1.1 ALGORITHM FOR THE DETECTION OF COFFEE LEAST RUST

David Calle González
Universidad Eafit
Country
dcalleg@eafit.edu.co

Julian Ramirez Giraldo
Universidad EAFIT
Country
jramirezg@eafit.edu.co

Mauricio Toro
Universidad Eafit
Colombia
mtorobe@eafit.edu.co

2. ABSTRACT

The problem is the necessity of detecting the presence of roya in coffee earlier, in this case, caturra coffee. This can be done by using certain variables like temperature, ph., and other variables obtained by sensors.

the solution of this problem is quite important due to the high loss of coffee regarding to the late detection of this plague in the crops. Also, especially in Colombia that is one of the most important coffee producers in the world, not being able to detect this sickness on time is a very big problem that could affect our coffee production, and hence, our economy.

Finally, the solution we decided to give was to implement a data structure called "trees of decisions" that allows us to make decisions, regardless of the redundancy, on a group of data that is entered or a data only (although for this it is necessary to have trained the tree) to precede whether coffee suffers from rust or not. From this We can say that we believe that it is the most efficient data structure to make predictions of this type since if it is programmed in the correct way and the values are entered in an optimal way we can have quite accurate and probable predictions.

KEYWORDS

Data processing, Roya detection

• Information systems~Linked lists • Theory of computation~Oracles and decision trees

3. INTRODUCTION

Initially, if we want to understand what conditions lead to the development of roya in coffee we need to understand what the coffee with roya is, known as well as "Roya del cafeto". this is a coffee plants sickness that produces a fungus named "Hemileia vastatrix" which is considered a parasite.

Coffee with roya was not a problem until the first two known epidemics, for example, the roya epidemics in Sri-Lanka could not be controlled, for this reason, the coffee crops had to be removed. The sickness arrival to Colombia took place in the 80, affecting the low areas crops.

Now, the conditions that allow coffee to get infected with roya and ease its propagations are: -The rain: the fungus needs water to exist.

-Temperature: it needs a temperature between 21 and 25 degrees Celsius

-Climatical Alterations: this generates stress in the crops and turns them into a propitious ambient for roya.

.

4. PROBLEM

The objective of this project consists in developing an algorithm that, through decision trees, could let us know whether a batch of coffee is or not infected with roya in an efficient way.

Solving this problem would prevent the loss of big amounts of coffee, something quite important for our country given the fact that we are the third biggest coffee producers in the world, whereby, it could affect our economy as well as it could affect other countries' economies since coffee is a global market .

5. RELATED WORK

5.1 C5.0

"This node uses the C5.0 algorithm to generate a decision tree or set of rules. C5.0 models divide the sample according to the field that offers the maximum information gain. The different subsamples defined by the first division are divided again, usually based on another field, and the process is repeated until it is impossible to divide the subsamples again. Finally, the divisions of the lower level are re-examined, and those that do not contribute significantly to the value of the model are eliminated or pruned."

"C5.0 can generate two types of models. A decision tree is a simple description of the divisions that have been found in the algorithm. The different terminal nodes (or "leaf") describe a subset of training data, and each of the cases included in the training data belongs exactly to a terminal node of the tree. In other words, it is possible to make exactly one prediction for each specific data record present in a decision tree."

5.2 QUEST

"QUEST, or rapid and efficient statistical tree, is a binary classification method to generate decision trees. One of the main motivations for its development has been the reduction of the processing time necessary for large-scale C&RT analyzes with several variables or several cases. A second objective of QUEST is to reduce the tendency of tree classification methods to favor entries that allow more divisions, that is, continuous input fields (numerical range) or those corresponding to various categories."

"QUEST uses a sequence of rules based on significance checks to evaluate the input fields of a node. For the purpose of selection, you should only perform a single check on the different entries of a node. Unlike what happens with C&RT, not all divisions are examined and, unlike C&RT and CHAID cases, combinations of categories are not checked when evaluating an input field for selection. This increases the speed of the analysis."

