大数据机器学习第四章作业

姓名：方颖　　学院：资源与环境学院　　学号：201628006010063

1、用 Matlab 或其它语言，选择下面 4 中算法的任意 2 个算法，实现 Bagging、Boostrap、Boosting、Adaboost 算法，并给出数据试验结果。

（1）Boosting

给定表中所示训练数据，假设弱分类器由 x<v 或 x>v 产生，其阈值 v 使该分类器在训练数据集上分类误差率最低。使用 boosting 算法学习一个强分类器。

（boosting 为课件上算法）

序列	1	2	3	4	5	6	7	8	9	10
x	0	1	2	3	4	5	6	7	8	9
y	1	1	1	-1	-1	-1	1	1	1	-1

Python 编程如下：

```
#coding:UTF-8
"""
Boosting 算法
"""
import os,sys
import numpy as np
import random

#载入数据
def loadData():
    dataMat=[0,1,2,3,4,5,6,7,8,9]
    classLables=[1,1,1,-1,-1,-1,1,1,1,-1]
    return dataMat,classLables

#基分类器训练算法
def weakClf(traindata,classlables):   #参数为训练数据
    weakC={}
    m=np.shape(traindata)[0]       #训练样本总数
    err1=[0 for i in range(m)]    #错分数目
    err2=[0 for i in range(m)]
    classow1=[[0 for i in range(m)] for j in range(m)]    #学习得到的分类数据
    classow2=[[0 for i in range(m)] for j in range(m)]

    #假设弱分类器由 x>thre 或 x<thre 产生
```

```
    #阈值 thre 使得该分类器在训练数据集上分类误差率最低
    for i in range(m):
        if (i==0):
            for p in range(m):
                classow1[i][p]=-1
                classow2[i][p]=1
        else:
            for p in range(i):
                classow1[i][p]=1
                classow2[i][p]=-1
            for p in range(m-i):
                classow1[i][i+p]=-1
                classow2[i][i+p]=1
        #计算误分类样本数目
        for k in range(m):
            if (classow1[i][k]!=classlables[k]):
                err1[i]+=1
        for k in range(m):
            if (classow2[i][k]!=classlables[k]):
                err2[i]+=1


    #确定误差率，并且找出误差率最小的阈值及弱分类器，计算此弱分类器的系数
    if (min(err1)<min(err2)):
        err=min(err1)
        thre=err1.index(min(err1))
        classow=classow1[thre]
    else:
        err=min(err2)
        thre=err2.index(min(err2))
        classow=classow2[thre]

    #弱分类器,thre:阈值 V  classresult：弱分类器分类结果
    weakC["thre"]=thre
    weakC["classresult"]=classow

    print "threshould:",thre
    print "misclassified number:",err
    print "result of weakclassfier:",classow

    return weakC,err,classow

#弱分类器（weakclassifier）对某一数据（data）分类所得的结果（标签）
def ClassifyByWeak(data,weakclassfier):
```

```
    if(data<weakclassfier["thre"]):
        classlable=weakclassfier["classresult"][0]
    else:
        classlable=weakclassfier["classresult"][-1]
    return classlable

#Boosting 算法
#参数为（训练数据，训练数据标签，第一个数据集的样本容量，第二个数据集的样本容量）
def Boosting(dataArr,classLables,n1,n2):
    weakClfArr=[]                    #弱分类器集合
    n=np.shape(dataArr)[0]      #原始样本总量
    dataArr3=[]
    classlables3=[]
    dataArr4=[]
    classlables4=[]
    budataArr=[]
    buclasslables=[]

    #输出原始训练数据
    print "the training  data(D):",dataArr
    print "lables of training data:",classLables,"\n"

    #抽取第一个数据集 D1，从原始样本中不放回地随机选取 n1 个样本点
    dataArrNum1=random.sample(range(n),n1)
    dataArr1=[0 for i in range(n1)]
    classlables1=[0 for i in range(n1)]
    for (i,j) in zip(range(n1),dataArrNum1):
        dataArr1[i]=dataArr[j]
        classlables1[i]=classLables[j]

    #输出第一个数据集 D1
    print "the first dataset(D1):",dataArr1
    print "lables of the first dataset(D1):",classlables1

    #根据第一个数据集训练第一个弱分类器
    weakC1,err1,classow1 = weakClf(dataArr1,classlables1)
    weakClfArr.append(weakC1)
    print "the first weak classifier:",weakC1,"\n"

    #抽取 n1 个样本后的剩余样本
    dataArrNum2=list(set(range(n))-set(dataArrNum1))
    dataArr2=[0 for i in range(n-n1)]
    classlables2=[0 for i in range(n-n1)]
    for (i,j) in zip(range(n-n1),dataArrNum2):
```

```
        dataArr2[i]=dataArr[j]
        classlables2[i]=classLables[j]

    #抽取第二个数据集 D2
    while len(dataArr3)<n2:      #保证生成 n2 个样本的数据集 D2
        r=random.random()    #生成随机数，作为判定投掷硬币的正反面

        #当硬币为正面时，如果是正面就选取 D 中剩余样本点一个一个送到 C 中进行分
        #类，遇到第一个被错分的样本加入集合 D2 中
        if r<0.5:
            #判定当剩余数据集不空时，利用第一个弱分类器对剩余样本进行分类
            if len(dataArr2)>0:
                for i in range(len(dataArr2)) :  #对于剩余样本中每个样本
                    classiedlable=ClassifyByWeak(dataArr2[i],weakC1)

                    #若是错分样本则加入到数据集 D2 中
