

# 模式识别——基于 ID3 算法的三次改进

191132 詹才韬 2016 年 7 月 7 日

## Abstract

ID3 算法是决策树的鼻祖，最早于 1986 年由 Quinlan 提出，全称是 Iterative Dichotomiser 3 [1]。在这篇课程报告中，我将对经典的 ID3 做出三次改进：1. 把 *info gain* 改进为 *gain ratio*；2. 把简单投票的过程改进为朴素贝叶斯的方法；3. 将许多颗 ID3 决策树打造成随机森林。本人将在 Weka 平台进行二次开发，并且用 Weka-Experiment 做大量实验，和其它著名的算法进行比较，最后做出综述。项目的源代码开源在本人的 GitHub 主页上。

## Improvement One

经典 ID3 算法构造一棵树的过程如下：

1. 构造根节点：输入数据集 *data*，找出 *info gain* 最大的属性 *attribute*，用属性 *attribute* 对数据集 *data* 划分成若干子节点。子节点中的数据集 *data'* 是其父节点中数据集 *data* 的一个子集
2. 如果子节点的 *info gain* 等于 0，则子节点成为叶节点，停止生长树
3. 如果子节点的 *info gain* 不等于 0，则以该子节点为“根节点”，继续长树，即回到步骤 1

这里可以改进的地方在于 *info gain*，在 ID3 中， $info\ gain = entropy(S) - entropy(S, A)$  [2]。其中， $entropy(S)$  为划分样本集 *S* 为 *c* 个类的熵， $entropy(S, A)$  为属性 *A* 划分样本集 *S* 导致的期望熵。当 *data* 越“纯”， $entropy$  就越小，子节点的  $entropy$  之和就越小，这样 *info gain* 就越大。我们希望 *info gain* 越大越好。

问题来了，现在输入一个数据集，有一个属性是这样的：有很多的取值，甚至每一个实例的该属性上的值都不一样。比如在 *Weaheer.nominal* 数据集中增加一个名为 *IDcode* 的属性，那么 ID3 算法构造的树如图-1。为了解决这个问题，提出了如下改进[3]：

引入  $gain\ ratio = \frac{info\ gain}{split\ info}$

```
IDcode = a: no
IDcode = b: no
IDcode = c: yes
IDcode = d: yes
IDcode = e: yes
IDcode = f: no
IDcode = g: yes
IDcode = h: no
IDcode = i: yes
IDcode = j: yes
IDcode = k: yes
IDcode = l: yes
IDcode = m: yes
IDcode = n: no
```

图-1：训练集=*Weather.nominal.IDcode*，算法=ID3。算法选择了 *IDcode* 这个属性对数据集进行划分。然而这样是无法对新来的实例进行预测的，因为每一个实例的 *ID code* 都不一样。

其中， $split\ info =$

$$\sum_{i=1}^n \left( - \frac{numSubset[i]}{numTotal} \times \log_2 \frac{numSubset[i]}{numTotal} \right)$$

这样一来，使用 *gain ratio* 来替代 *info gain*，可以抵消部分某属性的取值过多的不利因素，如图-2

▼ infoGains	(id=88)
▲ [0]	0.9402859586706307
▲ [1]	0.24674981977443894
▲ [2]	0.029222565658954536
▲ [3]	0.15183550136234125
▲ [4]	0.04812703040826921
▲ [5]	0.0
▼ splitInfo	(id=90)
▲ [0]	3.8073549220576055
▲ [1]	1.5774062828523452
▲ [2]	1.5566567074628228
▲ [3]	1.0
▲ [4]	0.9852281360342516
▲ [5]	0.0
▼ gainRatio	(id=91)
▲ [0]	0.24696566984684284
▲ [1]	0.15642756242117506
▲ [2]	0.018772646222418598
▲ [3]	0.15183550136234125
▲ [4]	0.04884861551152054
▲ [5]	0.0

