中山大学数据科学与计算机学院

移动信息工程专业-人工智能

本科生项目报告

（2017-2018学年秋季学期）

课程名称：**Artificial Intelligence**

教学班级	**1501**	专业（方向）	移动互联网
组号	3	组名	Nidhogg
学号	15352015	姓名	曹广杰

一、Project最终结果展示

1. 最终结果

binary	multiple	regression	Rank
21	18	14	14
0.910961816481	0.628299978298	54.974130157	

2. 组内分工

组长蔡林航（15352007）：二元分类；

蔡政（15352014）：回归模型；

曹广杰（15352015）：多元分类；

3. 个人工作

//以分点简述或图表的形式，概述自己在负责的部分做了哪些工作，例如什么时间试了哪些算法，什么时间调整某个算法，什么时间写报告展示。

周次	事务
14-15	梳理数据集、确定算法为神经网络
15-16	调整神经网络的结构
16-17	调节参数（节点数量等）
17-18	撰写报告

二、　工作流程

算法简介

该章节中将会介绍笔者在本次实验中所使用的模型以及针对模型的改进方法。

笔者在这一次project中处理情感识别分析，即多元分类。使用了神经网络模型的架构作为拟合本次模型的模型基础。原因如下：

1. 情感是有强弱之分的，神经网络更加符合数据集表达的信息性质，可解释性更强；
2. 神经网络是LR的加强版，通过隐藏层节点的拟合可以容纳更加复杂的模型；

神经网络架构

笔者使用了最简单的神经网络模型BPNN，具体模型的参数构造如下：

1. 输入节点929个，激活函数为sigmoid；
2. 隐藏层只有一层，节点数380个，激活函数为sigmoid；
3. 输出节点一个，输出连续数据，激活函数为$\boldsymbol{f(x)=x}$；
4. 所有节点的偏置项都初始化为2.2；
5. 对输出节点的连续值分类的阈值为：
    - 低于0.411的则为LOW；
    - 高于0.619的则为HIG；
    - 介于两者之间的为MID；

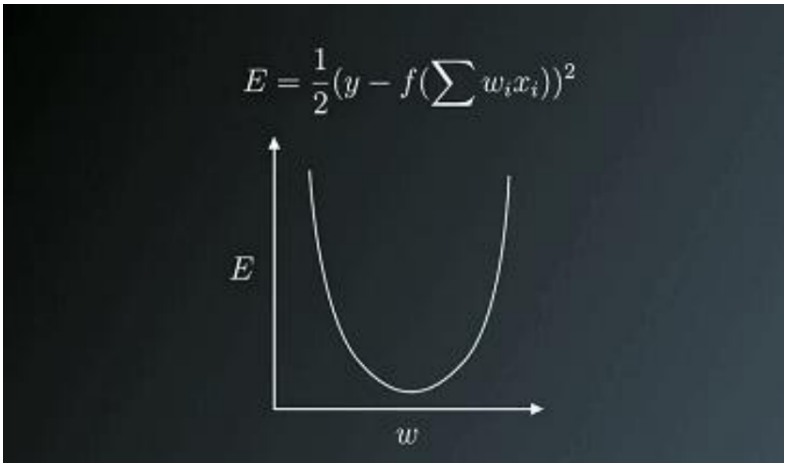
改进神经网络

笔者基于BPNN模型，但是对BPNN模型进行了一些改进。

改进一：根据斜率变化的动态学习率

这个改进也还是基于动态学习率的思想，但是不同的是笔者直接将步长选为梯度。首先需要基于神经网络的几个前提：

- MSE，即损失函数是权值的函数，其自变量为$\boldsymbol{w_i}$；



梯度下降法

- 因此，权值的迭代更新公式为：

$$\because w_i = w_i + \Delta w_i$$
$$\Delta w_i \propto -\frac{\partial E}{\partial w_i}$$
$$\therefore w_i = w_i - \eta\frac{\partial E}{\partial w_i}$$

其中η为步长，步长经常被设置为常数值。此时用梯度下降法，就是期望可以使得w_i可以到达使损失函数最小的位置。但是

1. 如果步长值太小，则收敛速度过慢；
2. 如果步长过大，则非常容易跳过最优解的位置；

因此，笔者对于步长设置的改进就是直接将步长设置为梯度。因为，随着迭代的结果趋近于最优解，梯度会渐渐变小。所以，**在靠近最优解的时候，迭代的程度会逐渐降低，以致微不可察**。