5.3 C&R

"The Classification and Regression Tree (C&R) node is a tree-based prediction and classification method. Similar to C5.0, this method uses repeated partition to divide training records into segments with similar output field values. The C&RT node begins by examining the input fields to find the best division, which has been measured by reducing the impurity index resulting from the division. The division defines two subgroups, which continue to be divided into two other subgroups successively until a stop criterion is activated. All divisions are binary (only two subgroups are created)"

5.4 CHAID

"CHAID, or automatic detection of chi-square interactions (Chi-squared Automatic Interaction Detection), is a classification method to generate decision trees using chisquare statistics to identify optimal divisions."

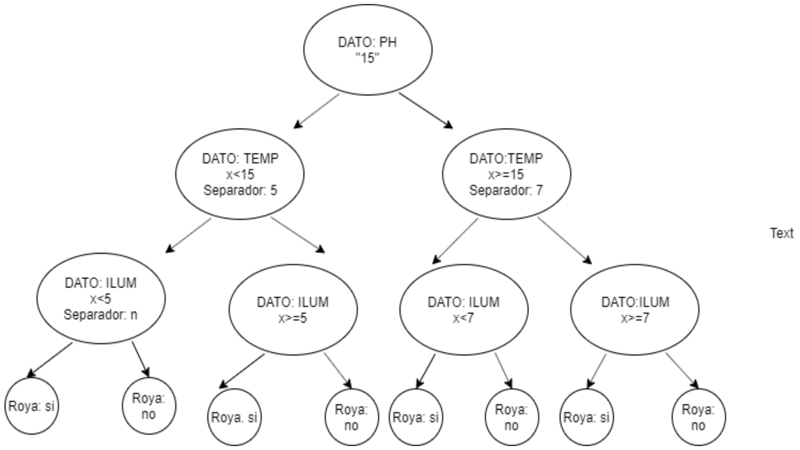
"CHAID first examines the cross-tabulation tables between the input fields and the results, and then controls the significance through a chi-square independence check. If several of these relationships are statistically important, CHAID will select the most relevant input field (the smallest P value). If an input has more than two categories, compare these categories and counteract those that do not show differences in results. To do this, the pair of categories that present the least difference will be put together, and so on. This process of merging categories stops when all remaining categories differ from each other at the specified check level. In the case of nominal input fields, all categories can be merged. However, in ordinal sets, they can access the adjacent categories."

6. 4. DECISION TREES

7.

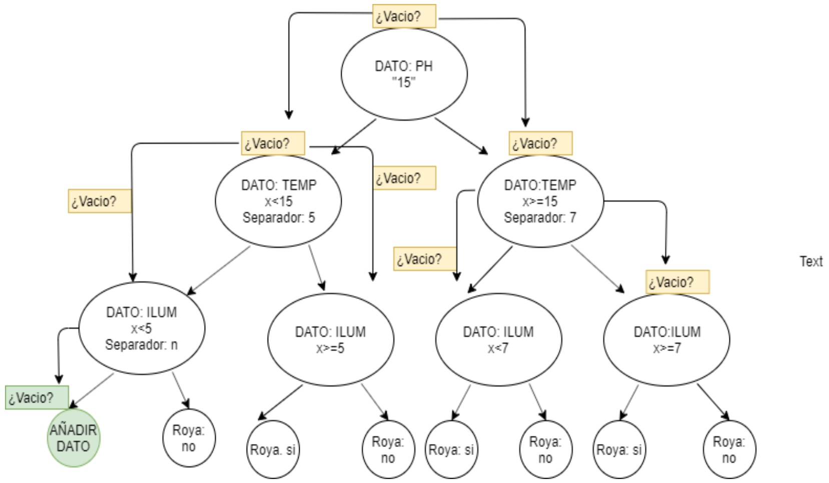
The data structure chosen is a decision tree created by us that follows the following structure:

It should be clarified that not only will work with the 5 generations of data present, they will be more but it is the initial model of the structure. It is about each node containing one of the dataset data and defining for each generation a value called "value that best separates" that is responsible for sending the decision to the left or right node.



