机器学习中的数学(3)-模型组合(Model Combining)之Boosting与Gradient Boosting

版权声明：

前言：

本来上一章的结尾提到，准备写写线性分类的问题，文章都已经写得差不多了，但是突然听说最近Team准备做一套分布式的分类器，可能会使用Random Forest来做，下了几篇论文看了看，简单的random forest还比较容易弄懂，复杂一点的还会与boosting等算法结合（参见iccv09），对于boosting也不甚了解，所以临时抱佛脚的看了看。说起boosting，强哥之前实现过一套Gradient Boosting Decision Tree（GBDT)算法，正好参考一下。

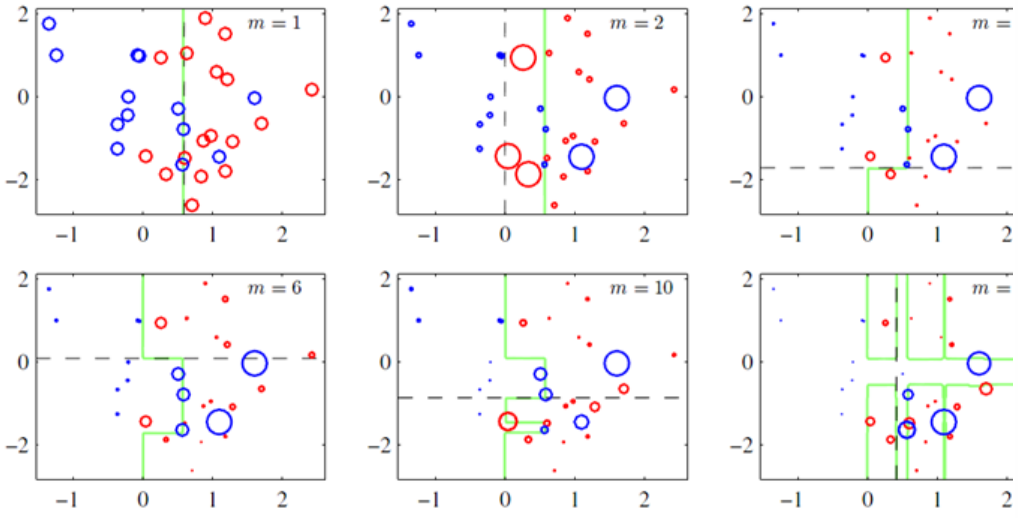
最近看的一些论文中发现了模型组合的好处，比如GBDT或者rf，都是将简单的模型组合起来，效果比单个更复杂的模型好。组合的方式很多，随机化（比如random forest），Boosting（比如GBDT）都是其中典型的方法，今天主要谈谈Gradient Boosting方法（这个与传统的Boosting还有一些不同）的一些数学基础，有了这个数学基础，上面的应用可以看Freidman的Gradient Boosting Machine。

本文要求读者学过基本的大学数学，另外对分类、回归等基本的机器学习概念了解。

本文主要参考资料是prml与Gradient Boosting Machine。

Boosting方法：

Boosting这其实思想相当的简单，大概是，对一份数据，建立M个模型（比如分类），一般这种模型比较简单，称为弱分类器(weak learner)每次分类都将上一次分错的数据权重提高一点再进行分类，这样最终得到的分类器在测试数据与训练数据上都可以得到比较好的成绩。



上图（图片来自prml p660）就是一个Boosting的过程，绿色的线表示目前取得的模型（模型是由前m次得到的模型合并得到的），虚线表示当前这次模型。每次分类的时候，会更关注分错的数据，上图中，红色和蓝色的点就是数据，点越大表示权重越高，看看右下角的图片，当m=150的时候，获取的模型已经几乎能够将红色和蓝色的点区分开了。

