有监督学习

张运吉 (211300063、211300063@nju.smail.edu.cn)

(南京大学人工智能学院, 南京　210093)

1　对于现有的特征提取方法，收集训练数据，尝试三种以上的学习方法，撰写学习方法的介绍，报告性能对比

1.1 我的游戏策略

因为“监督学习只能模拟你的行为，如果你的行为有错误，不能纠正你的错误”，所以收集到的数据对学习效果至关重要。在这个游戏，我主要使用以下策略进行一场漂亮的对战：开局先跑到一个固定的位置，然后根据目标运动的速度每隔一段时间间隔就发射弹药，如果我遗漏了一个目标使它跑到了下一层，那么我会优先消除掉这一个目标，因为这个目标相对而言是比较危险的，遇到炸弹的时候我会躲，然后重新选择位置进行射击，基于这种策略，我的胜率还是比较高的。

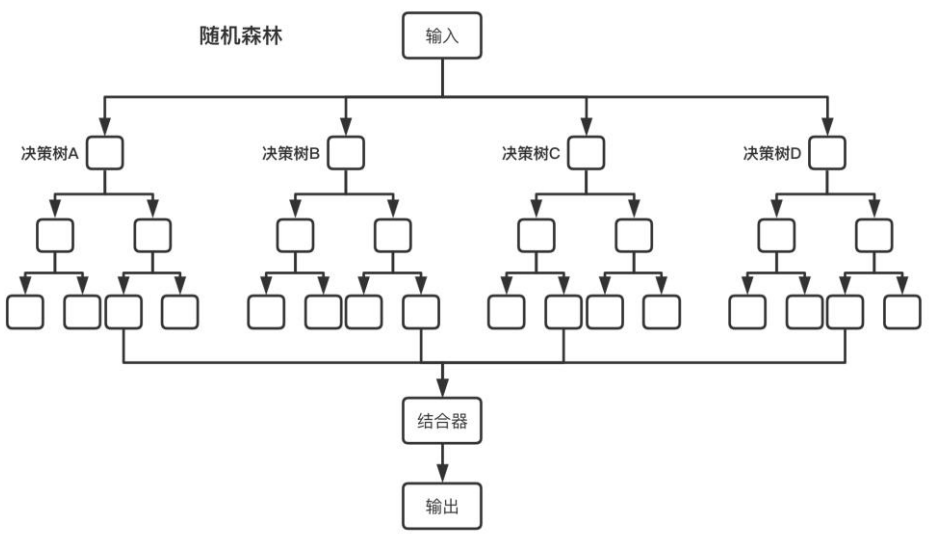
在进行足够多的对局之后，我开始使用这些数据进行来训练模型。

1.2 我选择的学习算法

1.2.1　RandomForest

随机森林算法一种基于决策树的集成学习算法，集成学习通过训练学习出多个估计器，当需要预测时通过结合器将多个估计器的结果整合起来当作最后的结果输出。

随机森林算法将多个决策树结合在一起，每次数据集是随机有放回的选出，同时随机选出部分特征作为输入。可以看到随机森林算法是以决策树为估计器的集成算法。



随机森林算法要求每棵决策树的输入必须尽可能的不相关，因为不同决策树的输入相关性越大，其输出的相似性就越大。举一个极端的例子，如果森林中每棵树的输入都相同，那么输出也应该都是一样的，采用森林就没有必要了。对于随机森林算法，影响其分类能力的最重要的参数就是森林中每棵树随机抽取的数据

的数量，数量大了则每棵树之间的相关性增强，从而使森林的分类能力减弱；数量小了则每棵树自身的分类能力减弱，也使森林的分类能力减弱。所以关键是找到二者之间的平衡。因为有每棵树的输入的随机性和基于统计结果进行决策这两种特性，随机森林算法有良好的抗噪声能力和防止过拟合的能力。

1.2.2 AdaBoost

AdaBoost 是一种迭代算法，核心思想也是集成学习的思想，针对同一个训练集训练不同的弱分类器，然后把这些弱分类器集合起来，构成一个更强的最终分类器（强分类器）。AdaBoost 是自适应增强的意思。AdaBoost 方法的自适应在于：前一个分类器分错的样本会被用来训练下一个分类器。AdaBoost 方法对于噪声数据和异常数据很敏感，相对于大多数其它学习算法而言，不会很容易出现过拟合现象。AdaBoost 方法中使用的分类器可能很弱（比如出现很大错误率），但只要它的分类效果比随机好一点（比如两类问题分类错误率略小于 0.5），就能够改善最终得到的模型。而错误率高于随机分类器的弱分类器也是有用的，因为在最终得到的多个分类器的线性组合中，可以给它们赋予负系数，同样也能提升分类效果。