                    #并且将其从剩余样本数据集中删除
                    #跳出，掷下一次色子，删除是因为不放回
                    if classiedlable!=classlables2[i]:
                        dataArr3.append(dataArr2[i])
                        classlables3.append(classlables2[i])
                        dataArr2.remove(dataArr2[i])
                        classlables2.remove(classlables2[i])
                        break

                    #若是分对的样本，则将其加入到备份数据集中
                    #并且将其从剩余样本数据集中删除
                    #跳出，掷下一次色子，删除是因为不放回
                    else:
                        budataArr.append(dataArr2[i])
                        buclasslables.append(classlables2[i])
                        dataArr2.remove(dataArr2[i])
                        classlables2.remove(classlables2[i])
                        break

        #当硬币为反面时，就选取一个被 C1 正确分类的样本点加入集合 D2 中
        else:
            #判断当剩余数据集不空时，找到一个分对的样本加入 D2 中
            if len(dataArr2)>0:
                for i in range(len(dataArr2)):

                    #若备份数据不空，则在备份数据中拿
                    #加入数据集 D2，并将其从剩余样本中删除
                    #跳出，掷下一次色子，删除是因为不放回
```

```
                    if len(budataArr)!=0:
                        dataArr3.append(budataArr[0])
                        classlables3.append(buclasslables[0])
                        budataArr.remove(budataArr[0])
                        buclasslables.remove(buclasslables[0])
                        break

                    #否则，用弱分类器找出分对的样本
                    #加入数据集 D2，并将其从剩余样本中删除
                    #跳出，掷下一次色子，删除是因为不放回
                    elif ClassifyByWeak(dataArr2[i],weakC1)==classlables2[i]:
                        dataArr3.append(dataArr2[i])
                        classlables3.append(classlables2[i])
                        dataArr2.remove(dataArr2[i])
                        classlables2.remove(classlables2[i])
                        break

            #当剩余数据集中空时，只能在备份数据集中找分对的样本
            else:
                dataArr3.append(budataArr[0])
                classlables3.append(buclasslables[0])
                budataArr.remove(budataArr[0])
                buclasslables.remove(buclasslables[0])

#输出第二个数据集 D2
print "the second dataset(D2):",dataArr3
print "lables of the second dataset(D2):",classlables3

#利用第二个数据集训练第二个弱分类器
weakC2,err2,classow2=weakClf(dataArr3,classlables3)
weakClfArr.append(weakC2)
print "the second weak classifier:",weakC2,"\n"

#剩余样本
for i in range(len(budataArr)):
    dataArr2.append(budataArr[i])
    classlables2.append(buclasslables[i])

#抽取第三个数据集 D3
#如 C1 和 C2 分类结果不同，就把该样本加入集合 D3
for i in range(len(dataArr2)):
    classiedlable1=ClassifyByWeak(dataArr2[i],weakC1)
    classiedlable2=ClassifyByWeak(dataArr2[i],weakC2)
    if classiedlable2!=classiedlable1:
```

```
                dataArr4.append(dataArr2[i])
                classlables4.append(classlables2[i])

        #若 C1 和 C2 恰好可以正确分类剩余样本，则无数据集 D3，也就没有第三个弱分类器
        if len(dataArr4)==0:
            print "there is no third dataset!"
            print "there is no third weak classifier!","\n"

        #若 C1 和 C2 都不能正确分类剩余样本，则将这类样本加入数据集 D3
        else:
            print "the third dataset(D3):",dataArr4
            print "lables of the third dataset(D4):",classlables4

            #用第三个数据集 D3 训练第三个弱分类器 C3
            weakC3,err3,classow3=weakClf(dataArr4,classlables4)
            weakClfArr.append(weakC3)
            print "the third weak classifier:",weakC3,"\n"

        print "the final classifier:",weakClfArr,"\n"
        return weakClfArr

#用 Boosting 算法对数据分类
def ClassifyByBoosting(data,weakClfArr):
        if ClassifyByWeak(data,weakClfArr[0])==ClassifyByWeak(data,weakClfArr[1]):
            classlableB=ClassifyByWeak(data,weakClfArr[0])
        else:
            classlableB=ClassifyByWeak(data,weakClfArr[2])
        return classlableB

#主函数
if __name__=='__main__':
        datMat,classLables=loadData()
        weakClfArr=Boosting(datMat,classLables,4,3)
        classlable=ClassifyByBoosting(8,weakClfArr)
        print "Boosting classlable of 8 is:",classlable
```

