深蓝学院
shenlanxueyuan.com

编程实践：基于决策树和 C4.5 算法进行二分类

主讲人 张阳阳

张阳阳，清华大学博士生，研究方向：人工智能、机器学习、深度学习。擅长 Python, MATLAB，Tensorflow, Pytorch 框架，熟悉神经网络，支持向量机，朴素贝叶斯，决策树，聚类分析等AI 算法和编程实践。曾作为 Python 课程讲师，有多年教学经验。

目录

- 二分类

outlook	temperature	humidity	windy	result
sunny	hot	high	false	N
sunny	hot	high	true	N
overcast	hot	high	false	Y
rain	mild	high	false	Y
rain	cool	normal	false	Y
rain	cool	normal	true	N
overcast	cool	normal	true	Y

问题背景

深蓝学院 shenlanxueyuan.com

- ## 训练集

outlook	temperature	humidity	windy	result
sunny	hot	high	false	N
sunny	hot	high	true	N
overcast	hot	high	false	Y
rain	mild	high	false	Y
rain	cool	normal	false	Y
rain	cool	normal	true	N
overcast	cool	normal	true	Y

- 测试集

outlook	temperature	humidity	windy
sunny	mild	high	false
sunny	cool	normal	false
rain	mild	normal	false
sunny	mild	normal	true
overcast	mild	high	true
overcast	hot	normal	false
rain	mild	high	true

编程实践

深蓝学院 shenlanxueyuan.com

```python
# -*- coding: utf-8 -*-
from math import log
import operator
import treePlotter

def calcShannonEnt(dataSet):
    """
    输入：数据集
    输出：数据集的香农熵
    描述：计算给定数据集的香农熵；熵越大，数据集的混乱程度越大
    """
    numEntries = len(dataSet)
    labelCounts = {}
    for featVec in dataSet:
        currentLabel = featVec[-1]
        if currentLabel not in labelCounts.keys():
            labelCounts[currentLabel] = 0
        labelCounts[currentLabel] += 1
    shannonEnt = 0.0
    for key in labelCounts:
        prob = float(labelCounts[key])/numEntries
        shannonEnt -= prob * log(prob, 2)
    return shannonEnt
```

编程实践


```python
def splitDataSet(dataSet, axis, value):
    """
    输入：数据集，选择维度，选择值
    输出：划分数据集
    描述：按照给定特征划分数据集；去除选择维度中等于选择值的项
    """
    retDataSet = []
    for featVec in dataSet:
        if featVec[axis] == value:
            reduceFeatVec = featVec[:axis]
            reduceFeatVec.extend(featVec[axis+1:])
            retDataSet.append(reduceFeatVec)
    return retDataSet
```

编程实践

深蓝学院
shenlanxueyuan.com

```python
def chooseBestFeatureToSplit(dataSet):
    """
    输入：数据集
    输出：最好的划分维度
    描述：选择最好的数据集划分维度
    """
    numFeatures = len(dataSet[0]) - 1
    baseEntropy = calcShannonEnt(dataSet)
    bestInfoGainRatio = 0.0
    bestFeature = -1
    for i in range(numFeatures):
        featList = [example[i] for example in dataSet]
        uniqueVals = set(featList)
        newEntropy = 0.0
        splitInfo = 0.0
        for value in uniqueVals:
            subDataSet = splitDataSet(dataSet, i, value)
            prob = len(subDataSet)/float(len(dataSet))
            newEntropy += prob * calcShannonEnt(subDataSet)
            splitInfo += -prob * log(prob, 2)
        infoGain = baseEntropy - newEntropy
        if (splitInfo == 0): # fix the overflow bug
            continue
        infoGainRatio = infoGain / splitInfo
        if (infoGainRatio > bestInfoGainRatio):
            bestInfoGainRatio = infoGainRatio
            bestFeature = i
    return bestFeature
```

