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

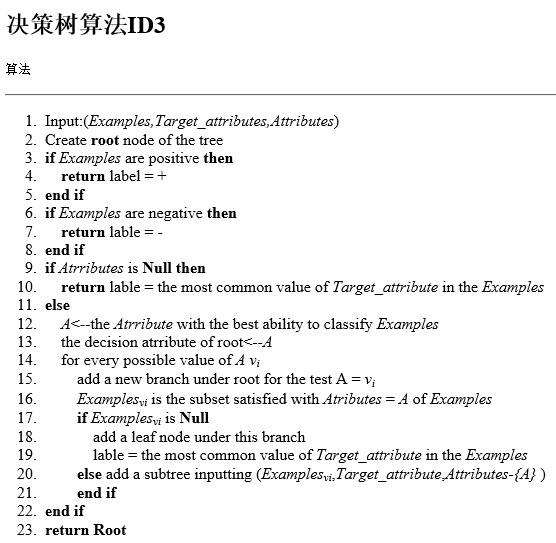
1.3适用问题

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。

X1

x2

w1

w2

wn

xn

x0=1

w0

**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

2019.4.18

3、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。

3.1贝叶斯法则

（1）贝叶斯公式



：先验概率，表示在没有训练数据时，假设h拥有的初始概率。如果没有这一先验知识，可以为每一个假设赋予相同的概率。，H为假设空间。

：要观察的训练数据D的先验概率，即D成立的概率。

：表示在假设h成立时，观察到D成立的概率。

：后验概率，就在观察到D成立，假设h成立的概率。这是我们要求的。

（2）在假设空间H中寻找给定数据集D最大成立可能性的假设h，这样具有最大可能性的假设称为**极大后验假设（MAP）**-》使用贝叶斯公式计算候选假设h1、h2、……hm的后验概率（）。



假设*假设空间中hi均具有相同的发生概率*，则。故常被称为给定h时数据D 的**似然度**，而使最大的假设为**极大似然（ML）假设**:



3.2蛮力贝叶斯

（1）算法

\* 对于假设空间中的每一个假设h,计算后验概率：



\* 输出具有最高后验概率的假设：



（2）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。-》对所有假设赋予相同的概率

（3）一致学习器：输出的假设在训练样例上有零错误率。-》假设成立，训练样例成立

H 上有均匀的先验概率，即每个假设的概率相同，且训练数据确定且无噪，任意的一致学习器将输出一个MAP假设。

（4）不足：计算量大。

3.3贝叶斯最优分类器-》给定训练数据，对实例最可能的假设（分类）是什么

（1）计算公式

V：分类集合{v1,v2,……,vm}，例如：正例、反例

在训练数据D中实例的分类为：



选取其中大概率的类别。

1. 举例

V={+，-}



所以，



=1\*0.4+0\*0.3+0\*0.3 =0.4



=0\*0.4+1\*0.3+1\*0.3 =0.6

故，该实例分类为反（-）。

3.4朴素贝叶斯分类器

（1）计算公式

训练数据中实例的表示形式：实例的属性值<a1,a2,……,an>，所以，

其中，。

：易计算，每个目标值在训练数据中的频率。

为简化计算，假设：*在给定目标值时属性值之间相互条件独立*。



综合以上内容，朴素贝叶斯的计算公式为：



（2）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 |

分类新实例：<Outlook=sunny,Temperature=cool,Humidity=high,Wind=strong>

解：

V={Yes,No}

a1=sunny,a2=cool,a3=high,a4=strong

P(Yes)=9/14,P(No)=5/14

P(sunny|Yes)=2/9 P(cool|Yes)=3/9 P(high|Yes)=3/9 P(strong|Yes)=3/9

P(sunny|No)=3/5 P(cool|No)=1/5 P(high|No)=4/5 P(strong|No)=3/5

V(Yes)=P(Yes)P(sunny|Yes)P(cool|Yes)P(high|Yes)P(strong|Yes)=0.0053

V(No)=P(No)P(sunny|No) P(cool|No)P(high|No)P(strong|No)=0.0206

故朴素贝叶斯分类结果为：No（可以对结果进行归一化）

对例子进行分析：

如果P(sunny|Yes)=0，则会使整个概率为0，但是并不能说明此实例一定分类为No。



所以可以对其中的概率采取一个估计值（m-估计）：



其中，p是将要确定的概率的先验概率，即训练样本中p(sunny)，m为等效样本的大小。

3.5贝叶斯信念网

（1）基本概念

任意随机变量集合Y1……Yn，其中Yi可取值的集合为V(Yi)。

联合空间：变量集合Y的联合空间为，每一项对应变量元组<Y1……Yn>的一个可能性。

联合概率分布：在联合空间上的概率分布，指定了元组<Y1……Yn>的每个可能的变量约束概率。

（2）条件独立性

设X、Y、Z为3个离散值随机变量。如果给定Z值时X服从的概率分布独立于Y的值，称X在给定Z时**条件独立**于Y，即：



其中，。通常上式可以写为：



将这一条件独立性的定义扩展到集合：



即：变量集合变量在给定集合时条件独立于。

（3）表示

有向无环图：节点表示随机变量，弧表示概率依赖。

局部条件概率集合：在给定前驱节点时，后继发生的概率。

*断言：每一个节点在其****前驱节点的值****给定后，此节点条件独立于其他非前驱节点。*

任意随机变量组合的联合条件概率分布可表示为：



举例说明：

Storm是Lightning的直接前驱，Lightning是的后继。

Campfire的条件概率集合：S(Storm)和B(BusTourGroup)均发生，C(Campfire)发生的概率为0.4。

即：

Storm

BusTourGroup

Campfire

ForestFire

Thunder

Lightning

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | S,B | S, | ,B | , |
| C | 0.4 | 0.1 | 0.8 | 0.2 |
|  | 0.6 | 0.9 | 0.2 | 0.8 |

Campfire

（4）推理

贝叶斯网络可用于早知道某些变量的值或者分布时计算网络中另外一部分的概率分布。

对任意贝叶斯网络的概率的确切推理为NP难题，近似推理可能也为一个NP难题。

（5）学习贝叶斯信念网-》学习条件概率表（上例右表）

\* 已知网络结构（上例左图）且变量可以从训练样例中完全获得时，学习条件概率表就比较简单，直接使用朴素贝叶斯分类器。

\* 网络结构已知，但是只有一部分变量值能在数据中观察到。

1）贝叶斯的梯度下降训练

目的：使最大时的贝叶斯网条件概率表的参数（在前驱发生时，后继发生的概率）

方法：用对概率求导。

设是在给定父节点时，网络变量的概率，即：



标记：

推导：



现在引入变量和的值，通过在其可能的值和上求和，即：



推导中的最后一步：来自于概率乘法公式：

，在此求和过程中唯一不为0的项







结论：

# 

#更新公式：，η为学习率。然后将wijk 归一化。

不足：只能找到局部最优解。

\*网络结构未知，学习贝叶斯信念网很困难。

1）贝叶斯评分尺度

2）K2启发是搜索

3）基于约束的学习贝叶斯网络结构

2019.4.23

4、实用拜占庭容错算法（PBFT：Practical Byzantine fault tolerance）

<https://www.jianshu.com/p/fb5edf031afd>

4.1算法基本思想：状态机（服务作为状态机）副本复制算法，状态机在分布式系统的不同节点进行副本复制。****每个状态机的副本都保存了服务的状态，同时也实现了服务的操作****。

客户端 主节点 …… 其他副本

请求调用服务

将请求广播给其他副本

执行请求

返回结果

等待f+1个相同结果

①<REQUEST,o,t,c>

②三阶段协议

③<REPLY,v,t,c,i,r>

f：可能失效的最大副本数。

4.2详细内容：

（1）相关概念

\* 视图（View）：所有的副本在一个被称为视图的轮换过程（succession of configuration）中运作，表示为连续编号的整数。

\* 主节点：在某个视图中，某副本为主节点，其它副本为备份。当主节点失效的时候就需要启动视图更换（view change）过程。

主节点的确定（编号）p = v mod |R|

其中，v是视图编号，p是副本编号，|R|是副本集合的个数。

\* PBFT对每个副本节点的两个限定条件：1）所有节点必须是确定性的，即在给定状态和参数相同的情况下，操作执行的结果必须相同；2）所有节点必须从相同的状态开始执行。保证PBFT算法对所有非失效副本节点的请求执行总顺序达成一致（安全性）。