运行结果如下：

```
L:\研一下学习\大数据机器学习\作业>python boosting.py
the training data(D): [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
lables of training data: [1, 1, 1, -1, -1, -1, 1, 1, 1, -1]

the first dataset(D1): [1, 7, 4, 3]
lables of the first dataset(D1): [1, 1, -1, -1]
threshould: 2
misclassified number: 0
result of weakclassfier: [1, 1, -1, -1]
the first weak classifier: {'classresult': [1, 1, -1, -1], 'thre': 2}

the second dataset(D2): [0, 5, 2]
lables of the second dataset(D2): [1, -1, 1]
threshould: 0
misclassified number: 1
result of weakclassfier: [1, 1, 1]
the second weak classifier: {'classresult': [1, 1, 1], 'thre': 0}

the third dataset(D3): [6, 8, 9]
lables of the third dataset(D4): [1, 1, -1]
threshould: 2
misclassified number: 0
result of weakclassfier: [1, 1, -1]
the third weak classifier: {'classresult': [1, 1, -1], 'thre': 2}

the final classifier: [{'classresult': [1, 1, -1, -1], 'thre': 2}, {'classresult': [1, 1, 1], 'thre': 0}, {'classresult'
: [1, 1, -1], 'thre': 2}]

Boosting classlable of 8 is: -1
```