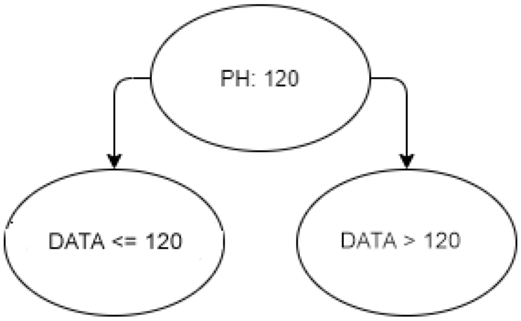
4.1 DATA STRUCTURE OPERATIONS

4.1.1 ADD



The algorithm is responsible for going node by node in the tree and asking “Is it empty or not?” Until you find an empty one (represented in green in the graph) and adds the data in the position.

4.1.2 SEPARATE DATA



This method allows to distribute the data depending on a data called "base data" which is what defines the separation of the data to each node (left and right), better called the data of the variable that best separates the data then, yes a data is less than that base data goes to the left side, if the data is larger than the base data goes to the right side.

It should be clarified that this function works in conjunction with the function mentioned in **4.1.1 ADD** to define where there is an empty space to add the data and how to separate them

4.2 CRITERIA FOR WHICH THE DATA STRUCTURE WAS CHOSEN

The decision to take the data structure of “decision trees” was given because it was concluded that the ease of data search is better in a binary structure than in a linked list because to search for an element in a linked list we must go through one to one the elements to find what is sought instead in a binary decision tree that by the same structure the data is already ordered in the case but the complexity is O (1) or rather, it is reduced as minimum search at 50% compared to the linked list

4.2 COMPLEXITY ANALYSIS

Print	**O(n), n is the number of data**
Tree	**O(2n), n the number of variables**
Scroll (Recorrer)	**nlog(n), n the number of data**
Add	**O(1)**

4.3 TIMES OF EXECUTION AND USE OF MEMORY

The graph shows different columns defined as:

Operation: The evaluated method

Mmax: Maximum memory used

Mmin: Minimum memory used

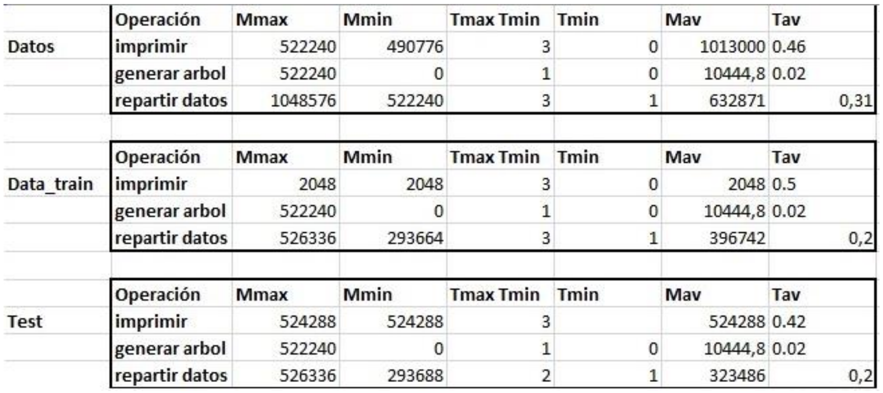
Tmax: Maximum time

Tmin; Minimum time

Mav: Average Memory

Tav: Average time

5. BETTER DECISION TREES



	Operación	Mmax	Mmin	Tmax Tmin	Tmin	Mav	Tav
Datos	imprimir	522240	490776	3	0	1013000	0.46
	generar arbol	522240	0	1	0	10444,8	0.02
	repartir datos	1048576	522240	3	1	632871	0,31

	Operación	Mmax	Mmin	Tmax Tmin	Tmin	Mav	Tav
Data_train	imprimir	2048	2048	3	0	2048	0.5
	generar arbol	522240	0	1	0	10444,8	0.02
	repartir datos	526336	293664	3	1	396742	0,2

	Operación	Mmax	Mmin	Tmax Tmin	Tmin	Mav	Tav
Test	imprimir	524288	524288	3		524288	0.42
	generar arbol	522240	0	1	0	10444,8	0.02
	repartir datos	526336	293688	2	1	323486	0,2

The data structure we use are decision trees, they are classes that contain nodes with information to determine an answer as the data passes through them. As we can see (Figure 1) each node contains a variable and a value to separate called “the value that best separates” to send data to the left or righ**t.**