编程实践

深蓝学院
shenlanxueyuan.com

```python
def majorityCnt(classList):
    """
    输入：分类类别列表
    输出：子节点的分类
    描述：数据集已经处理了所有属性，但是类标签依然不是唯一的，
          采用多数判决的方法决定该子节点的分类
    """
    classCount = {}
    for vote in classList:
        if vote not in classCount.keys():
            classCount[vote] = 0
        classCount[vote] += 1
    sortedClassCount = sorted(classCount.iteritems(), key=operator.itemgetter(1), reversed=True)
    return sortedClassCount[0][0]
```

编程实践


```python
def createTree(dataSet, labels):
    """
    输入：数据集，特征标签
    输出：决策树
    描述：递归构建决策树，利用上述的函数
    """
    classList = [example[-1] for example in dataSet]
    if classList.count(classList[0]) == len(classList):
        # 类别完全相同，停止划分
        return classList[0]
    if len(dataSet[0]) == 1:
        # 遍历完所有特征时返回出现次数最多的
        return majorityCnt(classList)
    bestFeat = chooseBestFeatureToSplit(dataSet)
    bestFeatLabel = labels[bestFeat]
    myTree = {bestFeatLabel:{}}
    del(labels[bestFeat])
    # 得到列表包括节点所有的属性值
    featValues = [example[bestFeat] for example in dataSet]
    uniqueVals = set(featValues)
    for value in uniqueVals:
        subLabels = labels[:]
        myTree[bestFeatLabel][value] = createTree(splitDataSet(dataSet, bestFeat, value), subLabels)
    return myTree
```

编程实践


```python
def classify(inputTree, featLabels, testVec):
    """
    输入：决策树，分类标签，测试数据
    输出：决策结果
    描述：跑决策树
    """
    firstStr = list(inputTree.keys())[0]
    secondDict = inputTree[firstStr]
    featIndex = featLabels.index(firstStr)
    for key in secondDict.keys():
        if testVec[featIndex] == key:
            if type(secondDict[key]).__name__ == 'dict':
                classLabel = classify(secondDict[key], featLabels, testVec)
            else:
                classLabel = secondDict[key]
    return classLabel
```

编程实践

深蓝学院 shenlanxueyuan.com

```python
def classifyAll(inputTree, featLabels, testDataSet):
    """
    输入：决策树，分类标签，测试数据集
    输出：决策结果
    描述：跑决策树
    """
    classLabelAll = []
    for testVec in testDataSet:
        classLabelAll.append(classify(inputTree, featLabels, testVec))
    return classLabelAll
```

编程实践


```python
def storeTree(inputTree, filename):
    """
    输入：决策树，保存文件路径
    输出：
    描述：保存决策树到文件
    """
    import pickle
    fw = open(filename, 'wb')
    pickle.dump(inputTree, fw)
    fw.close()
```

编程实践

```python
def grabTree(filename):
    """
    输入：文件路径名
    输出：决策树
    描述：从文件读取决策树
    """
    import pickle
    fr = open(filename, 'rb')
    return pickle.load(fr)
```

编程实践

```python
def createDataSet():
    """
    outlook->  0: sunny | 1: overcast | 2: rain
    temperature-> 0: hot | 1: mild | 2: cool
    humidity-> 0: high | 1: normal
    windy-> 0: false | 1: true
    """
    dataSet = [[0, 0, 0, 0, 'N'],
               [0, 0, 0, 1, 'N'],
               [1, 0, 0, 0, 'Y'],
               [2, 1, 0, 0, 'Y'],
               [2, 2, 1, 0, 'Y'],
               [2, 2, 1, 1, 'N'],
               [1, 2, 1, 1, 'Y']]
    labels = ['outlook', 'temperature', 'humidity', 'windy']
    return dataSet, labels
```

编程实践

```python
def createTestSet():
    """
    outlook->  0: sunny | 1: overcast | 2: rain
    temperature-> 0: hot | 1: mild | 2: cool
    humidity-> 0: high | 1: normal
    windy-> 0: false | 1: true
    """
    testSet = [[0, 1, 0, 0],
               [0, 2, 1, 0],
               [2, 1, 1, 0],
               [0, 1, 1, 1],
               [1, 1, 0, 1],
               [1, 0, 1, 0],
               [2, 1, 0, 1]]
    return testSet
```

