

机器学习作业-决策树

软件 51 庞建业 2151601012

<https://github.com/sherjy/Notes-RL>

本次主要针对决策树进行算法和编程的学习和复习

环境：

Ubuntu16.04

Atom+Anaconda3.6

首先跑两个 **scikit-learn** 决策树 Demo

(1) .这个例子是我写在当初在学习吴恩达的视频课中最开始讲的决策树例子，虽然很简单，但是比较容易理解决策树在做什么。见文件夹中 **test.py**

```
1 from sklearn import tree
2 features=[[140,0],[130,0],[150,1],[170,1]]
3 labels=["apple","apple","orange","orange"]
4 clf=tree.DecisionTreeClassifier()
5 clf=clf.fit(features,labels)
6 print (clf.predict([[150,1]]))
7
```

Atom Runner: Decision Tree.py

Atom Runner: test.py

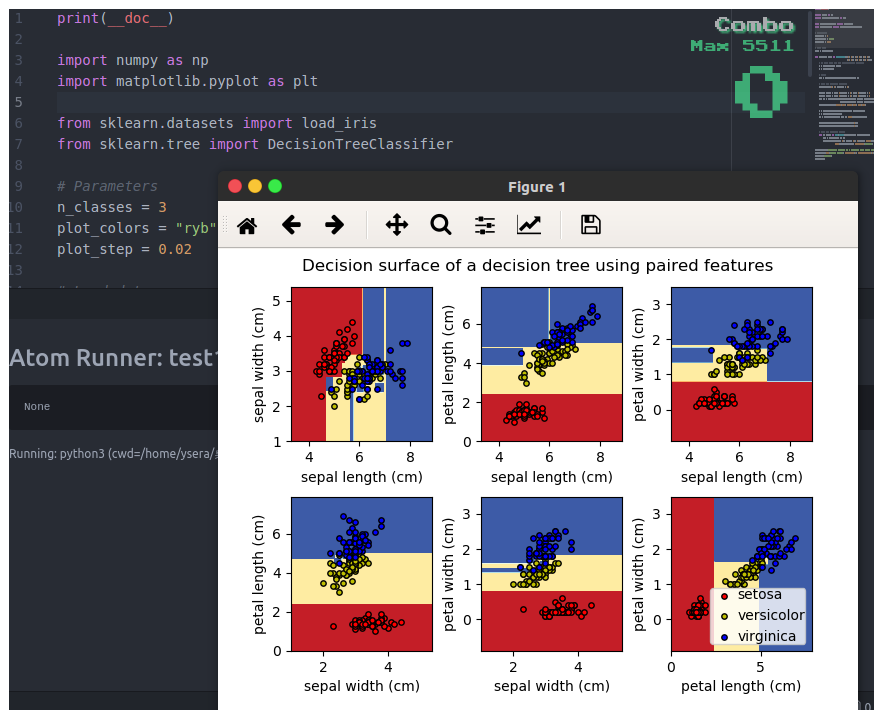
['orange']

Running: python3 (cwd=/home/ysera/桌面/test.py pid=24042). Exited with code=0 in 0.508 seconds.

(2) .这个例子是我跑的 sklearn 官网的可视化例子 见文件夹中 **test1.py**

源数据集是 sklearn 的 iris 数据集 在此附上代码和数据集链接

http://scikit-learn.org/stable/auto_examples/tree/plot_iris.html#sphx-glr-auto-examples-tree-plot-iris-py