1.2.3 KNN

KNN，又称 K 近邻算法：给定一个训练数据集，对新的输入实例，在训练数据集中找到与该实例最邻近的 K 个实例，这 K 个实例的多数属于某个类，就把该输入实例分类到这个类中（这就类似于现实生活中少数服从多数的思想）。K 值的选取对于 KNN 算法影响很大，具体地，如果选取较小的 K 值,就相当于用较小的领域中的训练实例进行预测，“学习”近似误差会减小，只有与输入实例较近或相似的训练实例才会对预测结果起作用，与此同时带来的问题是“学习”的估计误差会增大，换句话说，K 值的减小容易发生过拟合。如果选取较大的 K 值，就相当于用较大领域中的训练实例进行预测，其优点是可以减少学习的估计误差，但缺点是学习的近似误差会 增大。这时候，与输入实例较远(不相似的)训练实例也会对预测器起作用，使预测发生错误。在实际应用的，K 值一般取一个比较小的数值，例如采用交叉验证法(一部分做训练集，一部分做测试集)来选择最优的 K 值。

1.2.4 NaiveBayes

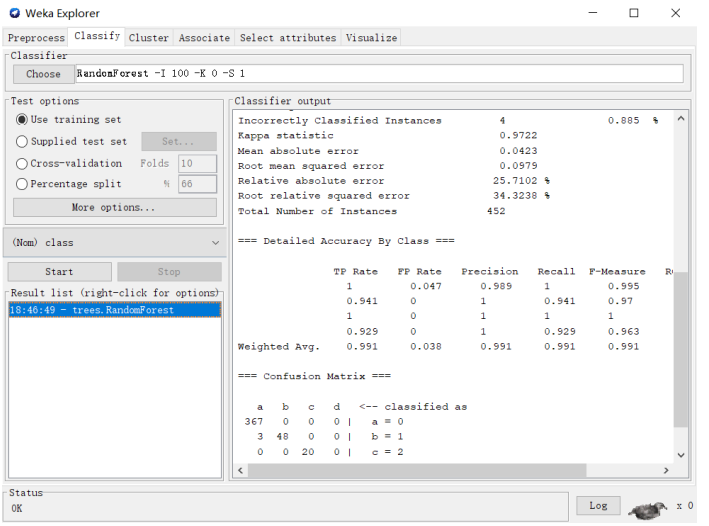
朴素贝叶斯分类是一种十分简单的分类算法，其基础是贝叶斯公式（下图），朴素贝叶斯的思想基础是这样的：对于给出的待分类项，求解在此项出现的条件下各个类别出现的概率，哪个最大，就认为此分类项属于哪个类别。

$$P(A_i|B) = \frac{P(B|A_i)P(A_i)}{\sum_j P(B|A_j)P(A_j)}$$

朴素贝叶斯算法假设了数据集属性之间是相互独立的，因此算法的逻辑性十分简单，并且算法较为稳定，当数据呈现不同的特点时，朴素贝叶斯的分类性能不会有太大的差异。换句话说就是朴素贝叶斯算法的健壮性比较好，对于不同类型的数据集不会呈现出太大的差异性。当数据集属性之间的关系相对比较独立时，朴素贝叶斯分类算法会有较好的效果。属性独立性的条件是朴素贝叶斯分类器的不足之处，数据集属性的独立性在很多情况下是很难满足的，因为数据集的属性之间往往都存在着相互关联，如果在分类过程中出现这种问题，会导致分类的效果大大降低。