这种设计的优势：

1. 在迭代更新在可接受的范围内，模型会逐渐趋于更优，不会跳过距离当前最近的最优解；
2. 收敛速度快。在梯度大的时候，收敛速度非常快，因为步长就是梯度。

当然也有不足：

1. 可能会陷入局部最优解。这种设计会快速收敛到最近的极小值但是不具备跳出局部最优解的能力，它只会向最近的优值收敛；

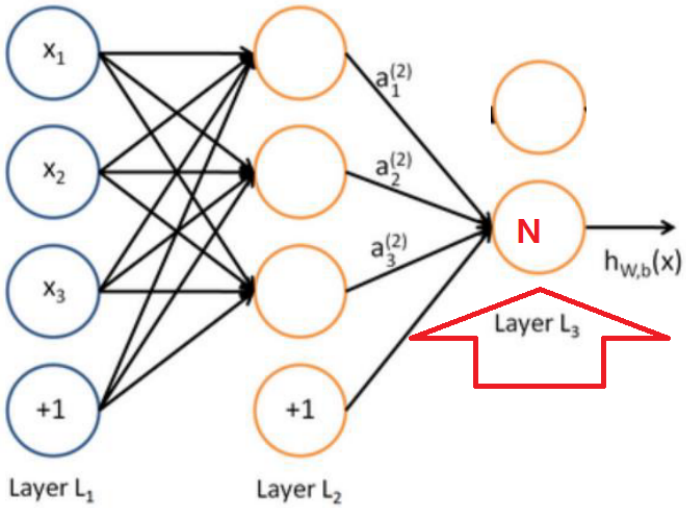
   解决方案：尽量使用不会产生局部最优解的冲激函数，比如sigmoid。

2. 可能会产生梯度爆炸。在梯度过大的时候，第一次迭代可能会使得w_i跳到与当前位置关于最优解对称的另一个梯度非常大的位置，使得梯度不断放大。

   解决方案：合理的初始化，避免梯度过大的情况发生。或者将步长设置为梯度的某一个函数——而不是直接使用梯度信息；

改进二：神经元轴突的阶梯型学习策略

这个策略的改进是针对每一个神经元进行的操作。假设当前神经网络模型中有一个神经元N：

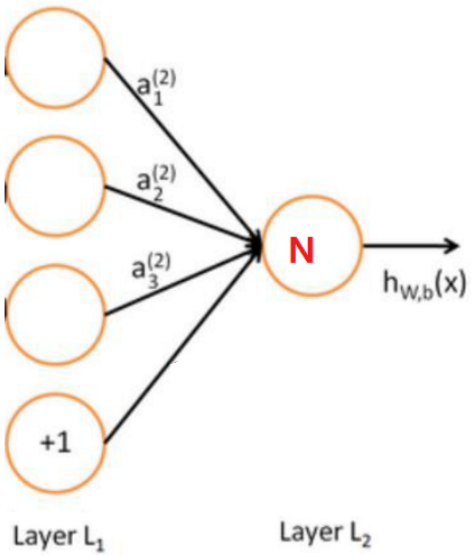


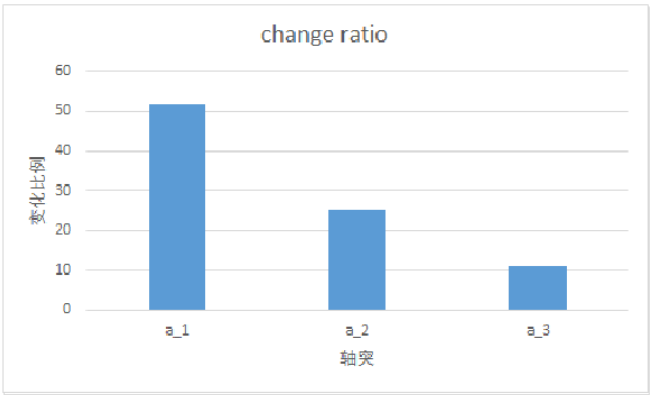
向上箭头所指的神经元就是被笔者选定的神经元了。对于神经元及其所在的神经层有如下的性质：