补充：

iris 数据集的中文名是安德森鸢

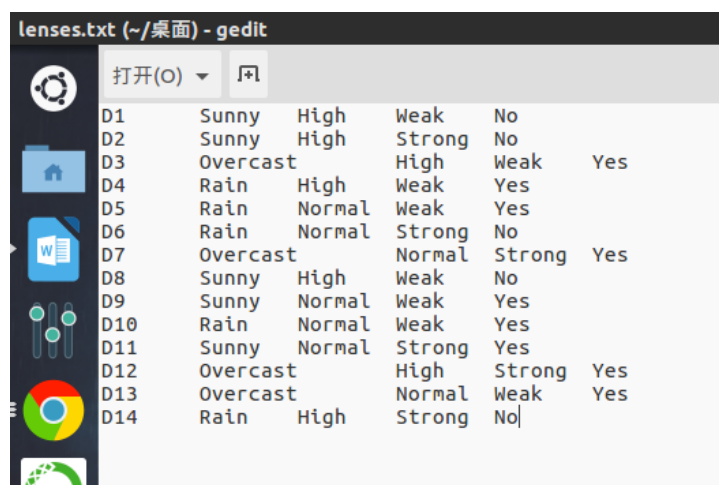
尾花卉数据集，英文全称是 Anderson's Iris dataset。iris 包含 150 个样本，对应数据集的每行数据。每行数据包含每个样本的四个特征和样本的类别信息，所以 iris 数据集是一个 150 行 5 列的二维表。通俗地说，iris 数据集是用来给花做分类的数据集，每个样本包含了花萼长度、花萼宽度、花瓣长度、花瓣宽度四个特征（前 4 列），我们需要建立一个分类器，分类器可以通过样本的四个特征来判断样本属于山鸢尾、变色鸢尾还是维吉尼亚鸢尾（这三个名词都是花的品种）。

作业实现

Training examples: 9 yes / 5 no				
Day	Outlook	Humidity	Wind	Play
D1	Sunny	High	Weak	No
D2	Sunny	High	Strong	No
D3	Overcast	High	Weak	Yes
D4	Rain	High	Weak	Yes
D5	Rain	Normal	Weak	Yes
D6	Rain	Normal	Strong	No
D7	Overcast	Normal	Strong	Yes
D8	Sunny	High	Weak	No
D9	Sunny	Normal	Weak	Yes
D10	Rain	Normal	Weak	Yes
D11	Sunny	Normal	Strong	Yes
D12	Overcast	High	Strong	Yes
D13	Overcast	Normal	Weak	Yes
D14	Rain	High	Strong	No
New data:				
D15	Rain	High	Weak	?

(1) 基于 sklearn 的 CART 算法实现

先做数据手动导入，将数据先手动写到 txt 中，然后用 python 引入，间隔用 TAB 制表符做，python 中用 \t 识别



D1	Sunny	High	Weak	No
D2	Sunny	High	Strong	No
D3	Overcast	High	Weak	Yes
D4	Rain	High	Weak	Yes
D5	Rain	Normal	Weak	Yes
D6	Rain	Normal	Strong	No
D7	Overcast	Normal	Strong	Yes
D8	Sunny	High	Weak	No
D9	Sunny	Normal	Weak	Yes
D10	Rain	Normal	Weak	Yes
D11	Sunny	Normal	Strong	Yes
D12	Overcast	High	Strong	Yes
D13	Overcast	Normal	Weak	Yes
D14	Rain	High	Strong	No

使用 sklearn 来做分类和预测

用 pandas 做序列化 one-hot 把字符输入编码方便我们可以直接引入 txt 做操作

用 pydotplus 画生成的决策树图写入 tree.pdf

如下图所示 我们可以看到预测结果为 yes

说明

```
#print(clf.predict(['D15','Rain','High','Weak']))
```

```
print(clf.predict([16,0,1,1]))
```

这组预测结果是 yes

备注：值得一提的是，这里我用的 sklearn 实现的，sklearn 的 sklearn.tree.DecisionTreeClassifier 决策树默认使用 CART，是对 CART 的优化处理，CART 算法本身做这里的预测并不稳定，生成的决策树是不稳定树，我进行了参数调整，将树高设置为 8（此处为小样本学习，一般样本的树高设置为 20 较好），

