PREVENTING ROYA IN COFFEE PLANTS USING DECISION TREES

Juan Sebastián Diaz Osorio
Universidad EAFIT
Colombia
jsdiazo@eafit.edu.co

Liz Oriana Rodrigues Cruz
Universidad EAFIT
Colombia
lorodriguc@eafit.edu.co

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

SUMMARY

The Roya it is a fungus that implants in the leafs of the coffee plants searching for survival, as they do that they became one of the most dangerous sicknesses that a plant can get, as it comes with a price to pay because a large amount of coffee powder does not have a way of recovery, as a consequence after one these plantations become sick, it has to extracted and fired up immediately and by that means the lost of the product, the coffee.

Machine Learning [7] would be an amazing support tool for this situation. Imagine a computer that it studies each second the crop ground characteristics and alert about Roya occurrence. Thus, the plague start point would be known and could be eliminated.

1. INTRODUCTION

The management and crop care is a hard job if you don't have correct tools, because diseases and parasites are avoided only when they are prevented.

The coffee, Colombia's main export product, is affected by a fungus called "Roya", which changes his leaves' color into a yellow tone until withering and the leaves fall, jamming his maturation and production process.

Even though laboratory tests have originated a transgenic coffee that is invulnerable to Roya [2], the replacement for this new crop could bring changes in flavor or care topics. In this text, we analyze the possibility to study the ground by collecting data and prevent Roya occurrence in coffee plants using some computer solutions.

2. PROBLEM

We look for preventing Roya occurrence in the coffee plants to avoid loss, through machine learning algorithms that analyze terrain variables and foretell when and where is safe to farm.

3. RELATED WORKS

A decision tree is a machine learning method that classifies new data. It has a tree-like structure made of nodes (linked with other through branches) and logic test branches (which allows going from a node to another). New data cross through this tree and when they finish it, they will have a new label.

An algorithm builds the tree from a dataset (called training data). More training data implies more accuracy in the decisions.

Now, we briefly show algorithms that create decision trees:

3.1 ID3 [1]

J. Ross Quilan, in 1979, created the ID3 algorithm (Induction Decision Trees). It uses artificial intelligence encompassing the search for a thesis or rules by means of a set of examples. In addition, it is a constructive algorithm to obtain decision trees based on the CLS algorithms.

It highlights the application and realization of decision trees from top to bottom, directly, without having to make use of Backtracking, however, this uses recursion and is based on the examples provided. For that, one must have basic knowledge of what is and how the concept of Information Gain is measured, which consists of identifying the most useful attribute for the process to be carried out.

To solve a problem through ID3 algorithms we must take into account elementary details such as the inputs, which consists of a list of examples with data or values, each stored with its attribute, the output, which consists of the tree of final decision that will have separated the examples with their respective classes, which can be predefined or discrete.

The decision tree is traversed from the root and a tour is made throughout its structure, that is, root and nodes based on the

Predictors				Target
Outlook	Temp	Humidity	Windy	Play Golf
Rainy	Hot	High	False	No
Rainy	Hot	High	True	No
Overcast	Hot	High	False	Yes
Sunny	Mild	High	False	Yes
Sunny	Cool	Normal	False	Yes
Sunny	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Rainy	Mild	High	False	No
Rainy	Cool	Normal	False	Yes
Sunny	Mild	Normal	False	Yes
Rainy	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes
Sunny	Mild	High	True	No

value of the attribute in classification until it reaches a terminal node.

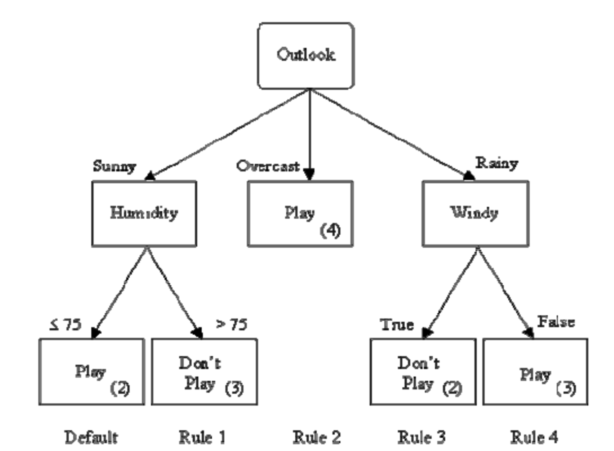
One of which of the trees with this type of algorithm is that it can adapt to certain types of problems that can be had.

3.2 C4.5 [4]

C4.5 algorithm is an extension of ID3, but this uses recursion to generate the decision tree from training data. It considers each data attribute to split the dataset into new groups, based on the highest information gain (this is a data mining term. It means that a variable can be used as a classifier thanks to great accuracy).

