ID3 project

The goal of this project is to implement a program for inferring imperfect decision trees.

Theory

We discussed in class the classic ID-3 approach for inferring decision trees. The algorithm keeps splitting nodes as long as the nodes have nonzero entropy and features are available. This project's goal is to infer imperfect decision trees with a small number of nodes and with small entropy.

Consider a tree that partitions the instances into k leaves: $S = \{s_1, s_2, \ldots, s_k\}$. The target attribute is T, and the purpose of the tree is to enable us to infer the target attribute. As in the case of ID-3 our goal is to minimize:

$$\mathsf{Entropy}(T|S) = \sum_{j=1}^{k} \mathsf{Prob}(s_j)\mathsf{Entropy}(T|s_j)$$

A greedy approach for inferring such a tree is to iteratively determine what node should be partitioned further. The contribution of a single leaf s_j to the total uncertainty is:

$$E_j = \mathsf{Prob}(s_j)\mathsf{Entropy}(T|s_j) = \frac{|s_j|}{n}\mathsf{Entropy}(T|s_j) \tag{1}$$

where $|s_j|$ is the number of instances in s_j and n is the total number of instances. Therefore, it makes sense to choose s_j with a maximal E_j. A better criterion is obtained using lookahead. Let $A_1, \ldots A_m$ be the attributes. Let T_j be the target attribute restricted to the subset s_j. If we choose s_j, it is to be partitioned using the attribute A_i that minimizes $\mathsf{Entropy}(T_j|A_i)$ computed as:

$$\mathsf{Entropy}(T_j|A_i) = \sum_{l} \frac{|a_l|}{|s_j|}\mathsf{Entropy}(T_j|a_l)$$

The total change in the entropy of the tree is:

$$E_{ji} = \mathsf{Prob}(s_j)(\mathsf{Entropy}(T_j|A_i) - \mathsf{Entropy}(T|s_j)) = -\mathsf{Prob}(s_j)\mathsf{Gain}(s_j, A_i)$$

$$\text{where:} \quad \mathsf{Gain}(s_j, A_i) = \mathsf{Entropy}(s_j) - \mathsf{Entropy}(s_j|A_i)$$

Therefore, we should choose s_j that minimizes E_{ji}, or, equivalently, the one that maximizes

$$F = \max_{i,j} \frac{|s_j|}{n}\mathsf{Gain}(s_j, A_i) \tag{2}$$

Example

	A_1	A_2	A_3	Target attribute
1	R	Y	0	a
2	R	X	0	a
3	R	X	0	b
4	R	X	1	a
5	G	X	1	b
6	G	X	1	a
7	G	X	1	a
8	G	X	1	b
9	B	Y	1	a
10	B	Y	0	b

Starting with the single leaf $\{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$ it must be chosen as the first leaf to split.

$$\begin{aligned} \mathsf{Entropy}(T) &= 0.970951 \\ \mathsf{Entropy}(T|A_1) &= 0.924511 & \mathsf{Gain}(T, A_1) &= 0.0464393 \\ \mathsf{Entropy}(T|A_2) &= 0.965148 & \mathsf{Gain}(T, A2) &= 0.00580215 \\ \mathsf{Entropy}(T|A_3) &= 0.95098 & \mathsf{Gain}(T, A3) &= 0.0199731 \end{aligned}$$

Therefore, the best choice is A_1. This produces the following partition into three leaves:

$$s_1 = \{1, 2, 3, 4\}, \quad s_2 = \{5, 6, 7, 8\} \quad s_3 = \{9, 10\}$$

we have:

$$\begin{aligned} \mathsf{Entropy}(s_1) &= 0.811278 \\ \mathsf{Entropy}(s_1|A_1) &= 0.811278 & \mathsf{Gain}(s_1, A_1) &= 0 \\ \mathsf{Entropy}(s_1|A_2) &= 0.688722 & \mathsf{Gain}(s_1, A_2) &= 0.122556 \\ \mathsf{Entropy}(s_1|A_3) &= 0.688722 & \mathsf{Gain}(s_1, A_3) &= 0.122556 \\ F_1 &= 0.4 \times 0.122556 = 0.0490224 \\ \mathsf{Entropy}(s_2) &= 1 \\ \mathsf{Entropy}(s_2|A_1) &= 1 & \mathsf{Gain}(s_2, A_1) &= 0 \\ \mathsf{Entropy}(s_2|A_2) &= 1 & \mathsf{Gain}(s_2, A_2) &= 0 \\ \mathsf{Entropy}(s_2|A_3) &= 1 & \mathsf{Gain}(s_2, A_3) &= 0 \\ F_2 &= 0.4 \times 0 = 0 \\ \mathsf{Entropy}(s_3) &= 1 \\ \mathsf{Entropy}(s_3|A_1) &= 1 & \mathsf{Gain}(s_3, A_1) &= 0 \\ \mathsf{Entropy}(s_3|A_2) &= 1 & \mathsf{Gain}(s_3, A_2) &= 0 \\ \mathsf{Entropy}(s_3|A_3) &= 0 & \mathsf{Gain}(s_3, A_3) &= 1 \\ F_3 &= 0.2 \times 1 = 0.2 \end{aligned}$$

Therefore, the next leaf to split is s_3, and it is splitted according to A_3. This produces the following partition into 4 leaves:

$$s_1 = \{1, 2, 3, 4\}, \quad s_2 = \{5, 6, 7, 8\}, \quad s_4 = \{9\}, \quad s_5 = \{10\}$$

Input/Output

The dataset is described in a file containing only integer numbers. The first number in the file is m, the number of instances; the second is n, the number of features. These two numbers are followed by mn feature values. **The last attribute is taken as the target attribute.** Example:

```
-------------------------
10 4
0  0  0  0
0  0  1  0
1  0  2  0
2  2  0  1
0  0  1  1
0  1  2  1
1  1  1  1
1  2  2  0
0  0  1  0
2  0  1  0
------------------------
```

You may assume that a feature has no more than 3 distinct values so that the numbers in each column can be eiter 0, 1 or 2. The target attribute (the last column) is binary, so that its values are 0 or 1.

A partition is described in a text file. For a situation where there are m instances, a partition is described by a sequence of numbers in the range $1 \ldots m$.
Each partition is specified in a separate line. The first string in a row describing a partition is the partition id.
With 10 instances, here is the file representing the partition {1,10}, {2,3,4,5}, {6,7,8,9}.
The first partition has id= X, the second id=$Y2$, and the id of the third is Z.

```
-------------------------
X 1 10
Y2 2 3 4 5
Z 6 7 8 9
------------------------
```

Your program should read as input one dataset file and one partition file. It should determine which partition to split, and according to what attribute. This information should be printed as a message. The program should then produce as output the new partition file. Example:

```
<you type:>
     MyProgram
<program prints:>
     Enter names of the files dataset input-partition output-partition
<you type:>
     dataset-1.txt partition-2.txt partition-3.txt
<program prints:>
     Partition Z was replaced with partitions Z1,Z2 using Feature 3
```

If the file partition-2.txt is as specified above, the file partition-3.txt might be:

```
-------------------------
X 1 10
```

```
Y2 2 3 4 5
Z1 7
Z2 6 8 9
-------------------------
```

Requirements:

Submission must be on eLearning.