树返回结果趋于 yes 稳定，但是还是有可能会有不稳情况发生。

```
# -*- coding: UTF-8 -*-
from sklearn.preprocessing import LabelEncoder, OneHotEncoder
from sklearn.externals.six import StringIO
from sklearn import tree
import pandas as pd
import numpy as np
import pydotplus

if __name__ == '__main__':
    with open('lenses.txt', 'r') as fr:
        lenses = [inst.strip().split('\t') for inst in fr.readlines()]
        lenses_target = []
        for each in lenses:
            lenses_target.append(each[-1])
        print(lenses_target)
```

Atom Runner: Decision Tree.py

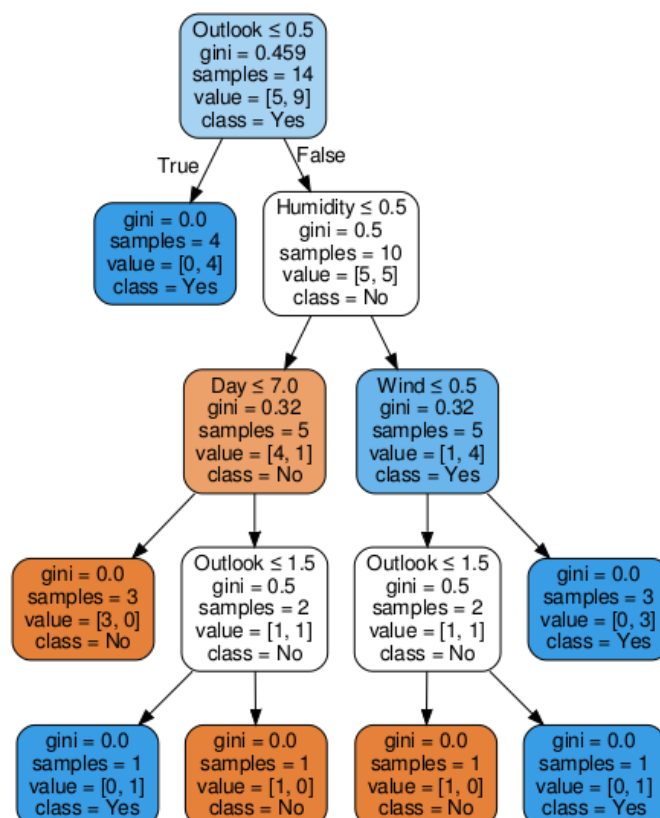
U14	Hign	Rain	Strong
Day	Humidity	Outlook	Wind
0	0	2	1
1	6	2	0
2	7	0	1
3	8	1	1
4	9	1	1
5	10	1	0
6	11	0	0
7	12	2	1
8	13	2	1
9	1	1	1
10	2	2	0
11	3	0	0
12	4	0	1
13	5	1	0

Running: python3 (cwd=/home/ysera/桌面/decisontree/Sklearn-Decision Tree.py pid=26024).. Exited with code=0 in 0.952 seconds.

为了避免过度拟合（over fit）而成为不稳定的树，叶结点需要裁剪（prune）。尽管 CART 提供了自动搜索潜在可能的树分支并根据测试集裁剪回来的策略，但事实上并不足以依赖；统计意义不是决策规则的决定因素，商业理解结合手工裁剪（custom split）可能是更好的选择。另外，少于 100 条数据的叶结点很可能是不稳定的，所以出现这里的不稳定树很正常。

Tree.pdf

Graphviz 可视化



在这里进行 ID3 的算法实现和可视化



Strong \rightarrow No 可以看到这里的预测结果也是 yes

这里进行了 ID4.5 的算法实现和可视化，由于 ID3 的树跟此差不多，所以这里不放出来了
可以看到返回也是 yes，综合 ID3 和 CART 可以知道结果是对的



