一种adaboost多类分类算法Matlab实现

一、adaboost算法简介

Adaboost算法的主要思想是给定一个训练集(x1,y1),…,(xm,ym),其中xi属于某个域或者实例空间X，yi=-1或者+1。初始化时Adaboost指定训练集上的分布为1/m,并按照该分布调用弱学习器对训练集上的分布，并按照该分布调用弱学习器对训练集进行训练，每次训练后，根据训练结果更新训练集上的分布，并按照新的样本分布进行训练。反复迭代T轮，最终得到一个估计序列h1,..,hT,每个估计都具有一定的权重，最终的估计H是采用权重投票方式获得。Adaboost算法的伪代码如 图1所示。

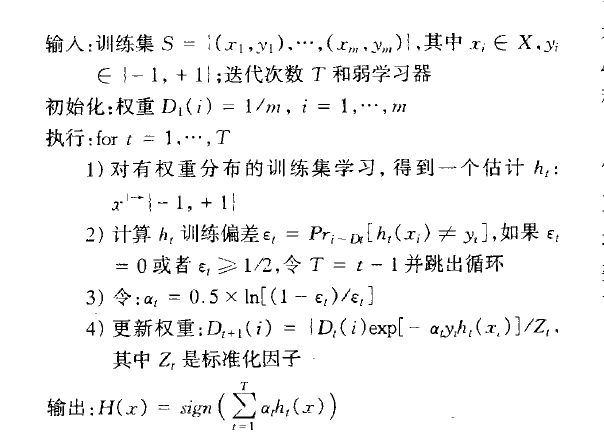


图1、Adaboost算法

二、多类问题

从上面的流程可以看出，Adaboost算法是针对二类问题的。但是我们面对的问题很多都是不是简单的非0即1，而是多类问题。常见的就是解决方法，就是把多类问题转换成二类问题。用的比较多就是两种组合方法，OAA和OAO，我这里就是采用对这种方法的结合，实现adaboost算法对多类问题的分类。

目前需要对7类问题进行分类，依次编号：0、1、2、3、4、5、6。

特征向量 28个。

样本总数 840个；

OAA分类器的个数 7 个

OAO分类器的个数 7（7-1）/2 = 21个。

弱分类器的个数 K= 10；

弱分类用BP神经网络

算法的思路：

Step1、把数据分成训练集和测试集

Step 2、训练OAA、OAO分类器；

Step3、保存相应的分类器和投票权重；

Step4、测试样本，预测所以OAA分类器的权重；

Step5、选择OAA预测值中最大的两个

Step6、选用OAO分类器对选取预测权重最大的两个类进行预测；

Step7、输出测试结果；

注：为了统一，在训练OAO分类器是，把类别序列在前的类为正样本，输出+1，类别序列号在后面的为负样本，输出为-1。

测试强分类器的识别率为：0.93左右。

三、小结

其实这个主要的思想就是用差异的样本和差异的分类器，组合较好的分类器器，提升样本的准确性和鲁邦性。

在对多类问题识别中。个人觉得本算法还是有一定的缺陷，如果从OAA分类器预测值中选取的最大的两个值，不包含在我们正确的样本类别中，那么我们最后的OAO决策就没有意义。因此对这个OAA预测值的组合也是一个比较关键的技术，也请大家有好的组合决策方法进行交流。

我对Adaboost算法理解的比较肤浅，还请大家指点。

如有需要请加QQ272122352，验证多类分类问题

谢谢合作，非诚勿扰！

附件1、OAO/OAA分类器训练

clear

clc

tic,

%加载数据

load Sheet1 Sheet1

%删除无效数据

while(1)

if(Sheet1(end,1) == 0)

Sheet1(end,:) = [];

else

break;

end

end

%分解输入输出

Input = Sheet1(:,2:end)';

Output = Sheet1(:,1)';

N = max(max(Output));

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%生成一对多（one-against-all)分类器 % %

%类别标志,把样本组合成一对多的分类器。 %

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

NetName\_OAA = {};

NetName\_OAA\_C = 0;

for Flag = 0 : N

%输入变量归一化处理

[inputn,inputps] = mapminmax(Input);

input\_train = [];

input\_test = [];

output\_train = [];

output\_test = [];

%生成测试集和训练集

%训练样本和测试样本比例

P = 0.8;

for NN = 0 : N

Index = find(Output == NN);