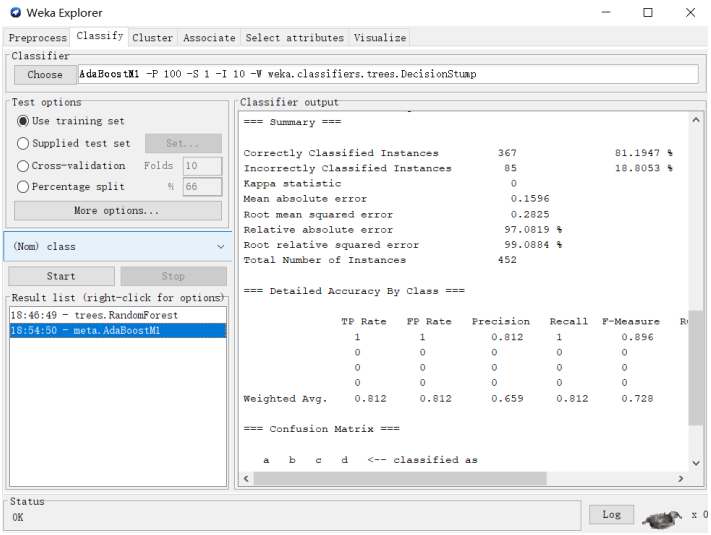
1.3 四种算法性能对比

1.3.1 RandomForest



使用这种算法训练出来的模型，精灵会左右移动躲避炸弹，但成功率不是很高，大概有百分之 70 的几率可以赢得游戏（平均 550ticks）。

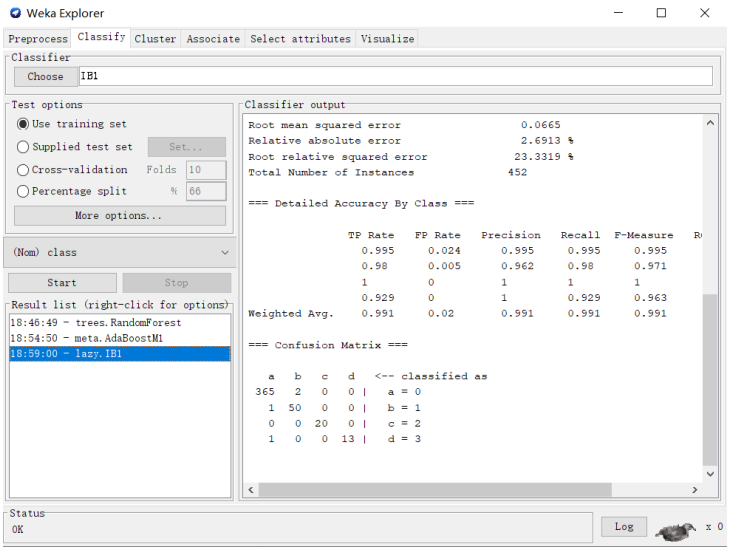
1.3.2 AdaBoost



不知道为什么，这种方法训练出来的模型，精灵会选好一个位置侯就不动了，然后进行连续射击，大概

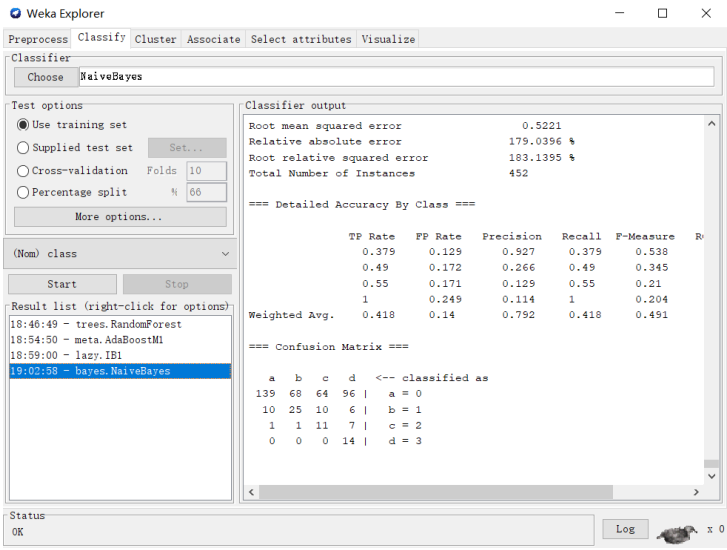
率精灵都会被炸弹炸死。

1.3.3 KNN



使用 KNN 训练出来的模型，躲避炸弹的成功率比使用随机森林算法的更高，但是射击的准度却下降，不过也能赢得游戏胜利，胜率大概为百分之 70 左右（平均 600ticks）。

1.3.4 NaiveBayes



这个模型相对其他模型，最大的优点就是它的射击把握得很好，基本上不会出现射空的情况，但在躲避炸弹上表现得不如随机森林和 KNN，但大概率能获胜(平均 ticks450).

1.3.5 总结

除了 AdaBoost 之外，其他的模型或多或少都能初步模仿我的游戏策略，在射击准度上：NaiveBaye> RandomForest> KNN>AdaBoost,在躲避炸弹上：KNN>RandomForest>NaiveBaye>AdaBoost.

2 尝试修改特征提取方法，得到更好的学习性能

2.1 修改特征提取方法

原代码中 4 中状态都记录在 feature[448],所以我进行了修改。

默认的特征提取方法记录的是整个地图的信息和四种游戏状态，我觉得不够具体，于是我在原有基础上增加了：（1）记录精灵所在列的炸弹（2）记录离精灵最近的目标的距离（3）目标剩余数量。

```
            map[x][y] = o.itype;
        }
        for(int y=0; y<14; y++)
            for(int x=0; x<32; x++)
                feature[y*32+x] = map[x][y];

        // 4 states
        feature[448] = obs.getGameTick();
        feature[449] = obs.getAvatarSpeed();
        feature[450] = obs.getAvatarHealthPoints();
        feature[451] = obs.getAvatarType();

        feature[452] = bomb * 100;
        feature[453] = target_nums * 20;
        feature[454] = target_dis * 60;

        return feature;
    }
```

2.2 修改后的性能

最终训练结果显示，学习器在训练集上的准确率有所降低，但是当将模型运用在实际游戏中产生了更好的效果。我认为之所以改变前在训练集上的准确率提高是因为过拟合了，而修改后的特征提取方法虽然在训练集上拟合差了点，但是预测新情况更准确，因为它考虑了更多的特征。

修改后提升比较明显的是 KNN 模型，在射击准确率和躲避炸弹成功率上都有一点提高，其次是 RandomForest 和 NaiveBaye，AdaBoost 还是一样摆烂，跑到一个位置固定不动进行射击。

3 结束语

这次作业是关于监督学习的，通过这次作业，我对监督学习也有了更深刻的了解，监督学习比较依赖训练集的好坏，这次作业的训练集都是我自己玩游戏获得，所以这个训练集太小而且不够普遍，这也是导致学习器效果不佳的主要原因。

对于课堂上讲的一些监督学习模型有了更直观的了解，但是对于其他一些更高级的算法我还没有能力去理解和运用。