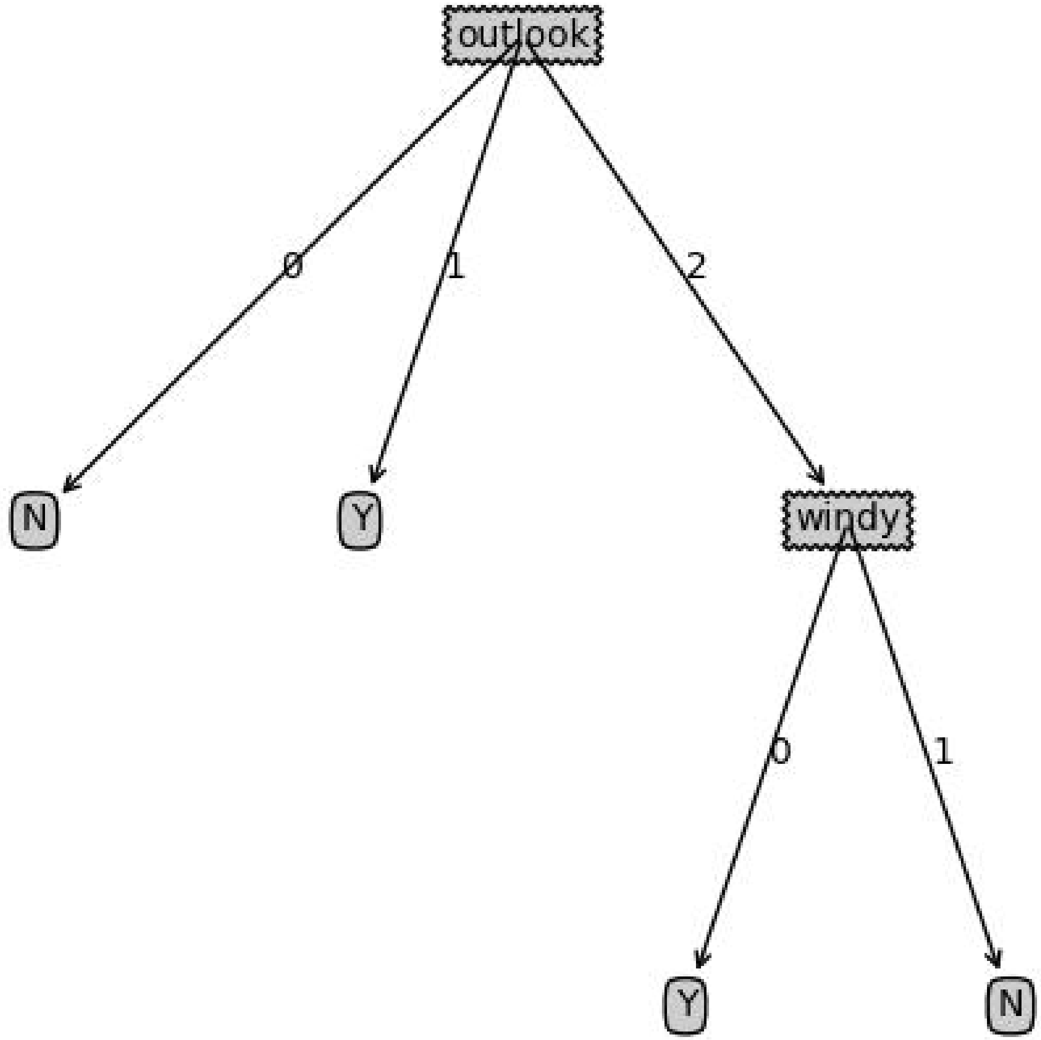
编程实践

深蓝学院
shenlanxueyuan.com

```python
def main():
    dataSet, labels = createDataSet()
    labels_tmp = labels[:] # 拷贝，createTree会改变labels
    desicionTree = createTree(dataSet, labels_tmp)
    #storeTree(desicionTree, 'classifierStorage.txt')
    #desicionTree = grabTree('classifierStorage.txt')
    print('desicionTree:\n', desicionTree)
    treePlotter.createPlot(desicionTree)
    testSet = createTestSet()
    print('classifyResult:\n', classifyAll(desicionTree, labels, testSet))

if __name__ == '__main__':
    main()
```