Boosting可以用下面的公式来表示：

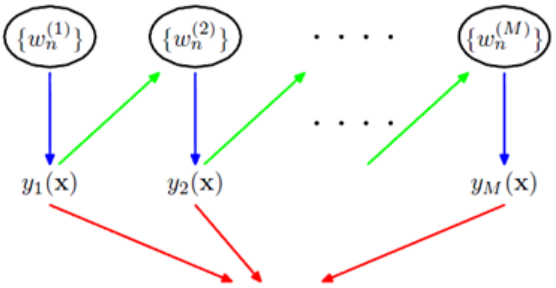
导航
博客园
首页
联系
订阅XML
管理

我的标签
机器学习(7)
搜索引擎(7)
Lucene(6)
machine learning(4)
pymining(3)
mathmatics(3)
人工智能(3)
数据挖掘(3)
PCA(2)
Model Combining(2)
更多

随笔分类
Hadoop(2)
Lucene C++重写心得(2)
Lucene JAVA心得(9)
安排，计划，总结(2)
分布式存储(2)
机器学习(7)
结构设计(1)
数学(7)

随笔档案
2012年2月 (1)
2011年9月 (1)
2011年8月 (1)
2011年5月 (5)
2011年3月 (1)
2011年2月 (1)
2011年1月 (3)
2010年12月 (2)
2010年11月 (1)
2010年10月 (1)
2010年9月 (2)
2010年8月 (1)
2010年1月 (12)
2009年12月 (2)
2009年11月 (4)

最新评论
1. Re:支持中文文本的数据挖掘平台开源项目PyMining发布
博主您好，为什么我运行naive_ayes_train_test.py 就要有IOError: [Errno 2] No such file or directory: u'mining/term.....
--aviva_159
2. Re:机器学习中的数学(5)-强大的矩阵奇异值分解(SVD)及其应用
很好！！谢谢



$$Y_M(x) = \text{sign}\left(\sum_m^M \alpha_m y_m(x)\right)$$

训练集中一共有n个点，我们可以为里面的每一个点赋上一个权重Wi(0 <= i < n)，表示这个点的重要程度，通过依次训练模型的过程，我们对点的权重进行修正，如果分类正确了，权重降低，如果分类错了，则权重提高，初始的时候，权重都是一样的。上图中绿色的线就是表示依次训练模型，可以想象得到，程序越往后执行，训练出的模型就越会在意那些容易分错（权重高）的点。当全部的程序执行完后，会得到M个模型，分别对应上图的y1(x)…yM(x)，通过加权的方式组合成一个最终的模型YM(x)。

我觉得Boosting更像是一个人学习的过程，开始学一样东西的时候，会去做一些习题，但是常常连一些简单的题目都会弄错，但是越到后面，简单的题目已经难不倒他了，就会去做更复杂的题目，等到他做了很多的题目后，不管是难题还是简单的题都可以解决掉了。

Gradient Boosting方法：

其实Boosting更像是一种思想，Gradient Boosting是一种Boosting的方法，它主要的思想是，每一次建立模型是在之前建立模型损失函数的梯度下降方向。这句话有一点拗口，损失函数(loss function)描述的是模型的不靠谱程度，损失函数越大，则说明模型越容易出错（其实这里有一个方差、偏差均衡的问题，但是这里就假设损失函数越大，模型越容易出错）。如果我们的模型能够让损失函数持续的下降，则说明我们的模型在不停的改进，而最好的方式就是让损失函数在其梯度（Gradient)的方向上下降。