两次运行结果不相同

```
L:\研一下学习\大数据机器学习\作业>python boosting.py
the training data(D): [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
lables of training data: [1, 1, 1, -1, -1, -1, 1, 1, 1, -1]

the first dataset(D1): [8, 0, 3, 1]
lables of the first dataset(D1): [1, 1, -1, 1]
threshould: 0
misclassified number: 1
result of weakclassfier: [1, 1, 1, 1]
the first weak classifier: {'classresult': [1, 1, 1, 1], 'thre': 0}

the second dataset(D2): [2, 6, 7]
lables of the second dataset(D2): [1, 1, 1]
threshould: 0
misclassified number: 0
result of weakclassfier: [1, 1, 1]
the second weak classifier: {'classresult': [1, 1, 1], 'thre': 0}

there is no third dataset!
there is no third weak classifier!

the final classifier: [{'classresult': [1, 1, 1, 1], 'thre': 0}, {'classresult': [1, 1, 1], 'thre': 0}]

Boosting classlable of 8 is: 1
```

（2）Adaboost

给定表中所示训练数据，假设弱分类器由 x<v 或 x>v 产生，其阈值 v 使该分类器在训练数据集上分类误差率最低。使用 Adaboost 算法学习一个强分类器。

序列	1	2	3	4	5	6	7	8	9	10
x	0	1	2	3	4	5	6	7	8	9
y	1	1	1	-1	-1	-1	1	1	1	-1

Python 编程如下：

```
#载入数据
def loadData():
    dataMat=[0,1,2,3,4,5,6,7,8,9]
    classLables=[1,1,1,-1,-1,-1,1,1,1,-1]
     return dataMat,classLables

#生成弱分类器,给定数据，根据 x>thre 或 x<thre 产生，其阈值使得该分类器
#在训练数据上的分类误差率最小
def weakClf(traindata,classlables,weight):   #参数为训练数据和权值分布
    weakC={}
    m=np.shape(traindata)[0]     #训练样本总数
    err1=[0,0,0,0,0,0,0,0,0,0]      #分类误差率
    err2=[0,0,0,0,0,0,0,0,0,0]
    classow1=np.zeros((m,10))     #学习得到的分类数据
    classow2=np.zeros((m,10))

    for i in range(m):
        thre=traindata[i]
        if (thre==0):
            for p in range(m):
                classow1[i][p]=-1
                classow2[i][p]=1
        else:
            for p in range(thre):
                classow1[i][p]=1
                classow2[i][p]=-1
            for p in range(m-thre):
                classow1[i][thre+p]=-1
                classow2[i][thre+p]=1
        for k in range(m):
            if (classow1[i][k]!=classlables[k]):
                err1[i]+=weight[k]
```

```
            for k in range(m):
                if (classow2[i][k]!=classlables[k]):
                    err2[i]+=weight[k]

        #确定误差率，并且找出误差率最小的阈值及弱分类器
        if (min(err1)<min(err2)):
            err=min(err1)
            thre=err1.index(min(err1))
            way=1          #当 way=1 时，表示 x<thre 时分为 1，x>thre 时分为-1
            classow=classow1[thre]
        else:
            err=min(err2)
            thre=err2.index(min(err2))
            way=-1      #当 way=-1 时，表示 x<thre 时分为-1，x>thre 时分为 1
            classow=classow2[thre]

        #弱分类器
        weakC["thre"]=thre
        weakC["classresult"]=classow

        print "threshould:",thre
        print "minerror:",err
        print "class of weakclassfier:",classow

        return weakC,err,classow

#adaboost 算法
def adaboost(dataArr,classLables,t=20):
        weakClfArr=[]            #弱分类器数组
        n=np.shape(dataArr)[0]       #样本的数目
        D=np.ones(n)/n               #创建权值分布矩阵大小为 n*1，值全为 1/n
        aggclass=np.zeros(n)

        #输出原数据
        print "original data:",dataArr
        print "original lable:",classLables,"\n"

        #对于 i=1,2....t
        for i in range(t):
            #弱分类器分类
            weakC,err,classow = weakClf(dataArr,classLables,D)
            print "current weight:",D

            #计算弱分类器权值
```

```
            cindex=0.5*(np.log((1-err)/max(err,1e-16)))
            weakC["index"]=cindex
            weakClfArr.append(weakC)
            print "index of this weak classifier :",cindex

            #更新权值分布
            expin=np.multiply(-1*cindex*np.array(classLables),np.array(classow))
            D=np.multiply(D,np.exp(expin))
            D=D/D.sum()
            print "new weight:",D

            #计算弱分类器加权累计值，f(x)=a1*G1(x)+a2*G2(x)....
            aggclass+=cindex*classow
            print "aggregrat value:",aggclass.T

          #计算误差
aggclassErr=np.multiply(np.sign(aggclass)!=np.array(classLables),np.array(np.ones(n)))
            errClassNum=aggclassErr.sum()
            print "misclassified data:",aggclassErr
            print"error number:",errClassNum,"\n"

            #设置程序停止
            if errClassNum == 0:
                  break
       return weakClfArr

#主函数
if __name__=='__main__':
      datMat,classLables=loadData()
      adaboost(datMat,classLables,6)
```

