实验二 决策树的实现

18308133 刘显彬

September 29, 2021

1 实验原理

1.1 决策树

1.2 ID3模型

1.3 C4.5模型

1.4 Cart模型

1.5 决策树的形成过程

2 伪代码

算法1:计算给定数据集的熵值;

Algorithm 1 cal-HD(l)

Input: $l : label\ of\ dataset(D)$
Output: $H(D)$

```
labels ← label ∈ l
freq ← frequency of label
n ← num of label
HD = 0
for i ← 0 to n do
    HD += freq[i] * log(freq[i]);
end for
reurn HD
```

算法2:计算信息增益

Algorithm 2 Gain(d, attr, HD)

Input: d: dataset, $attr$: split attr, HD: entropy of d
Output: $gain$

// 分裂数据
$values \leftarrow value \in d[attr]$;
$N \leftarrow len(d)$;
set $spData$ = empty dict;
for v such that $v \in values$ **do**
  $spData[v] = subD, wheresubD[attr] ==' v' andsubD \in d$
end for
$H(D)_A$=0
for sub such that $sub \in spData$ **do**
  // 计算各个子数据集的熵，并求和
  $H(D)_A \mathrel{+}= \frac{len(sub)}{N} * cal\_HD(sub)$;
end for
return $HD - H(D)_A$

Algorithm 3 GainRatio(d, attr, HD)

Input: d: dataset, $attr$: split attr, HD: entropy of d
Output: $gainRatio$

// 像前一个算法一样分裂数据集
$spData \leftarrow$ split d with val $\in attr$
// 计算$SplitInfo$
$splitInfo \leftarrow 0$
$N \leftarrow len(d)$;
for sub such that $sub \in spData$ **do**
  $SplitInfo \mathrel{+}= \frac{len(sub)}{N} * log(\frac{len(sub)}{N})$
end for
return $\frac{Gain(d,attr,HD)}{SplitInfo}$

Algorithm 4 Gini(d)

Input: d: dataset
Output: $gini$

// 分成不同的类
$spData \leftarrow$ split d with different label
$N \leftarrow len(d)$
计算这些类的比重
$freqs \leftarrow \frac{len(dset)}{N}\ \forall dset \in spData$

return $1 - \sum_{freq \in freqs} freq^2$

Algorithm 5 Gini(d, attr)

Input: d: dataset, $attr$: split attr
Output: $gini$

// 像前一个算法一样分裂数据集
$spData \leftarrow$ split d with val $\in attr$
$N \leftarrow len(d)$
$freqs \leftarrow \frac{len(dset)}{N}\ \forall dset \in spData$

$gini \leftarrow 0$
for $i \leftarrow 0$ *to* $len(freqs)$ **do**
  $gini\ +=\ freqs[i] * Gini(spData[i])$;
end for
return $gini$

Algorithm 6 buildTree(root, d, alg)

Input: d: dataset, $root$:决策树根, alg:计算信息熵的算法
  if $d.attr == null$ or only one label in d.labels **then**
    这是一片叶子
    root['attr']='leaf', root['val']=vote_max(d.labels)
    return root
  else
    $best \leftarrow -inf$;
    $bestattr \leftarrow''$;
    //根据给定的算法找出最优的属性
    for all $attr \in d.attr$ **do**
      $best \leftarrow alg(d, attr)$;
      $bestattr \leftarrow argmax(best, attr)$;
    end for

    //然后对最优属性进行分裂，对子节点进行迭代
    $root['attr'] \leftarrow bestattr$
    $spData \leftarrow$ d splitted by bestattr;
    for $sub \in spData$ **do**
      $root['val'] \leftarrow buildTree(root['val'], sub, alg)$;
    end for
  end if
Output: $root$

Algorithm 7 predict(root, data)

Input: root: 决策树树根, data: 待预测的数据
Output: predVal: 预测值
  $cur \leftarrow root$
  //当前不是叶子时，进行搜索
  $attr \leftarrow cur['attr']$
  while $attr$!= $'leaf'$ **do**
    //进入data[attr]对应的一支分枝
    $cur \leftarrow cur['val'][data[attr]]$
    $attr \leftarrow cur['attr']$
  end while
  //返回叶子的预测label
  return cur['val']