Long = length(Index);

% 80%的训练、20%测试

input\_train = [input\_train inputn(:,Index(1):Index(1) + floor(Long \* P))];

input\_test = [input\_test inputn(:,Index(1) + floor(Long \*P) + 1 : Index(end))];

output\_train = [output\_train Output(:,Index(1):Index(1) + floor(Long \* P))];

output\_test = [output\_test Output(:,Index(1) + floor(Long \* P) + 1 : Index(end))];

end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

Long1 = length(output\_train);

Long2 = length(output\_test);

for x = 1 : Long1

if(output\_train(x) == Flag)

output\_train(x) = 1;

else

output\_train(x) = -1;

end

end

for y = 1 : Long2

if(output\_test(y) == Flag)

output\_test(y) = 1;

else

output\_test(y) = -1;

end

end

%训练弱分类器

[Sizem,Sizen] = size(input\_train);

D(1,:)=ones(1,Sizen)/Sizen;

%弱分类器个数

K = 10;

[Out,Outn] = mapminmax(output\_train);

for KK = 1 : K

%错误率

error(KK) = 0;

Flag\_error = -1;

% NetName = strcat(num2str(Flag),'\_',num2str(KK),'Class');

% net = NetName;

%BP神经网络构建

net=newff(input\_train,output\_train,9);

net.trainParam.epochs = 50;

net.trainParam.lr = 0.1;

net.trainParam.goal = 0.00004;

%BP神经网络训练

net=train(net,input\_train,output\_train);

%训练集数据预测

an1 = sim(net,input\_train);

test\_simu1(KK,:)=mapminmax('reverse',an1,Outn);

% %测试数据预测

% an = sim(net,input\_test);

% test\_simu(KK,:) = mapminmax('reverse',an,Outn);

%

%统计训练集输出效果

kk1 = find(test\_simu1(KK,:)>0);

kk2 = find(test\_simu1(KK,:)<0);

aa(kk1) = 1;

aa(kk2) = -1;

%统计错误样本数

for j=1:Sizen

if aa(j)~=output\_train(j);

error(KK)=error(KK)+D(KK,j);

end

end

if error(KK) == 0

Flag\_error = KK;

break;

end

%弱分类器i权重

at(KK)=0.5\*log((1-error(KK))/error(KK));

%更新D值

for j=1:Sizen

D(KK+1,j)=D(KK,j)\*exp(-at(KK)\*aa(j)\*test\_simu1(KK,j));

end

%D值归一化

Dsum=sum(D(KK+1,:));

D(KK+1,:)=D(KK+1,:)/Dsum;

% NetName = strcat(num2str(Flag),'\_',num2str(KK),'Class');

% save(NetName,'net');

NetName\_OAA{Flag + 1,KK} = net;

NetName\_OAA\_C(Flag +1,KK) = at(KK);

% 强分类器分类结果

% OutPut = sign(at\*test\_simu);

end

if Flag\_error ~= -1

NetName\_OAA\_C(Flag + 1,Flag\_error : 10 ) = 0;

%如果第一次就为零、那么该分类器就是强分类器

if Flag\_error == 1

NetName\_OAA\_C(Flag + 1,1) = 1;

NetName\_OAA{Flag + 1,1} = net;

end

end

end

%保存分类器

save NetName\_OAA NetName\_OAA

save NetName\_OAA\_C NetName\_OAA\_C

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%生产一对一的分类器 %

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

tic,

%一共有N\*（N-1）/2个分类器

NetName\_OAO = {};

%训练样本和测试样本比例

P = 0.8;

input\_train = [];

input\_test = [];

output\_train = [];

output\_test = [];

for NN = 0 : N

Index = find(Output == NN);

Long = length(Index);

% 80%的训练、20%测试

input\_train = [input\_train inputn(:,Index(1):Index(1) + floor(Long \* P))];

input\_test = [input\_test inputn(:,Index(1) + floor(Long \*P) + 1 : Index(end))];

output\_train = [output\_train Output(:,Index(1):Index(1) + floor(Long \* P))];

output\_test = [output\_test Output(:,Index(1) + floor(Long \* P) + 1 : Index(end))];

end

for MM = 0 : N

for NN = 0 : N

if(MM < NN)

%选取样本

%样本输入

%样本输出

%为了保存一致性，两类分类器的组合，类别序号在前的为+1，类别序号在后的为-1

%如{3,4} ->{+1,-1};{MM,NN} ={+1,-1};