1. 该层的神经元从上一层接收到的数据都相同；
2. 期望该层的神经元有不同的输出；
3. 期望每一个神经元的输出可以既符合数学模型的整体趋势，又具备针对细节调整的能力；

笔者针对以上三点期望对该神经元的权值更新设计了该神经元N的权值更新策略。即，神经元的突触从上到下依次记为$a_1, a_2, a_3, \ldots, a_n$，每一个轴突都对应着一个权值$w_i$。

在这个策略中，每一个轴突权值更新都不同，而且是自下而上逐渐增强的——也就是说，对于神经元N来说，w_0的变化最为剧烈，相比之下，w_n就只有非常小的变化，但是也还是有变化的。





这种策略的优势在于：

1. 相比于刻意初始化神经元的权值，该策略中允许神经元初始化相同，便于操作；
2. 相比于随机初始化神经元的权值，该策略构造的模型表现更加稳定；
3. 变化剧烈的轴突适应数学模型趋势，变化小的轴突负责模型微调，二者兼顾；
4. 将神经网络各层之间的数据传递限定在一个范围之内，在一定程度上避免了梯度的爆炸或者消失（这是瞎说的）；

当然，这种策略也有不足：

1. 考虑某一层神经网络A的节点数量非常多；

2. 下一层的一个神经元`B`需要有与之相同数量的轴突，即相同数量的权值；
3. 为了保证`B`各个权值之间的梯度，变化最剧烈的权值就需要大幅度变化；

这种大幅度变化就可能造成梯度爆炸的情况，但是如果过于控制权值的这种变化，就会使得对于神经元`B`来说，输入节点的信息没有拉开差距，近乎于大量完全相同的节点的输入。这将成为该模型的一个瓶颈。

解决方案：避免某一层的节点数量过多，可以通过增加隐藏层解决该问题。

主系表解构文本

本次project的主要任务是根据文本判断文本所表达的情感程度。因为文本处理是人工智能领域中非常重要的一个组成部分，所以笔者专门为该部分设计了算法。该算法是依据英语与汉语中共同的语法结构进行关键词的筛选，不同于统计学的筛选方式，这种筛选的时间复杂度与空间复杂度都远低于根据完全词频与文本标签的相关性进行筛选的方式。

笔者在分析处理的对象的时候是将文本信息根据英语语法的规律提取出一些关键词——这里笔者期望只提取出形容词。具体操作：

1. 根据读入文件的分隔符将段落划分成句；
2. 根据句子结构针对主系表结构将表语结构提取出来；
3. 使用词袋模型管理表语结构，并对词袋模型进行筛选；
4. 对每一个句子根据词袋模型将词汇制成TF矩阵；
5. 将TF矩阵作为神经网络的输入节点信息，进行神经网络的训练；

使用主系表的依据：

1. 主系表结构是在人类日常语言中直接抒发情感的部分，其中的形容词词频最高而且词汇种类丰富；
2. 人类的语言是在应用中不断发展的，就汉语而言，从文言文到白话文经历了很多年的演变，这种演变的趋势是日常使用的方便，而这种方便的最明显表现就是体系化与系统化。（*注：文言文是书写型的语言，并不利于日常的说话交流*）
3. 使用统计学方法计算词汇相关性的方法最终筛选出来的词汇更多的也是属于表语体系，或者更多的也是属于形容词的一部分，当然，也可能有副词动词之类的。

这种数据处理的优势：

1. 减少空间复杂度。使用统计学的计算方法可以找到与标签值相关性最高的词汇，但是需要列出一个巨大的矩阵执行大量的浮点数运算，而这种数据集采集的方法几乎是线性的计算复杂度。截止到目前，笔者的算法运算占用内存大豆可以控制在1G以内，完全可以控制在1.1G以内；
2. 减少时间复杂度。由于使用对于文本处理的时候针对性比较强，所以在文本处理中做复杂的操作就是遍历与计数，省略了大量的浮点数运算，节约了计算资源。截止到目前，在不使用adaboost强化训练的情况下，从读取文件到迭代55次完成，总耗时可以控制在5min以内，准确率51%以上。

