平均的方法没有考虑到各个底层分类器分类性能的差异,对所有分类器的输出概率采取简单的平均化.而实际上,某些分类器对一些类的识别能力可能会强于其它分类器.因此,参考加权投票的规则,用分类器的分类准确率f_i作为权重值进行加权后再求平均值,准确率越大,赋与分类器的权重值也就越大.这样可以让那些具有较强分类性能的分类器发挥其优势,使得对T^1空间数据的训练能够更加重视分类准确高的分类器对样本的预测.为此加权的平均法可表示为:

$$p_i = (\sum_{k=0}^{M} p_{ki} \times f_i) / \sum f_i, i = 1,2,\cdots,n.$$

4 实验与分析

选用 UCI 数据库中的 2 个数据集 adult 和 spambase 进行测试. 首先从这 2 个数据集抽出 10% 的数据作为验证集样本,余下的数据分别划分成 5 个或 10 个子集(注:划分成 5 个子集与划分成 10 个子集时所对应的验证样本不相同). 底层分类器和元层分类器的训练使用 C4.5 决策树算法. 本文用叠加法、平均后验概率和加权平均后验概率 3 种方法得到元层训练样本,然后分别进行试验. 各个分类器的准确率 f 用 10 次十折交叉验证的平均值来表示. 表 3 给出的是 adult 和 spambase 5 个子集用 C4.5 构造分类器的分类准确率,表 4 给出的是 adult 和 spambase 10 个子集用 C4.5 构造分类器的分类准确率.

表 3 adult 和 spambase 的分类准确率

数据集	分类准确率	数据集	分类准确率
adult0	83.06±0.61	spambase0	89.66±0.64
adult1	82.34±0.4	spambase1	90.27±0.61
adult2	83.9±0.62	spambase2	88.85±0.36
adult3	83.86±0.43	spambase3	90.08±0.48
adult4	81.67±0.67	spambase4	88.61±0.48

表 4 adult 和 spambase 的分类准确率

数据集	分类准确率	数据集	分类准确率
adult0	80.42±0.89	spambase0	86.67±0.83
adult1	83.65±0.67	spambase1	90.73±0.7
adult2	81.87±0.44	spambase2	86.09±1.42
adult3	82.28±0.9	spambase3	90.11±0.96
adult4	82.83±1.07	spambase4	87.16±0.8
adult5	83.03±0.49	spambase5	86.44±0.92
adult6	81.42±0.67	spambase6	85.9±0.82
adult7	78.62±0.79	spambase7	87.9±0.86
adult8	80±1.27	spambase8	86.1±1.1
adult9	80.72±0.89	spambase9	89.09±0.78

用叠加法、平均后验概率和加权平均后验概率 3 种方法分别进行试验后得到的试验结果如表 5 所示.

表 5 3 种方法的试验结果

数据集	子集数	类标签数	Stacking	平均后验概率	加权平均后验概率
adult	5	2	83.98±0.57	81.19±0.16	81.51±0.55
	10	2	84.07±0.66	75.37±0.61	75.99±0.5
spambase	5	2	90.47±0.52	83.94±0.61	82.36±0.6
	10	2	93.82±0.39	72.56±0.84	74.33±1.18

从表 5 中可以看到,利用后验概率的平均值作为 T^1 空间特征,以未知样本的预测作为底层模型输出特征的叠加法,通过元学习训练后得到的分类准确率都高于在单个分类器上测试得到分类准确率,说明基于 Stacking 框架的学习确实有助于提高分类准确率,而用平均后验概率和加权的平均后验概率作为元训练样本,学习效果则要差于叠加法. 但是,平均后验概率的优点在于它能够使元层训练样本的规模不受底层分类器个数的限制,元层样本的属性个数始终与问题域中的类个数保持一致. 此外,对后验概率进行加权平均确实可以使得在 T^1 空间的训练更加重视分类准确高的分类器对样本的预测,从 adult-5、adult-10 和 spambase-10 的测试结果可以看到:用加权平均后验概率方法得到的结果略好于平均后验概率. 然而对 spambase-5,加权平均没有相应地提高准确率,原因是以准确率作为权重的加权平均方式产生了副作用,即对于某个待分类样本而言,即使是具有高分类准确率的分类器也有可能做出错误的预测,而如果以该分类器的准确率作为权值的话,则将会使错误的影响加大,使整个加权平均值降低. 如果有多个高准确率的分类器对同一个样本都做出错误的预测,就有可能使分类样本输出该错误类的后验概率要大于输出正确类的概率,从而导致该样本属于某个类的不确定性大大增加,进而误导元层训练算法.

5 结束语

基于 Stacking 框架的学习能够有效地提高学习效果,但是在分类器个数增大时可能会导致元层训练数据规模增加. 通过改变元训练集的属性,将预测结果改为类标签的后验概率,能够使元层属性个数保持稳定. 利用底层分类器的分类性能作为权值对各个分类器输出的后验概率进行加权平均,分类准确率大多高于平均法,但也存在因为加权而使错误的作用进一步放大的现象.

参考文献:

1 史忠植. 知识发现. 北京:清华大学出版社,2002.

2 Freund Y. Schapire R E. A short introduction to boosting. Journal of Japanese Society for Artificial Intelligence, 1999,14(5):771~780.

3 Wolpert D. Stacked generalization. Neural Networks, 1992,5(2):241~260.

4 Chan P,Stolfo S. Meta-learning for multistrategy and parallel learning. Proceedings of the Second International Workshop on Multistategy Learning,1993.

5 Ting K M,Ian H Witten. Issues in Stacked Generalization. Journal of Artificial Intelligence Research, 1999, (10):271~289.

6 Tsoumakas G,Vlahavas I. Distributed data mining of large classifier ensembles. Proceedings of 2nd Hellenic Conference on AI,2002.

(责任编辑:黎贞崇)