实验八　高级分类 Adaboost 算法

【实验目的】

1. 了解 Anaconda 和 python 的使用与设置。
2. 掌握高级分类 Adaboost 算法的基本原理和实现方法。

【实验学时】

建议 2 学时

【实验环境配置】

1、 Windows 环境

2、 Anaconda

3、 Pandas

【实验原理】

数据挖掘中高级分类 Adaboost 算法的实现：

1、Pandas 数据导入

2、高级分类 Adaboost 算法

3、sk-learn 库的高级分类 Adaboost 算法模块使用

【实验步骤】

1. 打开 Anaconda 的 jupyter notebook，创建实验八，明确标题和步骤。
2. 导入 pandas 库和 os 库，从 horseColicTraining2.txt 数据文件中，导入实验训练数据；从 horseColicTest2.txt 数据文件中，导入实验训练数据。

   （注：可能涉及的函数 os.path.join, os.walk, pandas.read_csv）
3. 对导入的实验数据，根据《机器学习实战》教程第七章，完成高级分类 Adaboost 算法，并完成对于 horse Colic 的分类识别。
4. 对导入的实验数据，根据 sklearn 的 Adaboost 算法模块 sklearn.ensemble.AdaBoostClassifier，完成对于 horse colic 数据的分类识别。

(1) 导入 numpy 包，编写加载数据函数，读取简单的测试数据用于初步测试：

```
def loadSimpData():
    """
    测试数据
    Returns:
        datMat       feature对应的数据集
        classLabels  feature对应的分类标签
    """
    datMat = array([[1. , 2.1],
                    [2. , 1.1],
                    [1.3, 1. ],
                    [1. , 1. ],
                    [2. , 1. ]])
    classLabels = [1.0, 1.0, -1.0, -1.0, 1.0]
    return datMat, classLabels
# 读取数据:
datMat, classLabels = loadSimpData()
```

图 1 测试数据

(2) 有了数据，接下来就可以通过构建多个函数来建立单层决策树。第一个函数将用于测试是否有某个值小于或者大于我们正在测试的阈值。第二个函数则更加复杂一些，它会在一个加权数据集中循环，并找到具有最低错误率的单层决策树。伪代码：

'''

将最小错误率 minError 设为+∞

对数据集中的每一个特征（第一层循环）：

对每个步长（第二层循环）：

对每个不等号（第三层循环）：

建立一棵单层决策树并利用加权数据集对它进行测试

如果错误率低于 minError，则将当前单层决策树设为最佳单层决策树

返回最佳单层决策树

'''

开始构造单决策树生成函数：

1、stumpClassify()函数将数据集，按照 feature 列的 value 进行 二分法切分比较来赋值分类。

2、buildStump()得到决策树的模型：先转换数据，再将数据初始化，最小误差为无穷大，循环所有的 feature 列，将列切分成 若干份，每一段以最左边的点作为分类节点，最后返回分类器的结果、错误率、训练后的结果集。

```
datMat, classLabels = loadSimpData()
buildStump(datMat, classLabels, D)
```

```
({'dim': 0, 'thresh': 1.3, 'ineq': 'lt'}, matrix([[0.2]]), array([[-1.],
       [ 1.],
       [-1.],
       [-1.],
       [ 1.]]))
```

图 2 构造结果

3、最后完成整个 Adaboost 算法的实现，构造基于单层决策树的 AdaBoost 训练过程，得到弱分类器的集合和预测的分类结果值。

(3) 测试算法：基于 AdaBoost 的分类。现在，需要做的就只是将弱分类器的训练过程从程序中抽出来，然后应用到马病的实例上去。每个弱分类器的结果以其对应的 alpha 值作为权重。所有这些弱分类器的结果加权求和就得到了最后的结果。

```
datArr, labelArr = loadSimpData()
classifierArr = adaBoostTrainDS(datArr, labelArr, 30)
adaClassify([0, 0], classifierArr)
adaClassify([[5, 5], [0, 0]], classifierArr)
```

```
  total error=0.2
  total error=0.2
  total error=0.0
  aggClassEst: [[-0.69314718]]
  aggClassEst: [[-1.66610226]]
  aggClassEst: [[ 0.69314718]
   [-0.69314718]]
  aggClassEst: [[ 1.66610226]
   [-1.66610226]]

matrix([[ 1.],
        [-1.]])
```

图 3 测试结果

(4) 预测患有疝气病的马的存活问题，这里的数据包括 368 个样本和 28 个特征，疝气病是描述马胃肠痛的术语，然而，这种病并不一定源自马的胃肠问题，其他问题也可能引发疝气病，该数据集中包含了医院检测马疝气病的一些指标，有的指标比较主观，有的指标难以测量，例如马的疼痛级别。另外，除了部分指标主观和难以测量之外，该数据还存在一个问题，数据集中有 30%的值是缺失的。

1、# 收集数据：提供的文本文件

训练数据：horseColicTraining.txt

测试数据：horseColicTest.txt

2、准备数据：确保类别标签是+1 和-1，而非 1 和 0。编写自适应数据加载函数，得到数据矩阵和标签向量。

3、加载训练集，得到错误率；加载测试集，得到预测结果；最后统计到错误总数。

```
#加载训练集，得到错误率
datArr, labelArr = loadDataSet('horseColicTraining2.txt')
classifierArr = adaBoostTrainDS(datArr, labelArr, 10)
#加载测试集，得到预测结果
testArr, testLabelArr = loadDataSet('horseColicTest2.txt')
prediction10 = adaClassify(testArr, classifierArr)
#得到错误总数
errArr = mat(ones((67,1)))
errArr[prediction10 != mat(testLabelArr) .T] .sum()
```

```
total error=0.2842809364548495
total error=0.2842809364548495
total error=0.24749163879598662
total error=0.24749163879598662
aggClassEst: [[ 0.46166238]
 [ 0.46166238]
 [-0.46166238]
 [-0.46166238]
 [ 0.46166238]]
18.0
```

图 4 训练结果

4、分析数据：统计分析。训练算法：在数据上，利用 adaBoostTrainDS() 函数训练出一系列的分类器。完成 ROC 曲线的绘制及 AUC 计算函数。最后调用该算法，观察马患病例子上的错误率。

```
datArr, labelArr = loadDataSet('horseColicTraining2.txt')
classifierArray, aggClassEst = adaBoostTrainDS(datArr, labelArr, 10)
```

```
total error=0.2842809364548495
total error=0.2842809364548495
total error=0.24749163879598662
total error=0.24749163879598662
total error=0.25418060200668896
total error=0.2408026755852843
total error=0.2408026755852843
total error=0.22073578595317725
total error=0.24749163879598662
total error=0.23076923076923078
```

图 5 错误率

5、最后根据最终预测结果的权重值和始数据的分类结果集绘制图像：

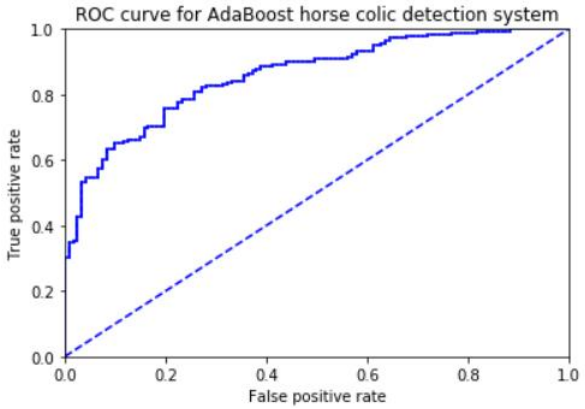

图 6 预测图象