另外，该算法还有一些限制：

1. 在语言中隐晦的表达方式计算机尚不能识别——因为理解这些语言需要生活经验，而这种经验化的信息难于被数字化，而且当前的计算机也没有储存这种信息。
2. 为了保证筛选出的形容词对文本的分类有较大程度的影响，需要保证该词汇的词频高于某一个阈值；
3. 该阈值难以被确定，即笔者难以衡量多少词才可以完全用于分辨文本的情感信息。

动词向形容词的划分

在一个句子中，除了有形容词可以用于表征发语人的情感特征外，动词也可以表征情感属性，有时候一个句子中的动词——“爱”或者“恨”，就可以表达发语人的情感信息。笔者在本次实验中只选择了一个动词作为修改的目标——“love”，具体的处理方法是在一个句子中遍历寻找该词，并将找到的该词替换为几个表达积极情感的形容词，

如：“friendly”、“good”、“nice”、“delicious”。

调参过程

神经网络的参数调节比较复杂，因为神经网络的需要考虑的信息比较多。在本次实验中，笔者主要调节的参数有如下几种：

1. 输入节点数量，或者采词的数量；
2. 隐藏层节点的数量；
3. 每一个神经元偏置项的初始化值；
4. 包含adaboost的迭代次数；
5. 对输出节点连续值的分类阈值；

输入节点数量

目前输入节点的数量为929，在此之前输入节点数量都尽量控制在600以内。实验数据表明，增加输入节点的数量可以在模型承受的范围内大幅度提升预测值的准确率：

输入节点数量	可以到达的最高准确率
291	47%+
600	50%+
700+	58.21%+
929	58.42%+

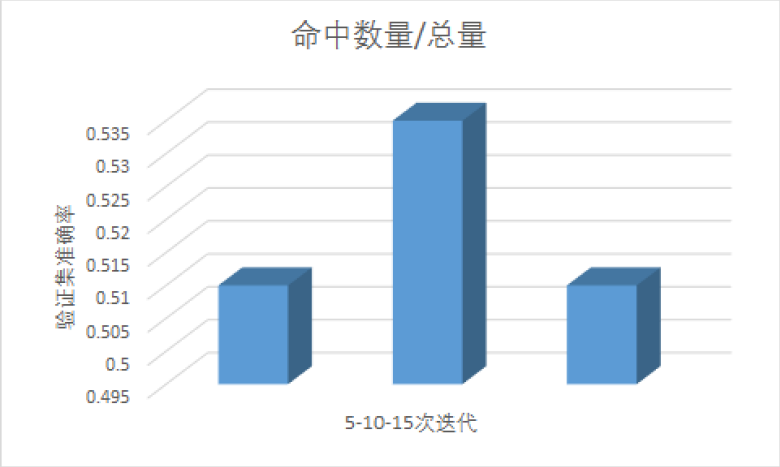
在此之后，输入节点的增加就不会造成什么影响了。至于输入节点1020多的测试，是因为在那次测试中笔者同时调节了其他的参数——输出的分类阈值，导致结果的下降。但是随着输入节点数量的上升，模型所能处理的信息的上限也逐渐显现出来。原因如下：

1. 使用BPNN模型，只有一个隐藏层，可以拟合的模型复杂度非常有限；
2. 使用“神经元轴突的阶梯型学习策略”（*详见算法简介/ 改进神经网络/ 改进二*），某一层的节点数过多会导致下一层的轴突学习梯度无法拉开，可以通过增加隐藏层数量解决。

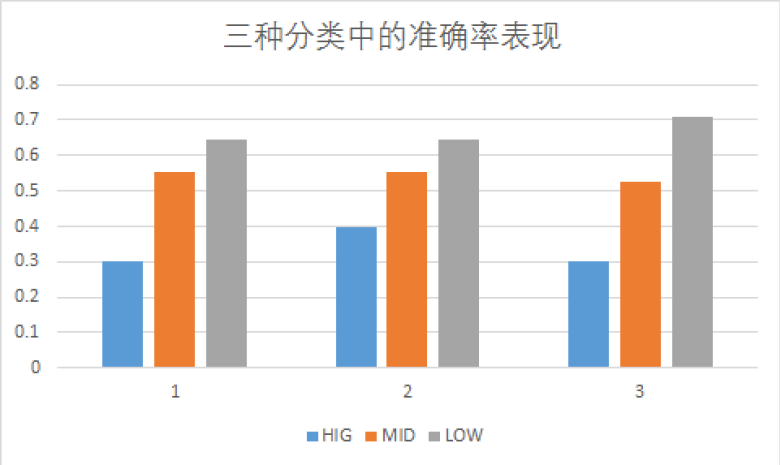
所以最后笔者取929作为输入节点的数量，这是笔者当前所使用的简单模型所能承受的较大的值。可以在保证传入的信息丰富的前提下不会耗费过量的计算资源。

