Coffee Rust Disease Identification Using Decision Tree Algorithms

Ana Sofia Gutiérrez Tejada
Santiago Hidalgo Ocampo
Medellín, 31/October

Data Structures Designed

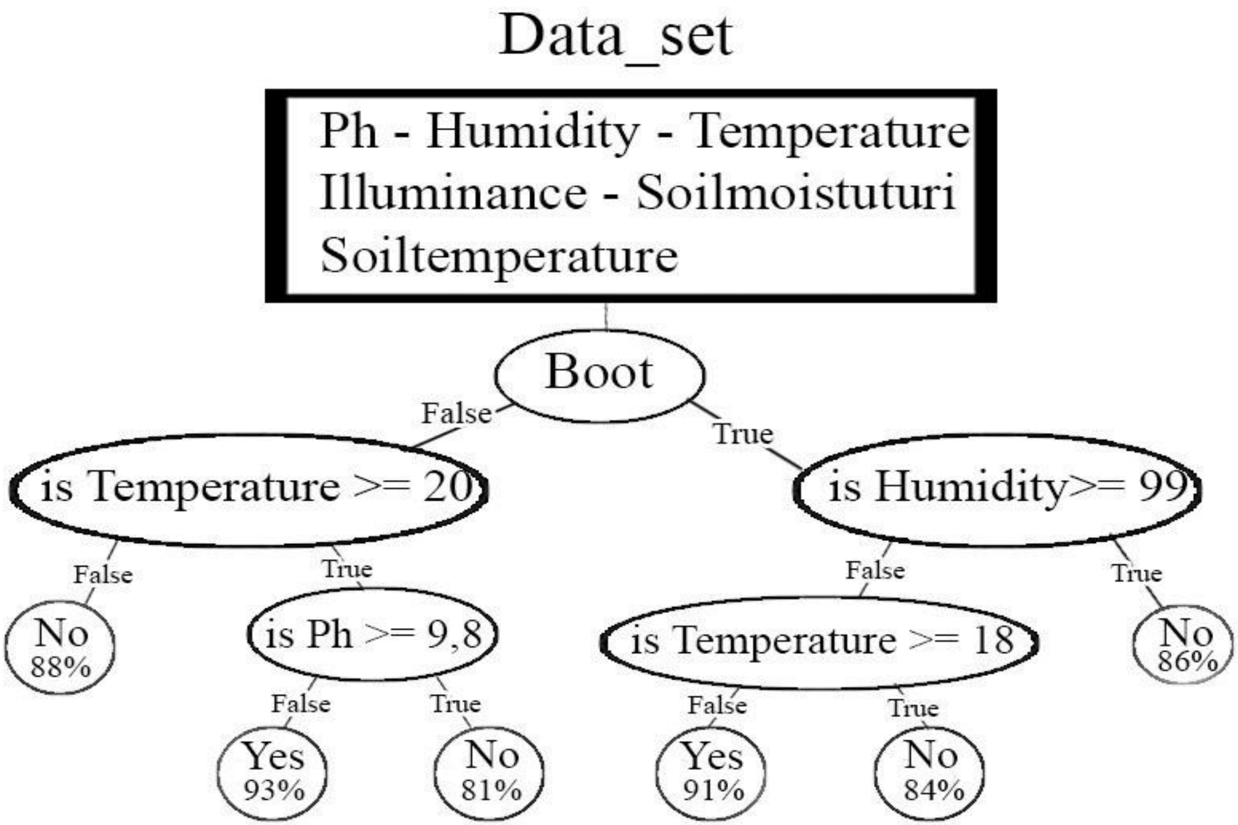

Figure 1: *Tree construction example with percentages*

Data Structure Operations

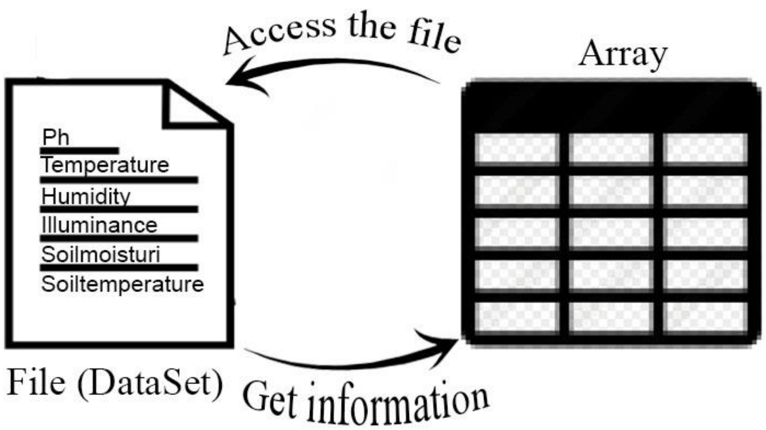

Figure 2: File reading.

Method	Complexity
Read data	O(n*m)
Unique values	O(n)
Is numeric?	O (1)
Find Best Split	O(n^2)
Gini	O(n)
Tree Building	O(n^2)
Is there rust?	O(n^2)

n: rows

m: columns

Table 1: Update table to report complexity analysis

Design Criteria of the Data Structure

- The CART algorithm was chosen because it has a great predictive capacity with respect to the other algorithms such as ID3, C4.5 or the CHAID

- The most striking aspect of this algorithm is that CART selects the cut that leads to the greatest decrease in impurity.

- The criterion of division of this algorithm allows to generate a tree with an acceptable purity with respect to the other algorithms.

- CART can work with continuous variables, which are adjusted to variables of the Data Set given.

Time and Memory Consumption

	Data Set 1	Data set 2	Data Set 3	Data Set 4
File Reading	0.004 sg	0.0049 sg	0.0045sg	0.0026 sg
Tree Building	0.6 sg	0.9905 sg	1.770 sg	0.7038 sg
Tree printing	0.0007 sg	0.0015 sg	0.0018 sg	0.0009 sg

Table 2: Update execution time of the operations of the data structure for each data set

	Data Set 1	Data set 2	Data Set 3	Data set 4
Memory consumption	126.1 Mb	126.4 Mb	127.0 Mb	126.8 Mb