This algorithm works for both discrete and continuous attributes, doing either a big data test (with n possible values) or binary test (two values only) to find the information gain. The system analyzes all data to decide which is the logic question more efficient and accuracy to do.

Some advantages over ID3 are:
- It works even if there is missing information
- It has a good computational efficiency
- It generates smaller decision trees.
- It includes a post-pruning process, where the tree is evaluated again to improve performance.

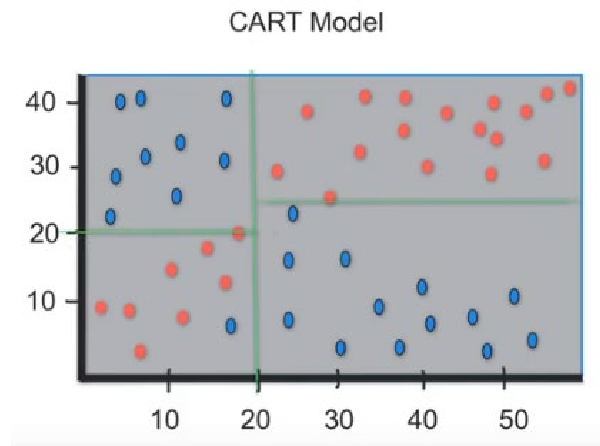


A C4.5 algorithm example is at scottjulian's repository [6]. It is implemented in Java and a great idea about how solve our problem.

3.3 CART [3]

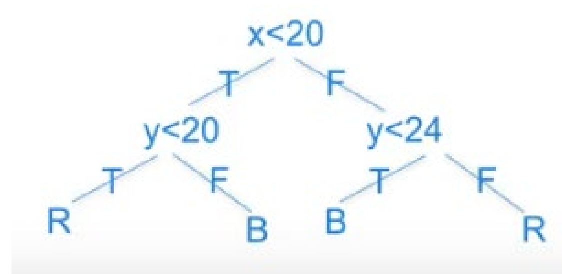
When a dataset can be classified with two labels only, (it means either "A group" and "B group", or "Full" and "Partial"...), a good idea to create a decision tree model is the CART algorithm.

Let x and y axis be two numerical categories of a dataset in the next picture.



Each data in the dataset can be positioned in the diagram and they can be split into subgroups. Each subgroup will have a dominating label and it means new data has a high probability to have that label.

CART algorithm generates a decision tree like this:



So, new entry data can be positioned and classified in the diagram according to values with good accuracy.

3.4 SLIQ [5]

SLIQ (Supervised Learning In Quest) is a decision tree classifier that can handle both numerical and categorical attributes. It stands out because it can handle a big amount of training data (unlike other algorithms) thanks to its pre-sort method, acquiring awesome accuracy and efficiency.

First, it creates a sorted list for each numerical and categorical attribute, including a list for the classes and tree nodes (class list).

Then, it builds nodes and links it according to previous lists, creating and updating nodes for the class list. Next, the algorithm deletes cloned nodes.

In the tree pruning, the tree is codified while some nodes are deleted. At this point, the class list will have each class with its respective node and new data must cross the tree, reach a node, and compare itself with the class list to be classified.

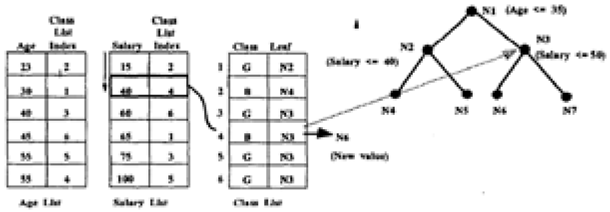


Fig. 7. Class List Update: Example

In performance tests, SLIQ shows to be faster than IND-CART and generates smaller trees than IND-C4.

REFERENCES

[1] Caparrini, F. 2018. Inductive Learning: Decision Trees [Translated title]. (December 2018). Retrieved August 9, 2019 from https://bit.ly/2TmPNza

[2] Croplife. Coffee plant's Roya [Translated title]. Retrieved August 9, 2019 from https://bit.ly/2w9OCbL

[3] Eckhardt, R. 2017. Introduction to the CART Model. (9 Feb 2017). Retrieved August 9, 2019 from https://bit.ly/2KrmfNO

[4] López, B. 2005. C4.5 Algorithm [Translated title]. (December 2005). Retrieved August 9, 2019 from https://bit.ly/2pcFG17

[5] Mehta, M., Rakesh A. and Rissanen J. SLIQ: A Fast scalable classifier for data mining. Retrieved August 9, 2019 from https://bit.ly/31Ak801

[6] scottjulian. 2015. MyC45. Java file. (6 March 2015). Retrieved from August 10, 2019 from https://bit.ly/2TobRte

[7] Vincent, J. 2019. The state of AI in 2019. (28 Jan 2019). Retrieved August 9, 2019 from https://bit.ly/2Te2RFY