隐藏层节点数量

笔者在调试中，保证输入节点数量为214，隐藏层节点为35，输出分类阈值以及其他指标都相同的条件下测试迭代次数对输出准确率的影响：



同时对于三种分类的准确率指标如下（自左向右分别是迭代5次、10次以及15次）：



笔者分析以上的测试结果得到以下结论：

1. 过多的迭代次数可能会导致整体的准确率下降。

   通过第二幅图猜测可能是由于在训练集中的LOW占较大的比例，所以在大量的迭代中导致预测结果中LOW占更大的比重。

2. 迭代次数在增加到一定程度之后，继续增加对于准确率的影响变得有限。

   从5次到10次的迭代增加会有较大的影响，但是再增加迭代次数在整体预测率上反而会下降，这其中当然有训练集分布的影响。但是，另一方面也说明，没有像之前那样明显的进步以抵消来自训练集分布的负面影响。

   对于这种情况笔者的解释是神经网络所能容纳的数学模型已经“饱和”，继续迭代下去的时候，神经网络在学习到新的规律的时候就会抹去旧的已经学习到的规律。

3. 迭代次数的增加，使得预测的数值向LOW和HIG两端分布。

   基于以上的结果以及猜测笔者增加隐藏节点为380，此后在test集上的准确率都为50%+。

输出节点连续值的分类阈值

由于神经网络的输出数据性质是连续的，但是在测试集上表现的时候，需要离散的数据进行准确率的计算，因此，笔者就需要知道在师兄的测试集中，HIG、MID与LOW大致占多大的比例，以实现笔者对于连续集的划分。在测试的时候的准确率算法如下：

已知测试集中共有A个HIG，B个MID以及C个LOW，提交上去的答案命中了a个HIG，b个MID以及c个LOW，则最后的输出准确率p为：

$$p=\frac{a/A+b/B+c/C}{3}$$

这种计算方式就使得测试数据集中的分布变得非常困难。笔者查询测试集中各个分类所占的比例操作如下：

1. 生成全部为LOW的数据S；
2. 通过阅读测试集文本，找到其中负面情感最为明确的10条评价；
3. 将S中的对应部分修改为HIG；

这种操作就使得上式中的a/A与b/B全都为0，而LOW中笔者又找出了绝对为LOW的10个数据特意预测为HIG，这样最后的输出结果就会小于33.333%，所以得知10条信息在LOW中所占的比例为多少——继而推算出LOW大致有多少，同理可以获得HIG的数据。

但是由于算力以及时间的关系，笔者还没有调节出合适的分类阈值，使得输出的分类可以较好地符合预期。不过调参过程总是较为枯燥，笔者并不想在这方面花费过多的时间。

引用

“文言的文字可读而听不懂；白话的文字即可读，又听得懂。”——《四十自述》胡适

笔者在实现本次实验的时候没有参照其他的论文或者博客，使用的方法都是根据测试的结果及模型的需求进行设计的。如果非要说有什么引用的话，在开始project的前一个月，笔者阅读过胡适的《四十自述》，对笔者后续实现的数据集处理造成了一些影响，以至于笔者后来实现了使用语法规则进行词汇筛选的数据集处理算法，真正的O(n)算法。因为胡适是致力于古文到白话文的改革，他对白话文的推进作出了很大的贡献，通过他的自述，笔者意识到人们日常使用的语言其实是有非常明显的语法规则的，而且这种规则并不是限于一个种族或者一种语言，这是在日常使用中一代一代总结改进出的规则，为了表意更为清晰。