下面的内容就是用数学的方式来描述Gradient Boosting，数学上不算太复杂，只要潜下心来看就能看懂：）

可加的参数的梯度表示：

假设我们的模型能够用下面的函数来表示，P表示参数，可能有多个参数组成，P = {p0,p1,p2….}，F(x;P)表示以P为参数的x的函数，也就是我们的预测函数。我们的模型是由多个模型加起来的，β表示每个模型的权重，α表示模型里面的参数。为了优化F，我们就可以优化{β,α}也就是P。

$$F(x;P) = F(x;\{\beta_m,\alpha_m\}_1^M) = \sum_{m=1}^M \beta_m h(x;\alpha_m)$$

我们还是用P来表示模型的参数，可以得到，Φ(P)表示P的likelihood函数，也就是模型F(x;P)的loss函数，Φ(P)=…后面的一块看起来很复杂，只要理解成是一个损失函数就行了，不要被吓跑了。

$$P^* = \arg\min(\Phi(P))$$
$$\Phi(P) = E_{y,x}L(y,F(x;P))$$

既然模型(F(x;P))是可加的，对于参数P，我们也可以得到下面的式子：

$$P^* = \sum_{m=0}^M p_m$$

这样优化P的过程，就可以是一个梯度下降的过程了，假设当前已经得到了m-1个模型，想要得到第m个模型的时候，我们首先对前m-1个模型求梯度。得到最快下降的方向，gm就是最快下降的方向。

$$g_m = \{g_{jm}\} = \{[\frac{\partial\Phi(P)}{\partial p_j}]_{p=p_{m-1}}\}$$

这里有一个很重要的假设，对于求出的前**m-1**个模型，我们认为是已知的了，不要去改变它，而我们的目标是放在之后的模型建立上。就像做事情的时候，之前做错的事就没有后悔药吃了，只有努力在之后的事情上别犯错：

$$P_{m-1} = \sum_{i=0}^{m-1} p_i$$

我们得到的新的模型就是，它就在P似然函数的梯度方向。ρ是在梯度方向上下降的距离。

$$p_m = -\rho_m g_m$$

我们最终可以通过优化下面的式子来得到最优的ρ：

$$\rho_m = \arg\min \Phi(P_{m-1} - \rho_m g_m)$$

可加的函数的梯度表示：

上面通过参数P的可加性，得到了参数P的似然函数的梯度下降的方法。我们可以将参数P的可加性推广到函数空间，我们可以得到下面的函数，此处的fi(x)类似于上面的h(x;a)，因为作者的文献中这样使用，我这里就用作者的表达方法：

$$F_{m-1}(x) = \sum_{i=0}^{m-1} f_i(x) \quad \text{类似于} \quad P_{m-1} = \sum_{i=0}^{m-1} p_i$$

同样，我们可以得到函数F(x)的梯度下降方向g(x)

$$g_m(x) = E_y\left[\frac{\partial L(y,F(x)}{F(x)} \mid x\right]_{F(x)=F_{m-1}(x)} \quad \text{类似于} \quad g_m = \{g_{jm}\} = \{[\frac{\partial \Phi(P)}{\partial p_j}]_{p=p_{m-1}}\}$$

最终可以得到第m个模型fm(x)的表达式：

$$f_m(x) = -\rho_m g_m(x)$$

通用的**Gradient Descent Boosting**的框架：

下面我将推导一下Gradient Descent方法的通用形式，之前讨论过的：

$$F(x;P) = \sum_{m=1}^{M} \beta_m h(x;\alpha_m)$$

对于模型的参数{β,a}，我们可以用下面的式子来进行表示，这个式子的意思是，对于N个样本点(xi,yi)计算其在模型F(x;a,β)下的损失函数，最优的{a,β}就是能够使得这个损失函数最小的{a,β}。 $\{\beta_m,\alpha_m\}_1^M$ 表示两个m维的参数：

$$\{\beta_m,\alpha_m\}_1^M = \arg\min \sum_{i=1}^{N} L(y_i, \sum_{m=1}^{M} \beta_m h(x_i;\alpha_m))$$

写成梯度下降的方式就是下面的形式，也就是我们将要得到的模型fm(x)的参数{αm,βm}能够使得fm的方向是之前得到的模型Fm-1(x)的损失函数下降最快的方向：

$$\beta_m,\alpha_m = \arg\min \sum_{i=1}^{N} L(y_i, F_{m-1}(x_i) + \beta h(x_i;\alpha))$$