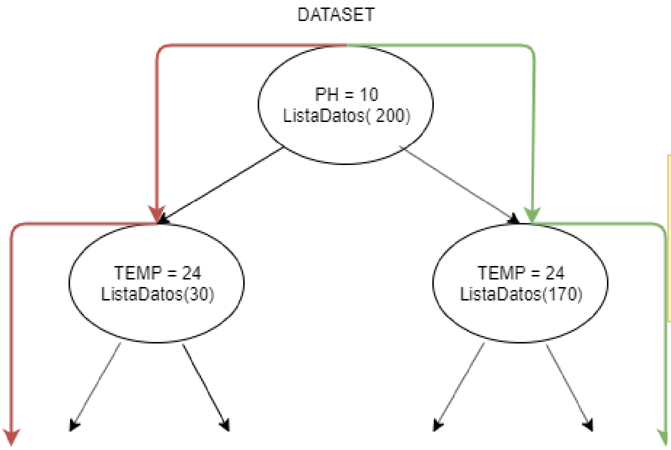


Figure 1: Representation of the data structure where you can see the name of the variable, the data that best separates and the dataset with the number of data that is distributed as it fall

5.1 OPERATIONS OF THE DATA STRUCTURE

5.1.1) DATA FILTER

The filter of the data that enters each node has a simple process but it can be complex to understand, it is that each node of the tree contains very important data called “the data that best separates” these are calculated with the impurity formula based on the Gini formula from which the best value is chosen, it is entered as a comparison value in the node and from there the data goes to one side or the other depending on the value they have in the variable being evaluated, So with all the nodes until you reach the conclusion.

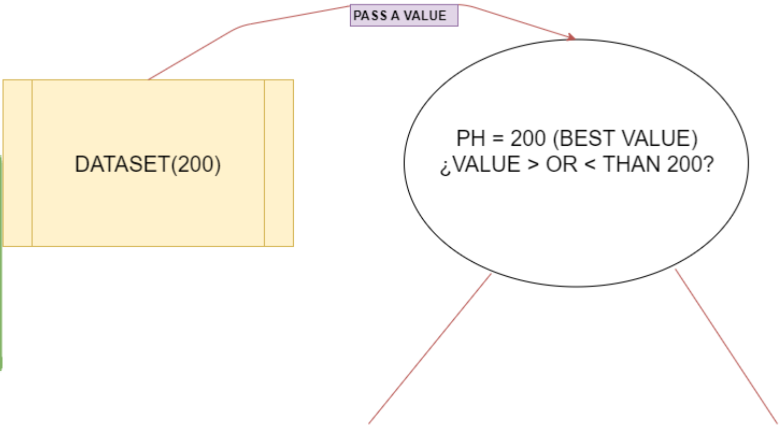
$$I_G(f) = \sum_{i=1}^{m} f_i(1 - f_i) = \sum_{i=1}^{m}(f_i - f_i{}^2) = \sum_{i=1}^{m} f_i - \sum_{i=1}^{m} f_i{}^2 = 1 - \sum_{i=1}^{m} f_i{}^2$$

Figure 1: Gini impurity formula

Figure 2: A value passing to the node from a data list

5.1.2) DATA ENTRY

For this, two things are done initially, the first is to train the tree so that it can have a higher percentage of accuracy before entering a new data, the second is to enter the new data with the respective values (ph, soil_temperature, soil_moisture, illuminance, env_temperature, env_humiduty, label) and hence the tree processes the response through the data filter.



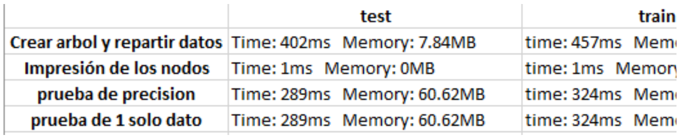
5.2) DESIGN CRITERIA OF THE DATA STRUCTURE

The tree itself was designed with the only purpose of being able to separate the data according to the best possible values in any order, specially the best order. The data was saved in a linked list and the nodes of the tree implement them due to its easiness to add nodes (cyclic linked list) and the only need of being analyzed in general, we really do not need to access specific values, another reason to include linked lists In every node is the possibility of analyzing the dispersion of the data in every single node, obviously this was implemented without taking into account memory usage since we copy data so there would be no problem with references.