因此对于本次实验中特征明显的数据集，笔者使用了语法中很小的一部分进行实现，以期完成较好的数据集处理效果。

数据集分析

1. 多元分类：词汇量较大，师兄给的数据集（.ss）文件中使用的是Unicode编码，需要使用代码调整，提取关键信息较为困难；
2. 二元分类：整体上与回归相似，需要处理非数字化的信息——type列。容易想到将字符串类型的这一列中的可能五种所有的取值转成数字0到4，方便进行数据处理。对于LR算法而言，测试集F1值可以从0.618提升到0.725，提升效果明显。；
3. 回归实验：在圣诞节区域的节假日信息十分混乱，规律难以找到。天气类型需要离散化处理；温度，连续数据，温度过低时离散处理；湿度与年份信息都需要离散化处理。

集成学习方法（Adaboost）

笔者在实验中使用了adaboost的思想进行改进。这也是笔者团队对于adaboost的使用。在介绍之前，首先需要知道adaboost的思想主要有两个：

1. 给判断失误的数据更大的权重，以实现强化训练；
2. 对多个弱分类器进行整合，实现弱分类器集成为强分类器；

由于笔者使用的是神经网络算法——并不是使用弱分类器，所以使用第二个思想实现神经网络的集成计算复杂度是非常高的，而且模型也更加复杂，不符合简单的设计理念。因此，笔者主要使用了第一个思想进行改进，即强化训练判断失误的信息。

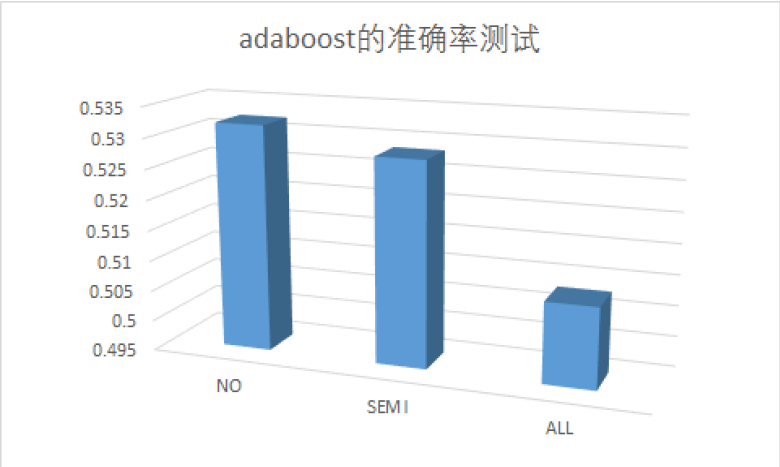
由于多元分类需要处理的数据量较大，将错误的判断数据与正确的判断数据按比例分配的话迭代一次的代价太高，这里笔者使用一个vector储存在一次遍历中的所有判断失误的数据，并在这一次遍历之后，将vector内部的上数据重新遍历一遍，以实现强化训练的效果。其实就是使用了一个比较极端的adaboost模型，在该模型中，错误的信息在强化迭代的时候权值被设置为1，而正确的参数权值被削弱为0 。

adaboost的表现

笔者使用adaboost生成测试集数据进行了3次测试（这三次测试的实验环境相同，参数一致），三次测试分别为：

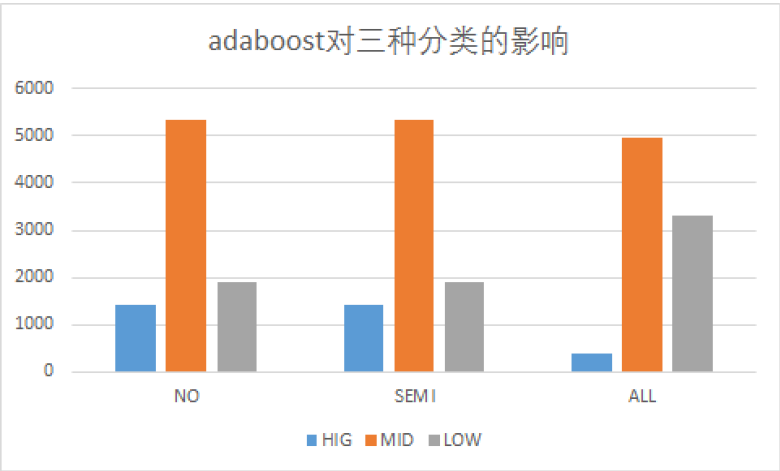
1. 完全不使用adaboost强化训练；
2. 前半部分使用adaboost强化训练；
3. 每一次迭代都使用adaboost强化训练；