（2）上图的详细说明

①<REQUEST,o,t,c>

o：请求的操作。

t：时间戳，全序排列以保证客户端请求只会执行一次。

②三阶段协议

# 触发：当主节点p收到客户端的请求m，主节点将该请求向所有副本节点进行广播。

# 预准备：主节点分配一个序列号n给收到的请求，然后向所有备份节点群发预准备消息。

预准备消息的格式：<<PRE-PREPARE,v,n,d>,m>

其中，v是视图编号，m是客户端发送的请求消息，d是请求消息m的摘要。

接收与准备消息的条件（并）：

请求和预准备消息的签名正确，并且d与m的摘要一致；

当前视图编号是v；

该备份节点从未在视图v中接受过序号为n但摘要d不同的消息m；

预准备消息的序号n必须在水线（watermark）上下限h和H之间。

说明：水线存在的意义在于防止一个失效节点使用一个很大的序号消耗序号空间。

# 准备：该节点向所有副本节点发送准备消息<PREPARE,v,n,d,i>，并且将预准备消息和准备消息写入自己的消息日志。

接收准备消息之后，验证：

消息的签名是否正确；

视图编号是否一致；

消息序号是否满足水线限制。

如果验证通过则把这个准备消息写入消息日志中，标志准备阶段完成。

说明：预准备阶段和准备阶段确保所有正常节点对同一个视图中的请求序号达成一致。

# 确认阶段：副本i将<COMMIT,v,n,D(m),i>向其他副本节点广播。

确认完成committed(m,v,n)为真得条件为：任意f+1个正常副本节点集合中的所有副本i其prepared(m,v,n,i)为真；

本地确认完成committed-local(m,v,n,i)为真的条件为：prepared(m,v,n,i)为真，并且i已经接受了2f+1个确认（包括自身在内）与预准备消息一致；

确认消息与预准备消息一致的条件是具有相同的视图编号、消息序号和消息摘要。

# 执行：每个副本节点i在committed-local(m,v,n,i)为真之后执行m的请求，并且i的状态反映了所有编号小于n的请求依次顺序执行。在完成请求的操作之后，每个副本节点都向客户端发送回复。副本节点会把时间戳比已回复时间戳更小的请求丢弃，以保证请求只会被执行一次。

③<REPLY,v,t,c,i,r>

v：视图编号。

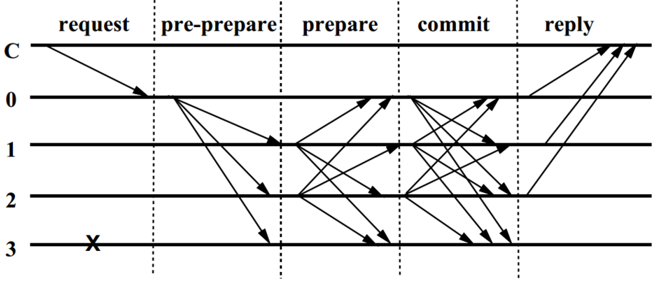
t：时间戳。

i：副本的编号.

r：请求执行的结果。

客户端等待f+1个从不同副本得到的同样响应，同样响应需要保证签名正确，并且具有同样的时间戳t和执行结果r。

（3）举例说明



Request：主机C向主节点0发送操作请求。

pre-prepare：主节点0向备份节点1、2、3发送操作请求。

Prepare：各备份节点将自己的执行结果发送给其他节点，包括主节点，

Commit：验证接收到的结果是否有f+1个结果一致。

Reply：验证结果发送给客户端。

在此过程中，备份节点3坏掉了，但是依然可以保证客户端收到的消息与没有坏掉的节点一致。