所以这里直接输入的数据是（1）中的 one-hot 编码，简化了手动输入的过程，效果是正确的

```
def createDataSet():
    dataSet = [[0,0,2,1,'No'],
[6,0,2 ,0,'No'],
[7,0,0,1,'Yes'],
[ 8,0, 1,1,'Yes'],
[9,1,1,1,'Yes'],
[10, 1,1,0,'No'],
[11,1,0, 0,'Yes'],
[12, 0,2,1,'No'],
[13,1,2, 1,'Yes'],
[1,1,1,1,'Yes'],
[2,1,2,0,'Yes'],
[3,0,0,0,'Yes'],
[4,1 ,0,1,'Yes'],
[5,0,1,0,'No']]
    labels = ['Day', 'Outlook', 'Humidity', 'Wind']
    return dataSet, labels
```

决策树优缺点分析

决策树的优点有：

- 能够简单的解释和理解其原理，并且能够以可视化的形式来显示决策树。
- 只需要对数据进行很少的准备工作。其他技术通常需要正则化后的数据，缺失的数据则需要添加假数据或者是删除空值。不过要注意的是这个模块还不支持处理缺失的数据。
- 树的计算代价（例如预测数据）是用于训练的树中，数据点数量的对数。
- 能够同时处理数值型和连续型的数据。其他技术通常只支持一种数据类型。可以查看算法文章以获得更多的信息。
- 可以处理带有多输出的问题。
- 白盒模型。如果在模型中可以观察到特定情况，那么就可以通过简单的布尔逻辑条件来表达出这一情况。相比之下，黑盒模型（例如人造神经网络）的结果可能就不是那么容易就可以表现出来了。
- 可以使用统计测试来验证模型。这使得模型能够更具有可靠性。
- 能够良好的辨别出人造的非法数据。

然后其缺点也有：

- 决策树学习器能够为当前问题创建出一个超复杂的树，但是这个树的泛化能力却很差，这种低效的现象称之为过拟合。可以通过对叶子节点设置最小值或者是给树的深度设置最大值这样的一些"修剪机制"来避免过拟合现象(暂时还不支持直接设置，需要手动修剪)。
- 决策树是不稳定的。可能会因为一个细微的改变而产生一个完全不同的树。不过这个问题可以通过树的整体方法来缓解。(即取多个生成树的均值)
- 能否生成一个最优决策树的问题在于，不管你是需要一个在多方面都是最优的决策树，还是说仅仅只需要一个理论上最优的决策树，他都是一个 NP 完全问题。所以在实际使用中的决策树学习算法都是基于像启发式算法这样的贪婪算法，使得在每个节点上都取其局部最优值。当然，使

用这样的算法并不能保证产生的树是一个全局最优树，所以可以让一组训练器训练出多个树，其中训练器的特征和数据点都使用随机获取。然后再去评估这些树以选出"全局最优树"。

- 有一些概念使用决策树并不能很好的去学习，像异或，奇偶性和复用器等问题。
- 决策树学习器会在偏斜数据集（某一类的数据占了多数）中创建出一个偏斜树。所以严重建议在拟合之前对数据集进行平衡。

参考

[1] ID3

<https://github.com/MachineLearning/MachineLearningID3>

[2] Graphviz

https://blog.csdn.net/john_bh/article/details/78321156?locationNum=9&fps=1

http://www.graphviz.org/Download_windows.php

[3] csv、txt 文件读取 <https://docs.python.org/3.5/library/csv.html?highlight=csv#module-csv>

<https://docs.python.org/2/library/csv.html?highlight=csv#module-csv>

[4] scikit-learn

<http://scikit-learn.org/stable/modules/tree.html#tree-algorithms-id3-c4-5-c5-0-and-cart>

[5] decision tree

<https://www.jianshu.com/p/0724dde480f0>

https://en.wikipedia.org/wiki/Decision_tree_learning

https://en.wikipedia.org/wiki/Predictive_analytics

L. Breiman, J. Friedman, R. Olshen, and C. Stone. Classification and Regression Trees. Wadsworth, Belmont, CA, 1984.

J.R. Quinlan. C4. 5: programs for machine learning. Morgan Kaufmann, 1993.

T. Hastie, R. Tibshirani and J. Friedman. Elements of Statistical Learning, Springer, 2009.