对于每一个数据点xi都可以得到一个gm(xi)，最终我们可以得到一个完整梯度下降方向

$$\overrightarrow{g_m} = \{-g_m(x_i)\}_1^N$$

$$-g_m(x_i) = -\left[\frac{\partial L(y_i,F(x_i)}{\partial F(x_i)}\right]_{F(x)=F_{m-1}(x)}$$

为了使得fm(x)能够在gm(x)的方向上，我们可以优化下面的式子得到，可以使用最小二乘法：

$$\alpha_m = \arg\min \sum_{i=1}^{N} (-g_m(x_i) - \beta h(x;\alpha))^2$$

得到了a的基础上，然后可以得到βm。

$$\beta_m = \arg\min \sum_{i=1}^{N} L(y_i, F_{m-1}(x_i) + \beta h(x_i;\alpha_m))$$

最终合并到模型中：

$$F_m(x) = F_{m-1}(x) + \rho_m h(x;\alpha_m)$$

算法的流程图如下

Algorithm 1: Gradient_Boost

$F_0(\mathbf{x}) = \arg\min_\rho \sum_{i=1}^{N} L(y_i, \rho)$

For $m = 1$ to M do:

$\tilde{y}_i = -\left[\frac{\partial L(y_i, F(\mathbf{x}_i))}{\partial F(\mathbf{x}_i)}\right]_{F(\mathbf{x})=F_{m-1}(\mathbf{x})}, \; i = 1, N$

$\mathbf{a}_m = \arg\min_{\mathbf{a},\beta} \sum_{i=1}^{N} [\tilde{y}_i - \beta h(\mathbf{x}_i; \mathbf{a})]^2$

$\rho_m = \arg\min_\rho \sum_{i=1}^{N} L(y_i, F_{m-1}(\mathbf{x}_i) + \rho h(\mathbf{x}_i; \mathbf{a}_m))$

$F_m(\mathbf{x}) = F_{m-1}(\mathbf{x}) + \rho_m h(\mathbf{x}; \mathbf{a}_m)$

endFor

end Algorithm

之后，作者还说了这个算法在其他的地方的推广，其中，Multi-class logistic regression and classification就是GBDT的一种实现，可以看看，流程图跟上面的算法类似的。这里不打算继续写下去，再写下去就成论文翻译了，请参考文章：Greedy function Approximation – A Gradient Boosting Machine，作者Freidman。

总结：

本文主要谈了谈Boosting与Gradient Boosting的方法，Boosting主要是一种思想，表示“知错就改”。而Gradient Boosting是在这个思想下的一种函数（也可以说是模型）的优化的方法，首先将函数分解为可加的形式（其实所有的函数都是可加的，只是是否好放在这个框架中，以及最终的效果如何）。然后进行m次迭代，通过使得损失函数在梯度方向上减少，最终得到一个优秀的模型。值得一提的是，每次模型在梯度方向上的减少的部分，可以认为是一个“小”的或者“弱”的模型，最终我们会通过加权(也就是每次在梯度方向上下降的距离）的方式将这些“弱”的模型合并起来，形成一个更好的模型。

有了这个Gradient Descent这个基础，还可以做很多的事情。也在机器学习的道路上更进一步了：）

分类: 机器学习, 数学

标签: gradient descent, 机器学习, Boosting, Gradient, GBDT, Model Combining, AdaBoost, 模型组合

LeftNotEasy
关注 - 16
粉丝 - 787
+加关注

« 上一篇：机器学习中的数学(2)-线性回归，偏差、方差权衡
» 下一篇：机器学习中的数学(4)-线性判别分析（LDA），主成分分析(PCA)

posted on 2011-01-02 21:48 LeftNotEasy 阅读(50946) 评论(12) 编辑 收藏

评论

#1楼 2011-03-05 10:02 开飞机的舒克

"如果分类正确了，权重提高，如果分类错了，则权重降低"是不是笔误写反了？

支持(1) 反对(0)

