Assignment 2 Report for JavaScript

潘瑞峰　1641478

程序运行方法

1. 用浏览器打开 decision_tree.html
2. 点击选择文件，选择该目录下的 car_train.data 数据集
3. 点击 train 按钮
4. 用户可以手动输入一行数据，然后点击 test input data 按钮进行决策
5. 也可以上传整个测试文件，对多行数据进行预测，上传方法：a. 点击 Test With File 下的选择文件，选择 car_test.data 数据集，点击 test 按钮
6. 最终的效果图如下

train data set: 选择文件 car_train.data trian

Test with One Row

buying: vhigh
maint: vhigh
doors: 3
persons: more
lug_boot: high
safety: low
test input data
predict result: unacc

Test With File

选择文件 car_test.data test

testNum:292 matchedNum:260 accuray:0.8904109589041096

input	target	predicted target
vhigh,vhigh,3,more,big,low,unacc	unacc	unacc
vhigh,vhigh,3,more,big,med,unacc	unacc	unacc
vhigh,vhigh,3,more,big,high,unacc	unacc	unacc
vhigh,vhigh,4,2,small,low,unacc	unacc	unacc
vhigh,vhigh,4,2,small,med,unacc	unacc	unacc
vhigh,vhigh,4,2,small,high,unacc	unacc	unacc
vhigh,vhigh,4,2,med,low,unacc	unacc	unacc
vhigh,vhigh,4,2,med,med,unacc	unacc	unacc
vhigh,vhigh,4,2,med,high,unacc	unacc	unacc
vhigh,vhigh,4,2,big,low,unacc	unacc	unacc
vhigh,vhigh,4,2,big,med,unacc	unacc	unacc
vhigh,vhigh,4,2,big,high,unacc	unacc	unacc
vhigh,vhigh,4,4,small,low,unacc	unacc	unacc
vhigh,vhigh,4,4,small,med,unacc	unacc	unacc
vhigh,vhigh,4,4,small,high,unacc	unacc	unacc
vhigh,vhigh,4,4,med,low,unacc	unacc	unacc
vhigh,vhigh,4,4,med,med,unacc	unacc	unacc
vhigh,vhigh,4,4,med,high,unacc	unacc	unacc
vhigh,vhigh,4,4,big,low,unacc	unacc	unacc

说明

1. 选择 JavaScript 的原因

   由于写之前参考过 python 代码，所以用 python 写的时候会向参考过的代码靠近，所以用 JavaScript 重写。由于 JS 本身是脚本语言，不同于编程语言，所以在代码中使用的数据结构与计算方法都会和编程语言不同。缺点是脚本语言运行速度慢。

2. 选择的数据集

   数据集：Car Evaluation Data Set

   属性：

   buying: vhigh, high, med, low.
   maint: vhigh, high, med, low.
   doors: 2, 3, 4, 5more.
   persons: 2, 4, more.
   lug_boot: small, med, big.
   safety: low, med, high.

   分类值:

   unacc, acc, good, vgood

3. 数据预处理

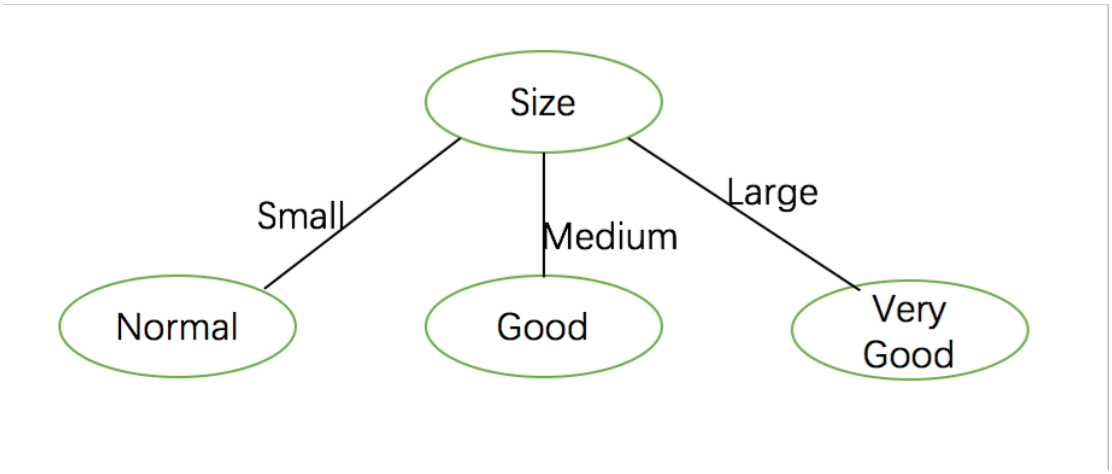
   由于本数据符合要求，所以未做处理。

   原始数据有 1728 条数据，取其中的 1436 条作为训练集，292 条作为测试集

4. 决策树的结构

   决策树是以 JSON Object 的形式存在的。

   假设有如下结构的树



   那么对应的 JSON Object 就如下

```
{
    “Size”: {
                “Small”: “Normal”,
                “Medium”: “Good”,
                “Large”: “Very Good”
            }
}
```

