《机器学习》

课内实验报告

（1）ID 算法实现决议树

2015-2016 学年第 2 学期

专业：　智能科学与技术

班级：　智能 1301 班

学号：　06133029

姓名：张争辉

一、实验目的：

理解 ID3 算法的基本源理，而且编程实现。

二、实验要求：

使用 C/C++/MATLAB 实现 ID3 算法。

输入：

若干行，每行 5 个字符串，表示

OutlookTemperatureHumidityWindPlayball

如上表。

输出：

决议树。

实验结果以下：

输入：

Sunny Hot High Weak No
Sunny Hot High Strong No
Overcast Hot High Weak Yes
Rain Mild High Weak Yes
Rain Cool Normal Weak Yes
Rain Cool Normal
Overcast Strong No
Sunny Cool Normal Stron Ye
g s
Sunny Mild High
Rain Cool Normal Weak No
Mild Normal
Sunny Weak Yes
Overcast Mild Normal Weak Yes
Stron
Overcast Mild High g Yes
Rain
输出： Hot Normal Stron Yes
Outlook g
Mild High Weak Yes

RainWind Strong No

OvercastYes
SunnyHumidity StrongNo
WeakYes

NormalYes
HighNo

三、详细实现：

实现算法以下：

```
#include<iostream>
#include<fstream>
#include<math.h>
#include<string>
usingnamespacestd;
# defineROW1 4
# defineCOL5
#definelog20.69314718055
typedefstructTNode
{
    chardata[15];
    charweight[15];
    TNode* firstchild, * nextsibling;
}*tree;
typedefstructLNode
{
    char    OutLook[15];
    char    Temperature[15];
            Humidity[15];
    char    Wind[15];
    char    PlayTennis[5];
    char
    LNode*next;
}*link;
typedefstructAttrNode
{
    charattributes[15];//属性
    intattr_Num;//属性的个数
    AttrNode* next;
}*Attributes;
char* Examples[ROW][COL]={//"OverCast","Cool","High","Strong","No",
                    //      "Rain","Hot","Normal","Strong","Yes",
                        "Sunny","Hot","High","Weak","No",
                          "Sunny","Hot","High","Strong","No",
                          "OverCast","Hot","High","Weak","Yes",
                    "Rain","Mild","High","Weak","Yes",
                  "Rain","Cool","Normal","Weak","Yes",
                  "Rain","Cool","Normal","Strong","No",
                  "OverCast","Cool","Normal","Strong","Yes",
                  "Sunny","Mild","High","Weak","No",
                  "Sunny","Cool","Normal","Weak","Yes",
                  "Rain","Mild","Normal","Weak","Yes",
                  "Sunny","Mild","Normal","Strong","Yes",
                  "OverCast","Mild","Normal","Strong","Yes",
                  "OverCast","Hot","Normal","Weak","Yes",
                  "Rain","Mild","High","Strong","No"
                  };
char* Attributes_kind[4]={"OutLook","Temperature","Humidity","Wind"};
intAttr_kind[4]={3,3,2,2};
char*OutLook_kind[3]={"Sunny","OverCast","Rain"};ch

ar*Temperature_kind[3]={"Hot","Mild","Cool"};char*Hu

midity_kind[2]={"High","Normal"};

char*Wind_kind[2]={"Weak","Strong"};
/*inti_Exampple[14][5]={0,0,0,0,1,
                  0,0,0,1,1,
                   1,0,0,1,0,
                  2,1,0,0,0,
                  2,2,1,0,0,
                  2,2,1,1,1,
                   1,2,1,1,0,
                  0,1,0,0,1,
                  0,2,1,0,0,
                  2,1,1,0,0,
                  0,1,1,1,0,
                   1,1,1,1,0,
                   1,1,1,0,0,
                  2,1,0,0,1
                  };*/
voidtreelists( treeT) ;
voidInitAttr( Attributes& attr_ link, char* Attributes_ kind[ ] , intAttr_ kind[ ] ) ; voidInitLink( link&



L,char*Examples[][COL]);



voidID3(tree&T,linkL,linkTarget_Attr,Attributesattr);voi

dPN_Num(linkL,int&positve,int&negative);
```

```
          doubleGain( intpositive, intnegative, char* atrribute, linkL, Attributesattr_ L) ;
voidmain( )
{
    linkLL,p;
    Attributesattr_L,q;
    treeT;
    T=newTNode;
    T- > firstchild= T- > nextsibling= NULL;
    strcpy(T->weight,"");
    strcpy(T->data,"");
    attr_L=newAttrNode;
    attr_L->next=NULL;
    LL =newLNode;L

    L->next=NULL;
    //成功成立两个链表
    InitLink(LL,Examples);
    InitAttr(attr_L,Attributes_kind,Attr_kind);
    ID3(T,LL,NULL,attr_L);
    cout<<"决议树以广义表形式输出以下：  "<<endl;
    treelists(T);//以广义表的形式输出树
//     cout<<Gain(9,5,"OutLook",LL,attr_L)<<endl;
    cout<<endl;
}
//以广义表的形式输出树
voidtreelists( treeT)
{
    treep;
    if(!T)
        return;
