2019.3.28

1. 决策树算法（Decision Tree）-- ID3

决策树是一种逼近**离散值**目标函数的方法。

1.1表示：决策树代表实例属性值约束的合取的析取式。

节点：属性

叶子节点：实例分类

分支：属性可能的取值

1.2分类：有监督的学习

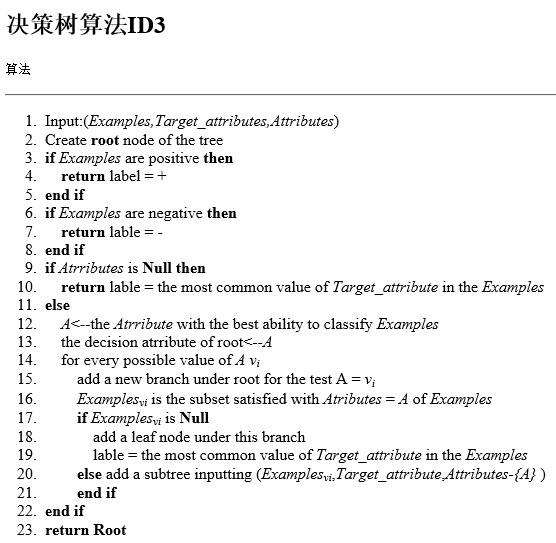
1.3适用问题

实例由“属性-值”对表示，即分类属性最好由离散值组成；

目标函数具有离散的输出值；

=》分类问题

1.4算法过程



1.5核心概念：属性A的选择

1. 熵(Entropy)：系统S中目标属性具有c个不同的值，则

，pi为S中i类的比例

（2）信息增益(评估函数)：一个属性A相对于样例集合S的信息增益



Values(A)：属性A 所有可能值的集合，Sv是S中属性A的值为v的子集。

（3）使系统信息增益增加得多的属性作为分类属性。

1.6举例

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Day | Outlook | Temperature | Humidity | Wind | PlayTennis |
| D1 | Sunny | Hot | High | Weak | No |
| D2 | Sunny | Hot | High | Strong | No |
| D3 | Overcast | Hot | High | Weak | Yes |
| D4 | Rain | Mild | High | Weak | Yes |
| D5 | Rain | Cool | Normal | Weak | Yes |
| D6 | Rain | Cool | Normal | Strong | No |
| D7 | Overcast | Cool | Normal | Strong | Yes |
| D8 | Sunny | Mild | High | Weak | No |
| D9 | Sunny | Cool | Normal | Weak | Yes |
| D10 | Rain | Mild | Normal | Weak | Yes |
| D11 | Sunny | Mild | Normal | Strong | Yes |
| D12 | Overcast | Mild | High | Strong | Yes |
| D13 | Overcast | Hot | Normal | Weak | Yes |
| D14 | Rain | Mild | High | Strong | No |

（1）计算系统的熵







1. 确定根节点的属性

Outlook：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Sunny | Overcast | Rain | 和 |
| PlayTennis = Yes | 2 | 4 | 3 | 9 |
| PlayTennis = No | 3 | 0 | 2 | 5 |









Temperature、Humidity、Wind均进行以上相同计算，取Gain值最大的属性作为根节点属性。

1.7特点

（1）假设空间中可能不存在目标函数。

（2）收敛到局部最优而非全局最优。

（3）使用统计信息可疑降低个别训练样例错误的敏感性。

1.8归纳偏执：可以分类未见实例

（1）近似的归纳偏置：较短的树比较长的树优先（广度优先搜索算法：BFS-ID3）。

（2）更近似：较短树比较长树优先；信息增益高的属性更靠近根节点的树优先。

1.9优化

1. 避免过度拟合数据（将噪声数据当作正常训练数据）

解决办法：

及时停止树的增长：当验证数据集取得较好结果时，停止增长。主要用于有大量数据可供使用的情况，1/3的数据用于验证，2/3的数据用于训练。

过度拟合之后修剪：将决策树尽可能拟合训练数据；将决策树转换为等价的规则集合（从根节点到叶子节点的一条路径为一条规则）；删除任何能导致估计精度（一种方法：验证数据的结果）降低的前件来泛化规则。