图-2：训练集=*Weather.nominal*，算法=ID3\_gain-ratio。*IDcode* 的例子，其 *IDcode* 的 *info gain* = 0.940，是第二大 0.246 的接近 5 倍。它的 *gain ratio* = 0.247，只是第二大的 0.156 的 1.5 倍多一点。

## Improvement Two

ID3 在构造的决策树之后，对于一个新样本的进行预测的时候，只是简单的对叶节点进行投票，所谓投票，就是少数服从多数。这样虽然简单，但是未必就是最好的，如图-3 所示。我们可以利用局部学习的原理[4]对算法进行优化：对叶子节点进行朴素贝叶斯分类。本人在 UCI 的官网上找到一个数量更大的训练集 Car-Evaluation[5]，作为新的数据训练集。

```
=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances      1544      89.3519 %
Incorrectly Classified Instances    178      3.5301 %
Kappa statistic                     0.9071
Mean absolute error                 0.019
Root mean squared error             0.1379
Relative absolute error             9.4937 %
Root relative squared error         45.0502 %
UnClassified Instances              123      7.1181 %
Total Number of Instances          1728
```

图-3：训练集=Car-Evaluation，算法=ID3。发现有不少实例“未分类”，这可能是空叶子节点造成的，影响了分类正确率。

改进方法：对 ID3 树的生长做出一定限制，少长几层，然后在叶节点上面做 naïve bayes 优化。这个简单的思想，效果十分不错，如图-4 所示：

```
=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances      1614      93.4028 %
Incorrectly Classified Instances    114      6.5972 %
Kappa statistic                     0.8557
Mean absolute error                 0.0387
Root mean squared error             0.1557
Relative absolute error             16.9075 %
Root relative squared error         46.0355 %
Total Number of Instances          1728
```

图-4：训练集= Car-Evaluation，算法=ID3-NB。消除了未分类的实例，虽然其中部分转换成了错误的，但是也有部分转换成了正确的，因此预测正确率提升了 4 个百分点。Car-Evaluation 有 6 个属性，1 个类标签。当树的深度限制在 5 的时候，正确率=93.4028%

## Improvement Three

ID3 算法是一个经典确定性算法，我在第三次改进方法中，试图在确定性算法中增加随机性，利用多颗随机的树，形成一个随机森林(random tree)。我的版本的随机森林比较原始，参照了随机子空间树[6]，方法如下：假设森林里面有 n 棵树，那么使用 n 个不同

的数据集训练这 n 棵树。这 n 个数据集的不同之处在于：不同的数据集缺少不同的属性。在生成这 n 个数据集的时候，会从原始的数据集中随机删除固定数量的属性。我写的随机森林运行结果如图-5，Weka 提供的随机森林运行结果如图-6 所示：

```
=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances      1419      82.1181 %
Incorrectly Classified Instances    309      17.8819 %
Kappa statistic                     0.5355
Mean absolute error                 0.1232
Root mean squared error             0.2414
Relative absolute error             53.8101 %
Root relative squared error         71.3841 %
Total Number of Instances          1728
```

图-5：训练集=Car-Evaluation,算法=ID3-random\_forest 正确率一般在 80%上下徘徊。很遗憾，我写的随机森林比 ID3 算法接近 90%的正确率低了 10 个百分点。背后的原因有待进我一步思考。随机森林里面是需要下很多功夫的。

```
=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances      1625      94.0394 %
Incorrectly Classified Instances    103      5.9606 %
Kappa statistic                     0.8703
Mean absolute error                 0.0745
Root mean squared error             0.1667
Relative absolute error             32.5232 %
Root relative squared error         49.3132 %
Total Number of Instances          1728
```

图 -6：训练集 =Car-Evaluation,算法 = Weka.trees.RandomForest。Weka 里面的专家写出来的随机森林性能很高，达到了 94.0394%。