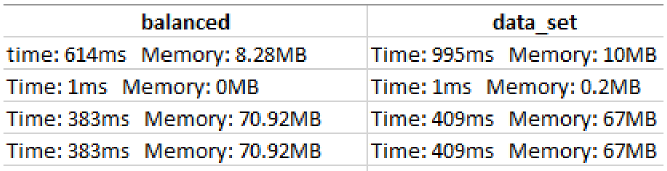
5.3) COMPLEXITY ANALYSIS

Print the information in the nodes	**O(n), n is the amount of data.**
Creating the tree	**O(2m+m*n), m is the number of generations of the tree and n the amount of data.**
Test the precisión of the tree	**O((2n +m*n)*p), m is the amount of data, n the number of generations and p the number of random trees.**
Test 1 Data	**O((2n +m*n)*p), m is the amount of data, n the number of generations and p the number of random trees.**

5.4-5.5) EXECUTION TIME AND MEMORY USED



	test	train
Crear arbol y repartir datos	Time: 402ms Memory: 7.84MB	time: 457ms Mem
Impresión de los nodos	Time: 1ms Memory: 0MB	time: 1ms Memory
prueba de precision	Time: 289ms Memory: 60.62MB	time: 324ms Mem
prueba de 1 solo dato	Time: 289ms Memory: 60.62MB	time: 324ms Mem



balanced	data_set
time: 614ms Memory: 8.28MB	Time: 995ms Memory: 10MB
Time: 1ms Memory: 0MB	Time: 1ms Memory: 0.2MB
Time: 383ms Memory: 70.92MB	Time: 409ms Memory: 67MB
Time: 383ms Memory: 70.92MB	Time: 409ms Memory: 67MB

5.6) RESULT ANALYSIS

The criteria for choosing the structure used and the modifications made from the previous structure (which was the basic version) was based on the same criteria, that is, based on the complexity of searching for data (which is minimal), in addition to the trees They allow us to make decisions in a more simple and orderly way (that's why it's called a search tree). Obviously everything was thought based on the treatment of the datasets since a LinkedList would be something inefficient to process the data, all based on the review of complexities.

6. CONCLUSIONS

The most important issued of this report, we think, is how we implemented the data structure, a decision tree with linked lists in each node to save the data so it could be analized easily.

the most important results we got in this project are the diferences we found when choosing the right values to separate Data in the right order so it could be more precise.

Mainly, the difference between our first solution and our last solution is the implementation of random trees forests, this little change did not change much, neverthless, we are sure when we say it helped slightly so the tree could be more precise

As a future work, we would like to do 2 concrete modifications to the project, first, enhance the algorythm to choose the best values to separate the data, and second, implement a model to read multispectral images in order to have more data about the coffee and, therefore, be more precise. regarding to the presicion, we noticed that the variables we got were not good enough to separate the data precisely, more variables are needed and a better way to measure their preciseness.

6.1 FUTURE WORK

As we explaines in our conclussions, we would like to use more variables and choose them better, also, we think that learning more about this kind of decision trees, this because there is a lot of things we figured out in our way but possibly there are better and more efficient ways to implement them.

ACKNOWLEDGEMENTS

8. REFERENCES

IBM, Nodo CHAID: https://www.ibm.com/support/knowledgecenter/es/SS3RA7_sub/modeler_mainhelp_client_ddita/clementine/chaidnode_general.html

IBM, Nodo Árbol C&R:

https://www.ibm.com/support/knowledgecenter/es/SS3RA7_sub/modeler_mainhelp_client_ddita/clementine/cartnode_general.html

IBM, Nodo QUEST:

https://www.ibm.com/support/knowledgecenter/es/SS3RA7_sub/modeler_mainhelp_client_ddita/clementine/questnode_general.html IBM, Nodo C5.0

https://www.ibm.com/support/knowledgecenter/es/SS3RA7_sub/modeler_mainhelp_client_ddita/clementine/c50node_general.html