% Input\_OAO =

Long1 = length(output\_train);

XX = 1;

input\_train\_OAO = [];

output\_train\_OAO = [];

for x = 1 : Long1

if(output\_train(x) == MM)

output\_train\_OAO(XX) = 1;

input\_train\_OAO(:,XX) = input\_train(:,x);

XX = XX + 1;

end

if(output\_train(x) == NN)

output\_train\_OAO(XX) = -1;

input\_train\_OAO(:,XX) = input\_train(:,x);

XX = XX + 1;

end

end

%BP神经网络构建

net=newff(input\_train\_OAO, output\_train\_OAO,9);

net.trainParam.epochs = 50;

net.trainParam.lr = 0.1;

net.trainParam.goal = 0.00004;

%BP神经网络训练

net=train(net,input\_train\_OAO, output\_train\_OAO);

NetName\_OAO{MM + 1,NN + 1} = net;

else

NetName\_OAO{MM + 1,NN + 1} = 0;

end

end

end

toc

save NetName\_OAO NetName\_OAO

附件2：测试

clear

clc

%加载分类器

load NetName\_OAA

load NetName\_OAO

load NetName\_OAA\_C

%加载测试数据

load Sheet1

%删除无效数据

while(1)

if(Sheet1(end,1) == 0)

Sheet1(end,:) = [];

else

break;

end

end

%分解输入输出

Input = Sheet1(:,2:end)';

Output = Sheet1(:,1)';

%生产测试集、

[inputn,inputps] = mapminmax(Input);

input\_train = [];

input\_test = [];

output\_train = [];

output\_test = [];

N = max(max(Output));

K = 10;

P = 0.8;

for NN = 0 : N

Index = find(Output == NN);

Long = length(Index);

% 80%的训练、20%测试

input\_train = [input\_train inputn(:,Index(1):Index(1) + floor(Long \* P))];

input\_test = [input\_test inputn(:,Index(1) + floor(Long \*P) + 1 : Index(end))];

output\_train = [output\_train Output(:,Index(1):Index(1) + floor(Long \* P))];

output\_test = [output\_test Output(:,Index(1) + floor(Long \* P) + 1 : Index(end))];

end

tic,

%测试

%% 第一级分类

long = length(input\_test);

Temp\_Out\_OAA = 0;

for X = 1 : long

% N 类 K 个 OAA 分类器决策

test\_simul(1:N+1) = 0;

for NN = 1 : N + 1

Temp\_test\_simul = 0;

for KK = 1 : K

net = NetName\_OAA{NN,KK};

coef = NetName\_OAA\_C(NN,KK);

if coef == 0

% Flag\_0 = KK;

break

end

Temp\_test\_simul = Temp\_test\_simul + coef \* sim(net,input\_test(:,X));

end

test\_simul(NN) = Temp\_test\_simul;

end

Temp\_Out\_OAA(X,1 : N + 1) = test\_simul;

end

toc

%%第二级分类

Ouput\_Class = [];

for X = 1 : long

Temp\_OAO = Temp\_Out\_OAA(X,:);

[p1,p2] = sort(Temp\_OAO,2);

Max1 = p1(end);

Max2 = p1(end - 1);

Class\_M1 = p2(end);

Class\_M2 = p2(end - 1);

%加载单OAO分类器

Change\_Flag = 0;

if Class\_M1 < Class\_M2

M1 = Class\_M1;

M2 = Class\_M2;

Change\_Flag = 0;

else

M1 = Class\_M2;

M2 = Class\_M1;

Change\_Flag = 1;

end

net = NetName\_OAO{M1,M2};

%预测输出

out\_oao = sim(net,input\_test(:,X));

%类别判断

if out\_oao > 0

if Change\_Flag == 0

Output\_Class(X) = Class\_M1 - 1;

else

Output\_Class(X) = Class\_M2 - 1;

end

else

if Change\_Flag == 0

Output\_Class(X) = Class\_M2 - 1;

else

Output\_Class(X) = Class\_M1 - 1;

end

end

end

%% 统计分类效果

Error\_Num = 0;

for X = 1 : long

if Output\_Class(X) ~= output\_test(X)

Error\_Num = Error\_Num + 1;

end

end

disp('强分类器识别率');

Error\_Num/X

2104年9月