编程实践



课后作业

这些年来很多人因为癌症逝世，其中不乏娱乐圈中的一线明星们，而乳腺癌也成了很多女星不幸离世的原因。这种病症已经引起了来自社会的关注，定期检查变得很有必要。Wisconsin医学院的william H.Wolberg博士提供乳腺癌数据样本。所欲数据来自真实临床案例，每个案例有9个属性。

数据来源：http://archive.ics.uci.edu/ml/machine-learning-databases/breast-cancer-wisconsin/

课后作业

深蓝学院
shenlanxueyuan.com

 # Attribute Domain

-- ---

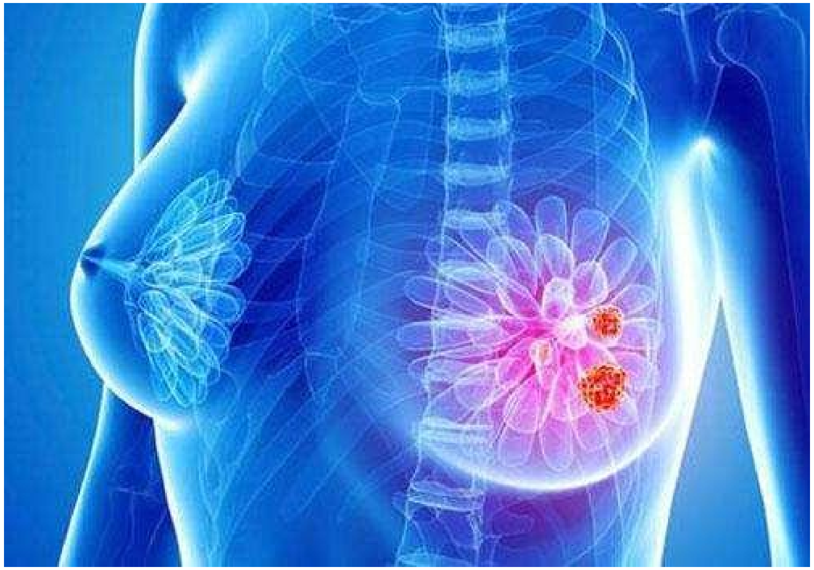
1. Sample code number id number
2. Clump Thickness 1 - 10
3. Uniformity of Cell Size 1 - 10
4. Uniformity of Cell Shape 1 - 10
5. Marginal Adhesion 1 - 10
6. Single Epithelial Cell Size 1 - 10
7. Bare Nuclei 1 - 10
8. Bland Chromatin 1 - 10
9. Normal Nucleoli 1 - 10
10. Mitoses 1 - 10

11. Class: (2 for benign, 4 for malignant)

Class distribution:

Benign: 458 (65.5%)

Malignant: 241 (34.5%)



课后作业

深蓝学院
shenlanxueyuan.com

请你使用附件中提供的数据：Breastdata.txt

使用决策树和 C4.5 算法，选择 80% 的数据作为训练集，剩余 80% 的数据作为测试集，然后预测下面这位患者【5,7,6,10,8,4,6,5,4】（9个属性的检测结果，分别用1-10表示）是患良性还是恶性肿瘤。提示：重难点是原始数据的读取，切割，后面稍微调整。

数据来源：http://archive.ics.uci.edu/ml/machine-learning-databases/breast-cancer-wisconsin/

Q&A

感谢各位聆听

请批评指正