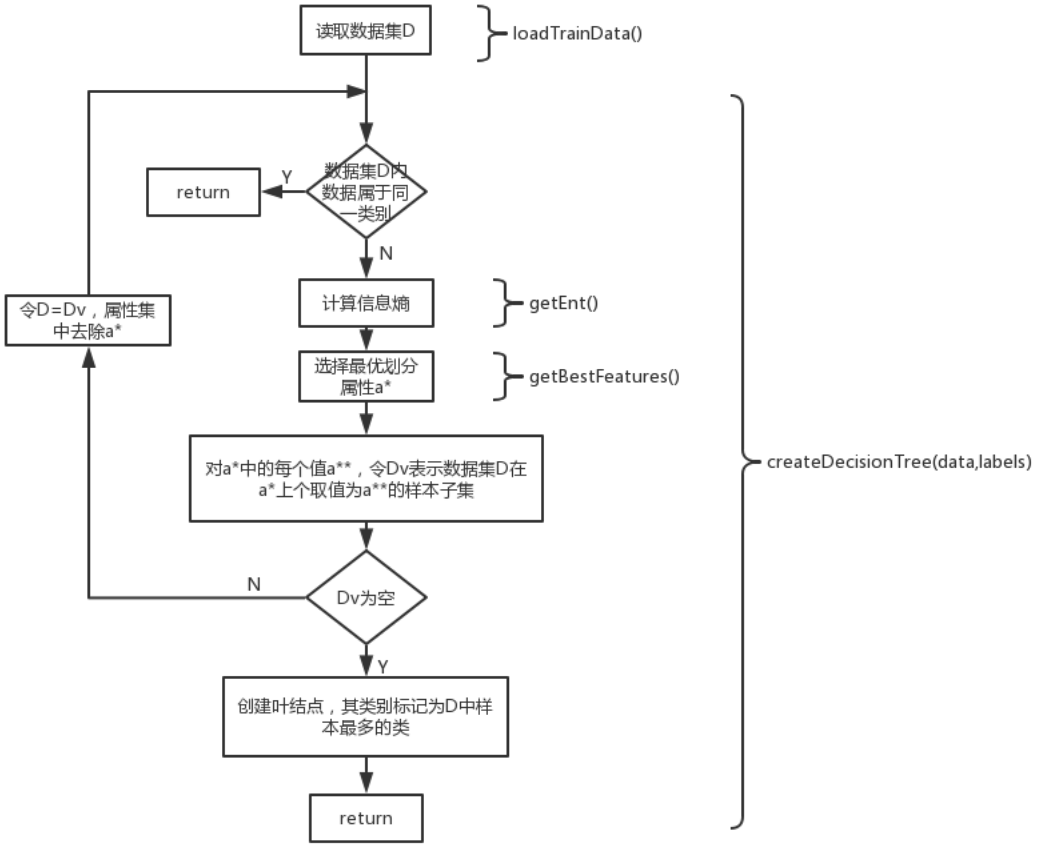
叶子结点的值就是决策结果。

叶子结点也可能是一颗子树

本程序使用循环的方式查找叶子节点。通过判断子树是 JSON 对象还是字符串来决定迭代次数，具体可看 decision_tree.js 文件中第 61 行的 predict 函数。

5. 程序设计

程序设计图如下



注：本图为本人于 2016-11-17 在 processon 上所画，链接地址如下：
https://www.processon.com/view/link/584998eae4b0d594ec648b4b

6. 函数说明(仅与算法有关的函数)

getEnt(): 计算信息熵，返回浮点数
getDistinctCount(pos)：计算 pos 特征的所有属性的分布，返回 JSON Object
getBestFeatures():通过计算最优信息熵，来寻找最优划分属性
subData(index,feature):返回去除最优划分属性后的数据集
countTarget():判断当前数据集是否属于同一类别，返回结果布尔值
deleteElement(index):返回去除最优划分属性后的分类标签
getIndex(value): 获取 value 在数据集中的位置
createTree(subData,input):根据当前子数据集和输入类别，递归创建树，返回 JSON 对象或者 String
predict(testData): 使用循环的方式查找叶子节点，如果叶子结点是一个 undefined 的结果，说明找不到决策值，返回 unknow，否则返回相应的决策值

7. 文件说明

car_train.data 训练数据集
car_test.data 测试数据集
origin_data 原始数据集所在的文件夹
decision_tree.html 本程序的界面
decision_tree.js 本程序的代码文件
readme.pdf 本程序说明文件

8. 不足与提升空间

本程序将所有特征定义为离散的，所以对于一些连续的数值型特征不友好。提升空间是可以通过用划分数值区间的方式来对连续的值进行分类。