Table 3: Memory used for each operation of the data structure and for each set data sets

Implementation

```
Run:   s
                Is ph >= 7.5?
                --> True:
                  Predict {'yes': 1}
                --> False:
                  Predict {'no': 7}
            --> False:
              Is ph >= 6.1?
              --> True:
                Is soil_temperature >= 23.0?
                --> True:
                  Is env_temperature >= 20?
                  --> True:
                    Is env_temperature >= 22?
                    --> True:
                      Predict {'yes': 1}
                    --> False:
                      Predict {'no': 1}
                  --> False:
                    Predict {'yes': 2}
                --> False:
                  Predict {'yes': 13}
              --> False:
                Predict {'no': 1}
          --> False:
            Predict {'no': 3}
    --> False:
      Is ph >= 6.71?
      --> True:
        Is illuminance >= 2701?
        --> True:
          Is env_temperature >= 32?
          --> True:
            Is soil_temperature >= 25.0?
```

Figure 3: Part of a printed tree

```
Actual: yes. Predicted: {'yes': '100%'}
Actual: yes. Predicted: {'yes': '95%', 'no': '4%'}
Actual: yes. Predicted: {'yes': '88%', 'no': '11%'}
Actual: yes. Predicted: {'yes': '72%', 'no': '27%'}
Actual: no. Predicted: {'no': '85%', 'yes': '14%'}
Actual: no. Predicted: {'yes': '88%', 'no': '11%'}
Actual: no. Predicted: {'no': '75%', 'yes': '25%'}
```

Figure 4: Printing of some evaluated data