## Weka-Experiments

真正的研究工作需要对不同算法对不同数据集做大量的实验工作，此时需要使用实验者界面。实验结果如图-7 所示。

## Conclusions and Future Work

本文提出的三次改进，可以分为两个方向：1.构造树的方法；2.局部学习原理。其中改进一和改进三是在构造树的方面下功夫，改进二是在整个训练实例的局部下功夫，改进一有一定的效果，改进二取得了不错的效果。改进二就是 ID3+naïve bayes，相当于是一个 NBtree 的雏形。但是改进三没有取得预期的效果，可能是因为我写的随机森林效果不好。

未来工作有以下几点：

1. 在 Improvement Two 中，ID3 树的生长的深度限制在多少，可以进行进一步的研究。
2. 在 Improvement Three 中，本人费劲心思写出来的“随机森林”，正确率反而比 ID3 更差。虽然很使我伤心，但是我在写代码、调试代码、还有思考的过程中有了不少长进。看来随机森林不是那么容易就可以随机出来的。
3. 可以把这三种改进方法糅合在一起，看看三种改进组合在一起，能不能产生性能更加的算法。
4. 上述所有方法都是基于属性为 nominal 的数据集，可以进一步研究属性为 numerical，甚至是两者混合的数据集。

Dataset	(1) caitao.ID3   (2) caita (3) caita (4) caita (5) trees (6) trees						
car.evaluation	(100)	89.19	81.36 *	92.85 v	78.89 *	94.43 v	93.40 v
weather.symbolic	(100)	79.00	56.50	79.00	79.00	57.50	68.50
		(v/ /*)	(0/1/1)	(1/1/0)	(0/1/1)	(1/1/0)	(1/1/0)

Key:

```
(1) caitao.ID3 '' -2693678647096322561
(2) caitao_gainRatio.ID3 '' -2693678647096322561
(3) caitao_naiveBayes.ID3 '' -2693678647096322561
(4) caitao_randomForest.ID3 '' -2693678647096322561
(5) trees.NBTree '' -4716005707058256086
(6) trees.RandomForest '-I 10 -K 4 -S 1 -depth 10' -2260823972777004705
```

图-7: Weka-experiment 实验结果。总共 6 个算法，2 个数据集。6 个算法中(1)是原始的 ID3 算法，后面(2)-(4)是本人的改进算法，(5)和(6)是 Weka 平台自带的算法。

## GitHub

本次模式识别上机实习的代码，全部公开在本人的 GitHub 主页上面，url 地址如下：

### 1. Improvement One:

[https://github.com/caitaozhan/ID3\\_improvements/tree/gain\\_ratio](https://github.com/caitaozhan/ID3_improvements/tree/gain_ratio)

### 2. Improvement Two:

[https://github.com/caitaozhan/ID3\\_improvements/tree/naive\\_bayes](https://github.com/caitaozhan/ID3_improvements/tree/naive_bayes)

### 3. Improvement Three:

[https://github.com/caitaozhan/ID3\\_improvements/tree/random\\_forest](https://github.com/caitaozhan/ID3_improvements/tree/random_forest)

较风趣；另一方面，蒋老师在我上机实习的过程过，回答了我不少疑惑，虽然这些疑惑对于蒋老师而言可能十分幼稚，但是依然完整解决了我的问题。

## REFERENCES

- [1][https://en.wikipedia.org/wiki/ID3\\_algorithm](https://en.wikipedia.org/wiki/ID3_algorithm)
- [2]决策树，蒋良孝的 PPT Chapter 2-8
- [3]*Data Mining Practical Machine Learning Tools and Techniques* -- Chapter4.3
- [4]贝叶斯分类，蒋良孝的 PPT Chapter 3-15
- [5]<http://archive.ics.uci.edu/ml/datasets/Car+Evaluation>
- [6][https://en.wikipedia.org/wiki/Random\\_subspace\\_method](https://en.wikipedia.org/wiki/Random_subspace_method)

## Acknowledgements

感谢蒋良孝老师对于我的指导。一方面，蒋老师上课讲解十分到位，关键部位一点就通了，不仅如此还比