#2楼[楼主] 2011-03-05 10:19 LeftNotEasy

@开飞机的舒克
呵呵，谢谢指出这个笔误，立刻改过：）

支持(0) 反对(0)

#3楼 2011-09-18 11:28 新彩

刚开始接触机器学习，写的很好，很受启发，谢谢分享。

支持(0) 反对(0)

#4楼 2012-06-11 16:48 bgi_dkill

最新准备实现RF和GBT，看了收获很大，谢谢！

支持(0) 反对(0)

#5楼 2012-06-27 13:37 Aries的小窝

老大，强哥的博客连接访问不了啊，需要翻墙么。

支持(0) 反对(0)

#6楼 2012-07-06 10:04 FanTaSy_HJ

意思有点懂。。但是没有具体事例，难理解啊
尤其是具体问题中，α β怎么定义？

支持(0) 反对(0)

#7楼 2012-11-01 10:05 小熊@实验室

博主 我要转载你的文章，在此给你打个招呼哈~我会注明出处的~

支持(0) 反对(0)

#8楼 2012-11-09 11:06 可健康了

感谢博主的文章，让我学到不少知识。我是做模式识别方向的，用过一些机器学习的算法，想和楼主探讨学习一下，方便加个Q吗42604952，

支持(0) 反对(0)

#9楼 2013-08-02 23:01 孤独剑客zzy

之前研究过boosting和Adaboost，可是没有写出来，到时候跟楼主探讨

支持(0) 反对(0)

#10楼 2013-11-10 16:09 zxlei

应用："我觉得Boosting更像是一个人学习的过程，开始学一样东西的时候，会去做一些习题，但是常常连一些简单的题目都会弄错，但是越到后面，简单的题目已经难不倒他了，就会去做更复杂的题目，等到他做了很多的题目后，不管是难题还是简单的题都可以解决掉了。"
再难的题，只要是误差或者噪音，那么adaboost的效果就会越差,这是会放大误差拟合导致模型的泛化能力不好的

支持(0) 反对(0)

#11楼 2014-02-21 11:14 从此醉shero

楼主你好，非常谢谢你写的文章，让我豁然开朗，不过还有一点小疑问，利用最小二乘法不就可以求出α和β了么？为什么的第5步还利用损失函数来求β啊？

支持(0) 反对(0)

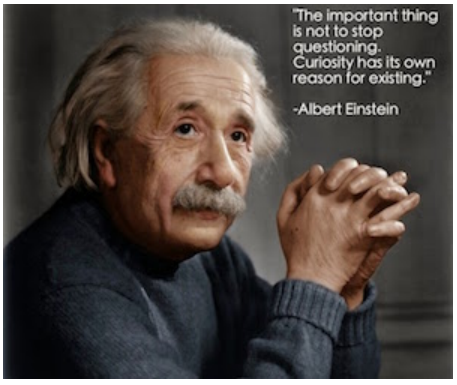
#12楼 2015-12-21 09:24 hi，daring

楼主，你好，请问你的那些图片使用什么工具得到的呢？

支持(0) 反对(0)

刷新评论 刷新页面 返回顶部

注册用户登录后才能发表评论，请 登录 或 注册，访问网站首页。



最新IT新闻：
· 这位艺术家把整个宇宙画入了一幅图中
· 社交数据太庞大 Facebook结盟松下研发1TB光盘
· “高分四号”突破遥感卫星技术五个“首次”
· 硅谷无人驾驶车专利不如汽车厂商多：丰田居首
· Twitter CEO解释取消推文140字符限制的原因
» 更多新闻...

最新知识库文章：
· Docker简明教程
· Git协作流程
· 企业计算的终结
· 软件开发的核心
· Linux概念架构的理解
» 更多知识库文章...