运行结果如下：

```
L:\研一下学习\大数据机器学习\作业>python weakclassfier.py
original data: [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
original lable: [1, 1, 1, -1, -1, -1, 1, 1, 1, -1]

threshould: 3
minerror: 0.3
class of weakclassfier: [ 1.  1.  1. -1. -1. -1. -1. -1. -1. -1.]
current weight: [ 0.1  0.1  0.1  0.1  0.1  0.1  0.1  0.1  0.1  0.1]
index of this weak classifier : 0.423648930194
new weight: [ 0.07142857  0.07142857  0.07142857  0.07142857  0.07142857  0.07142857
  0.16666667  0.16666667  0.16666667  0.07142857]
aggregrat value: [ 0.42364893  0.42364893  0.42364893 -0.42364893 -0.42364893 -0.42364893
 -0.42364893 -0.42364893 -0.42364893 -0.42364893]
misclassified data: [ 0.  0.  0.  0.  0.  0.  1.  1.  1.  0.]
error number: 3.0

threshould: 9
minerror: 0.214285714286
class of weakclassfier: [ 1.  1.  1.  1.  1.  1.  1.  1.  1. -1.]
current weight: [ 0.07142857  0.07142857  0.07142857  0.07142857  0.07142857  0.07142857
  0.16666667  0.16666667  0.16666667  0.07142857]
index of this weak classifier : 0.649641492065
new weight: [ 0.04545455  0.04545455  0.04545455  0.16666667  0.16666667  0.16666667
  0.10606061  0.10606061  0.10606061  0.04545455]
aggregrat value: [ 1.07329042  1.07329042  1.07329042  0.22599256  0.22599256  0.22599256
  0.22599256  0.22599256  0.22599256 -1.07329042]
misclassified data: [ 0.  0.  0.  1.  1.  1.  0.  0.  0.  0.]
error number: 3.0

threshould: 6
minerror: 0.181818181818
class of weakclassfier: [-1. -1. -1. -1. -1. -1.  1.  1.  1.  1.]
current weight: [ 0.04545455  0.04545455  0.04545455  0.16666667  0.16666667  0.16666667
  0.10606061  0.10606061  0.10606061  0.04545455]
index of this weak classifier : 0.752038698388
new weight: [ 0.125       0.125       0.125       0.10185185  0.10185185  0.10185185
  0.06481481  0.06481481  0.06481481  0.125     ]
aggregrat value: [ 0.32125172  0.32125172  0.32125172 -0.52604614 -0.52604614 -0.52604614
  0.97803126  0.97803126  0.97803126 -0.32125172]
misclassified data: [ 0.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
error number: 0.0
```

注：threshould：指训练得到的弱分类器的阈值 v

minerror：指弱分类器的分类误差率

class of weakclassfier：指弱分类器的分类结果

current weight：指当前的权值分布

index of this weak classifier：指当前弱分类器的投票权值

new weight：指更新的权值分布

aggregrat value：指由当前弱分类器构成的强分类器的分类结果

misclassified data：指分错的数据

error number：指错分的数据的个数

最终的强分类器就是把以上几个弱分类器的加权和。