1. 合并连续值（分类属性值为连续）

>X：A类；<=X：B类

阈值的确定：可以取突变值的平均值

1. 缺乏属性值的训练样本

可以补充该属性中最常见的属性值；该节点被分类后的最常见的属性值。

2019.4.15

2、人工神经网络——反向传播算法

人工神经网络对于逼近**实数值**、**离散值**或者**向量值**的目标函数提供一种健壮的方法。

2.1适用问题

实例用很多“属性-值”对表示。

目标函数的输出可以是离散值、实数值或者有若干实数属性或者离散属性组成的向量。

训练数据中可以包含错误。

可容忍长时间的训练。

可能需要快速求出目标函数的值。

人类能否理解学到的目标函数并不重要。学到的神经网络难以传达给人类。

2.2感知器：值加权求和,再加一个简单的阈值函数（>0，为正例；<0，为反例）

xn

x0=1

w0

x1

x2

w1

w2

wn

**Σ**





（1）表征能力

线性可分：一个感知器可以分类所有实例。例如：与、或、非、或非。

仅用两层感知网络就可以表示所有的布尔函数，包括异或。

（2）感知器训练法则

过程：从随机权值开始，反复应用到感知器的每个训练样例，只要误分类样例就修改权值。重复，直到对所有样例正确分类。权值修正公式如下：



t为当前训练样例的目标输出（标准输出），o是感知器输出（实验输出，包括阈值化），η是一个正的常数称为学习率，通常很小（0.1）。

前提：样例线性可分，且使用充分小的学习速率。

2.3梯度下降和delta法则

（1）针对：样例线性不可分。

（2）权值更新公式为：



t为目标输出值，η为学习速率，o为将输入向量放入训练单元的非阈值化输出值。的更新公式表现出一定的累计性，即将之前的累加到本次误差中。

（3）推导过程：

（4）梯度下降的应用场景：假设空间包含连续参数化的假设；误差对于这些假设参数可微。

（5）问题：收敛过程缓慢；可能收敛到局部最小。

（6）缓解：增量梯度下降（随机梯度下降），即将的更新公式变为：



消除之前的累计效果，每次都从关心本次的误差计算。

（7）前提：使用充分小的学习速率。

2.4对比

感知器训练法则：阈值化（t和o的计算）、样例线性可分

梯度下降：非阈值化、无论线性可分与否

2.5多层网络和反向传播算法

1. 可微阈值单元（挤压函数）-》多层网络的感知器

为了表征非线性函数的网络，需要一种函数，其输出是输入的非线性函数，且输出是输入的可微函数。所以找到logistic函数：



其导数，易求。

x2

w1

w2

wn

xn

x0=1

w0

X1

**Σ**





（2）反向传播算法：学习网络的权值

输入：（training\_examples,η,nin ,nout ,nhidden）

//training\_examples中的训练样例形式为<,>的序偶对，为网络输入向量，为目标输出向量。η为学习速率，nin 为网络输入的数量，nout 为输出单元的数量，nhidden 为隐藏层单元数量。

# 初始化网络中的所有权值为较小的随机数（例如：-0.5到0.5之间的数）；

# 直到遇到终止条件：

# 对于训练样例中的每个<,>：

把输入沿网络向前传播

1. 把实例输入网络，并计算网络中的每一个单元u的输出ou (包括隐藏和输出层)

使误差沿网络反向传播

1. 对于网络中的每个**输出单元k**,计算误差项



其中，为ok 的导数，参见以上挤压函数。

1. 对于网络的每个**隐藏单元h**，计算误差项



其中，表示单元h到单元k的权值

1. 更新每个网络权值



其中，。表示单元i到单元j的输入。

（3）终止条件

迭代次数到一个固定值；

训练样例误差降到某阈值下；

验证误差符合某标准。

（4）变体

增加冲量项：使第n次迭代的权值和第n-1次的权值相关。即将更新公式变为：



可以加速收敛。

学习任意无环网络：将两层反向传播算法扩展到多层，例如：第m层的单元r的值由更深层的m+1层的决定，即：



对应以上第3步的公式（隐藏层更新公式）。

（5）问题：

\* 可能收敛到局部极小点，但是没有那么严重，可以使用启发式规则来解决这个问题，例如：

冲量项；

随机梯度下降；

使用不同的随机值初始化每个网络。

\* 表征能力。

任何布尔函数可以使用两层网络准确表示；每个有界连续函数可以由一个两层网络以任意小的误差逼近；任意函数可以以三层网络任意精度逼近。

\* 假设空间搜索和归纳偏置。空间连续；偏置在数据点之间平滑插值。

\* 隐藏层。隐藏层可以自动发现有用表示的能力。

\* 泛化、过渡拟合和停止判断。可以使用以下方法：

权值衰减，在迭代过程以某个小因子降低每个权值；

训练数据外再为算法提供一套验证数据，在验证集合上产生最小误差的迭代次数。

2.6举例

贝叶斯学习

难度：需要概率的初始知识；一般情况下确定贝叶斯最优假设的计算代价比较大。

1.贝叶斯法则

贝叶斯公式，表6-1公式

具有最大可能性的假设称为极大后验假设-使用贝叶斯公式计算候选假设的后验概率。

似然度，极大似然假设

2.蛮力贝叶斯

算法公式

前提：训练数据D是无噪声的；目标概念c在假设空间H中；没有理由认为某假设比其他假设的可能性更大。

一致学习器：输出的假设在训练样例上有零错误率。