效果如下：



可以看到，随着adaboost的使用，准确率逐渐下降。但是这并不能说明adaboost的效果就非常差，因为测试集的判断结果是与最后笔者的输出阈值有关的，但是笔者的输出阈值始终没有到达合理的位置，这种结果只能说明，当前的阈值对于adaboost强化训练之后的连续数据的分类与期望不符合而已。

所以，对adaboost的分析还是需要根据更为原始的数据：



这是上文中数据的详细分类结果，可以看到随着adaboost的使用范围逐渐扩大，预测的结果中LOW所占的比例直线上升，这已经非常明显地说明了adaboost会受到训练集中的分类比例的影响，即由于LOW的基数比较大，在强化迭代的过程中LOW也占据了较大的比例，这种迭代的结果就是使得整个预测的信息都有向下的趋势。

为了弥补这种趋势，笔者在每一次遍历中统计了判断失误的数据中各个分类的数目，控制adaboost的强化训练中，三种分类的数量相同。这种做法的结果就获得了当前的最优解58.21%+。

三、 课程总结

第一句话，人工智能还是很有意思的。记得我写的第一个函数就是main函数，输出的第一句话就是“Hello World”，当时还没有觉得有多神奇。当时的感觉就是C语言只不过是一种工具，是过一段时间就会被淘汰掉的工具。当时随着之后的学习，我渐渐意识到，这个“一段时间”，可能是2年、7年、10年很多年，而C语言到Java等一系列高级语言，走到现在，远远不再是一门语言这么简单——从算法到设计，表现了一种艺术。很多年以后，我在知乎上看到一个问题：

“为什么第一行代码是Hello World？”

——“因为那一刻，你的程序有了生命。”

我在这次实验中使用的模型正是神经网络，看着它一点点地学习到文本中的信息，真的是感觉到了程序的生命。而人工智能给了程序学习的能力，这将会为世界带来不同。这时候如果我说在代码中有一种情怀那是不切实际的，因为代码中有的远远不止这么简单，代码中所蕴含的何止是情怀，代码中蕴含的，根本就是未来。我知道编程这种东西将来会被淘汰的，但是我暗自揣度，那时候大概也已经是我们这一代百年之后了。师兄不复健壮，老师也不复聪慧，这些岁月和最基础的BPNN一样，是将来那个不需要编程的时代的基石。根本就不需要有人去记得或者去追忆，人类在这条长河中扮演的角色太微不足道了——但是有这么一群人，在那个时代，用他们的时间，创造过。

第二句话，现在实现的人工智能还是太简单了。尽管我什么都不知道但是我也可以这么断言，一方面这是受限于当前的计算机架构，另一方面身边的改变还没有受到其特别明显的影响。但是我相信人工智能是会大有可为的，最近查了一点关于Jim Keller的行踪——据说也去做人工智能硬件了？这种半仙的眼光总不会错，想必将来人工智能的硬件优化出来，新的智能就会同时具有人类的思维能力和计算机的计算能力，相比之下，人类的进化实在是过于缓慢了。

第三句话，人工智能的实验真是非常必要。据说软工是没有人工智能实验的啊？我觉得人工智能最精彩的部分正是实验的部分，中大的学生向来眼高手低（woc谁说的！？），在编程的时候有一些能力才是真正可以得到锻炼，同时在自己实现的时候，对于模型的修改以及思考才是这种课程真正有意义的地方，怎么说呢，我觉得知识相比之下，反而不值一提（我TM……难怪我绩点这么低）——当然，这种模式并不是对所有人都适合的，作为一种普适的可以给资质不是特别优秀的学子的课程，其深度是非常难以把握的。只是希望学分可以更高一些，能学到自己喜欢的课程自然是非常有趣啦，但是在最后的Project的时候又正是复习周，挑战性高但是学分少，就略有一点鸡肋的味道。

最后一句话当然是要感谢师兄和饶老师一学期的陪伴啊，祝师兄学业有成，祝老师工作顺利，生活幸福。（咋，没有人祝福我吗？我的QQ：1553118845）致敬！