    cout<<"{"<<T->weight<<"}";
    cout<<T->data;
    p=T->firstchild;
    if(p)
    {
        cout<<"(";
        while(p)
        {
            treelists(p);
            p=p->nextsibling;
            if(p)cout<<',';
        }
        cout<<")";
    }
}
voidInitAttr( Attributes& attr_ link, char* Attributes_ kind[ ] , intAttr_ kind[ ] )
{
    Attributesp;
    for(inti=0;i<4;i++)
    {
        p= newAttrNode;
        p- > next= NULL;
        strcpy( p- > attributes, Attributes_ kind[ i] ) ;
        p- > attr_ Num= Attr_ kind[ i] ;
        p- > next= attr_ link- > next;
        attr_ link- > next= p;
    }
}
voidInitLink( link& LL, char* Examples[ ] [ COL] )
{
    linkp;
    for(inti=0;i<ROW;i++)
    {
        p= newLNode;
        p- > next= NULL;
        strcpy(p->OutLook,Examples[i][0]);
        strcpy(p->Temperature,Examples[i][1]);
        strcpy(p->Humidity,Examples[i][2]);
        strcpy(p->Wind,Examples[i][3]);
        strcpy(p->PlayTennis,Examples[i][4]);
        p- > next= LL- > next;
        LL- > next= p;
    }
}
voidPN_ Num( linkL, int& positve, int& negative)
{
    positve=0;
    negative=0;
    linkp;
    p=L->next;
    while(p)
    {
        if( strcmp( p- > PlayTennis, " No" ) = = 0 )
            negative+ + ;
        elseif( strcmp( p- > PlayTennis, " Yes" ) = = 0 )
            positve+ + ;
        p= p- > next;
    }
}
//计算信息增益
//linkL:样本会合 S
//attr_L：属性会合
doubleGain( intpositive, intnegative, char* atrribute, linkL, Attributesattr_ L)
{
    intatrr_kinds;//每个属性中的值的个数
    Attributesp= attr_ L- > next;
    linkq= L- > next;
    intattr_th=0;//第几个属性
    while( p)
    {
        if( strcmp( p- > attributes, atrribute) = = 0 )
        {
            atrr_ kinds= p- > attr_ Num;
            break;
        }
        p= p- > next;
        attr_th+ + ;
    }
    doubleentropy, gain= 0 ;
    doublep1 = 1 . 0 * positive/ ( positive+ negative) ;
    doublep2 = 1 . 0 * negative/ ( positive+ negative) ;
    entropy=-p1*log(p1)/log2-p2*log(p2)/log2;//会合熵
```

```
gain= entropy;
//获得每个属性值在训练样本中出现的个数
//获得每个属性值所对应的正例和反例的个数
//申明一个 3*atrr_kinds 的数组
intkinds=newint*[3];
for(intj=0;j<3;j++)
{
    kinds[j]=newint[atrr_kinds];//保留每个属性值在训练样本中出现的个数
}
//初始化
for(intj=0;j<3;j++)
{
    for(inti=0;i<atrr_kinds;i++)
    {
        kinds[j][i]=0;
    }
}
while(q)
{
    if(strcmp("OutLook",atrribute)==0)
    {
        for(inti=0;i<atrr_kinds;i++)
        {
            if(strcmp(q->OutLook,OutLook_kind[i])==0)
            {
                kinds[0][i]++;
                if(strcmp(q->PlayTennis,"Yes")==0)
                    kinds[1][i]++;
                else
                    kinds[2][i]++;
            }
        }
    }
    elseif(strcmp("Temperature",atrribute)==0)
    {
        for(inti=0;i<atrr_kinds;i++)
        {
            if(strcmp(q->Temperature,Temperature_kind[i])==0)
            {
                kinds[0][i]++;
                if(strcmp(q->PlayTennis,"Yes")==0)
                    kinds[1][i]++;
                else
                    kinds[2][i]++;
            }
        }
    }
    elseif( strcmp( " Humidity" , atrribute) = = 0 )
    {
        for( inti= 0 ; i< atrr_ kinds; i+ + )
        {
            if( strcmp( q->Humidity, Humidity_ kind[i] )==0 )
            {
                kinds[0][i]++;
                if( strcmp( q->PlayTennis,"Yes" )==0 )
                    kinds[1][i]++;//
                else
                    kinds[2][i]++;
            }
        }
    }
    elseif( strcmp( " Wind" , atrribute) = = 0 )
    {
        for( inti= 0 ; i< atrr_ kinds; i+ + )
        {
            if( strcmp( q->Wind, Wind_ kind[i] )==0 )
            {
                kinds[0][i]++;
                if( strcmp( q->PlayTennis,"Yes" )==0 )
                    kinds[1][i]++;
                else
                    kinds[2][i]++;
            }
        }
    }
    q= q-> next;
}
//计算信息增益
double* gain_ kind= newdouble[ atrr_ kinds] ;
intpositive_ kind= 0 , negative_ kind= 0 ;
```