机器学习第一次课后作业

问题描述：

使用Python实现两种概念学习的方法，Find-S和候选消除消除算法，并对题目中所给数据进行处理分析，并学习PlayTinnis这一概念。

基本思路：

主要使用到了两种概念学习方法，Find-S和候选消除算法。这两种算法都是经典的的概念学习方法。Find-S的思路为只通过正例来寻求极大特殊集合，反例直接无视。候选消除算法就是FindS与列表消除法的结合，一个从最大特殊集，一个从最大泛化集，不断的对样本进行筛选，直到得出最后的结果。

解题步骤：

处理流程

首先选择合适的数据结构将题目中所给的数据转换为程序可以直接处理的形式。随后使用程序对数据进行处理。这里选择将数据转换为CSV格式，使用Python的Pandas库进行数据的读取。

算法描述

两种算法的描述分别如下：

Find-S算法：

1. 将h初始化为H中最特殊假设
2. 使用每个正例，将h中属性替换为更一般的约束

3. 输出h

候选消除算法

将G集合初始化为H中极大一般假设 将S集合初始化为H中极大特殊假设 对每个训练例d，进行以下操作：

如果d是一正例

- 从G中移去所有与d不一致的假设
- 对S中每个与d不一致的假设s
- 从S中移去s
- 把s的所有的极小一般化式h加入到S中，其中h满足 h与d一致，而且G的某个成员比h更一般
- 从S中移去所有这样的假设：它比S中另一假设更一般

如果d是一个反例

- 从S中移去所有d不一致的假设
- 对G中每个与d不一致的假设g
- 从G中移去g
- 把g的所有的极小特殊化式h加入到G中，其中h满足 h与d一致，而且S的某个成员比h更特殊
- 从G中移去所有这样的假设：它比G中另一假设更特殊

算法实现

算法的具体实现参见homework1.py文件。

结果与分析

实验内容和步骤

实验在Ubuntu18.04下进行，使用Pycharm编译器，Python版本为3.7.9。实验内容为使用Python实现两种概念学习方法，并使用实现的方法对题目中所给出的数据进行处理，学习目标概念 PlayTinnis。
首先将所给数据转换为CSV数据，并使用Pandas进行读取。在代码中使用Sample类进行表示，Sample包括属性和概念。
对于Find-S算法，使用Find-S类来实现。Find-S类包括训练集数据数量，训练集的属性等变量。通过类中的Run函数进行具体实现。首先将s初始化为最特殊，每个属性对应都为空列表。随着样本的输入，更改S，最后输出S。
对于候选消除算法，使用CandidateElimination类进行实现。初始化S，最特殊成员，每个属性对应都为空，用/代替 。初始化G，最一般成员，每个属性可取任何值，用?代替 。对于每个样本数据根据标签的值，如果是“yes"就选择对G特化，S泛化，如果是0就选择对G泛化，S特化。随着样本的输入，更改S,G，最后输出S，G，之间的所有概念。

实验结果

Find-S

```
S0 [[], [], [], []]
第1个样本 ['Sunny' 'Hot' 'High' False 'No']
S1 [[], [], [], []]
第2个样本 ['Sunny' 'Hot' 'High' True 'No']
S2 [[], [], [], []]
第3个样本 ['Overcast' 'Hot' 'High' False 'Yes']
S3 [['Overcast'], ['Hot'], ['High'], [False]]
第4个样本 ['Rain' 'Mild' 'High' False 'Yes']
S4 [['Overcast', 'Rain'], ['Hot', 'Mild'], ['High'], [False]]
第5个样本 ['Rain' 'Cool' 'Normal' False 'Yes']
S5 [['Overcast', 'Rain'], ['Hot', 'Mild', 'Cool'], ['High', 'Normal'], [False]]
第6个样本 ['Rain' 'Cool' 'Normal' True 'No']
S6 [['Overcast', 'Rain'], ['Hot', 'Mild', 'Cool'], ['High', 'Normal'], [False]]
第7个样本 ['Overcast' 'Cool' 'Normal' True 'Yes']
S7 [['Overcast', 'Rain'], ['Hot', 'Mild', 'Cool'], ['High', 'Normal'], [False, True]]
第8个样本 ['Sunny' 'Mild' 'Higth' False 'No']
```

```
S8 [['Overcast', 'Rain'], ['Hot', 'Mild', 'Cool'], ['High', 'Normal'], [False, True]]
第9个样本 ['Sunny' 'Cool' 'Normal' False 'Yes']
S9 [['Overcast', 'Rain', 'Sunny'], ['Hot', 'Mild', 'Cool'], ['High', 'Normal'], [False, True]]
第10个样本 ['Rain' 'Mild' 'Normal' False 'Yes']
S10 [['Overcast', 'Rain', 'Sunny'], ['Hot', 'Mild', 'Cool'], ['High', 'Normal'], [False, True]]
第11个样本 ['Sunny' 'Mild' 'Normal' True 'Yes']
S11 [['Overcast', 'Rain', 'Sunny'], ['Hot', 'Mild', 'Cool'], ['High', 'Normal'], [False, True]]
第12个样本 ['Overcast' 'Mild' 'High' True 'Yes']
S12 [['Overcast', 'Rain', 'Sunny'], ['Hot', 'Mild', 'Cool'], ['High', 'Normal'], [False, True]]
第13个样本 ['Overcast' 'Hot' 'Normal' False 'Yes']
S13 [['Overcast', 'Rain', 'Sunny'], ['Hot', 'Mild', 'Cool'], ['High', 'Normal'], [False, True]]
第14个样本 ['Rain' 'Mild' 'High' True 'No']
S: [['Overcast', 'Rain', 'Sunny'], ['Hot', 'Mild', 'Cool'], ['High', 'Normal'], [False, True]]
```

候选消除

```
S0 [('/', '/', '/', '/')]
G0 [('?', '?', '?', '?')]
第1个样本 ['Sunny' 'Hot' 'High' False 'No']
S1 [('/', '/', '/', '/')]
G1 [('Rain', '?', '?', '?'), ('?', '?', 'Higth', '?'), ('?', 'Cool', '?', '?'), ('?', '?', '?', True), ('?', '?', 'Normal', '?'), ('Overcast', '?', '?', '?'), ('?', 'Mild', '?', '?')]
第2个样本 ['Sunny' 'Hot' 'High' True 'No']
S2 [('/', '/', '/', '/')]
G2 [('Rain', '?', '?', '?'), ('?', '?', 'Higth', '?'), ('?', 'Cool', '?', '?'), ('?', '?', 'Normal', '?'), ('Overcast', '?', '?', '?'), ('?', 'Mild', '?', '?')]
第3个样本 ['Overcast' 'Hot' 'High' False 'Yes']
```

```
S3 [('Overcast', 'Hot', 'High', False)]
G3 [('Overcast', '?', '?', '?')]
第4个样本 ['Rain' 'Mild' 'High' False 'Yes']
S4 [None]
G4 []
第5个样本 ['Rain' 'Cool' 'Normal' False 'Yes']
S: []
G: []
set()
```

结果分析

Find-S

算法最终将所有的属性全部添加到了概念之中。这是因为Find-S算法使用所有的正例进行概念的学习，没有考虑负例的影响。并且所给的训练数据刚好覆盖了全部的属性，导致了这种情况的出现。

候选消除

最终出现的特化边界与泛化边界都是相同的，应该是样本给的有问题，样本之间主要是维度的差距，同一二维甚至一维上有差距的数据比较少，导致直接被分在了同一个空间。