COFFEE RUST DETECTION FOR CATURRA VARIETY USING DECISION TREES

Paulina Pérez Garcés
Universidad EAFIT
Colombia
pperezg@eafit.edu.co

Santiago Cartagena Agudelo
Universidad EAFIT
Colombia
scartagena@eafit.edu.co

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

ABSTRACT

Coffee is one of the main plant-based goods consumed today, as well as the second most traded commodity in modern economy after oil. Its production takes place mostly in developing countries such as Colombia, where a malfunction during the process could affect thousands of coffee farmers. Coffee rust does just that, reduce the quality and quantity of grains devastating entire productions yearly. [1]

One of the most effective ways to prevent this disease from spreading is an early detection, which makes effective the extraction and replacement of withered crops. Using wireless sensor networks, it is possible to obtain data from each plant's conditions; which allows us to determinate whether a specific plant is in the early stages of infection or perfectly healthy.

Keywords

Coffee Rust; Decision Trees; Data Structures; Big O Notation; Complexity; Caturra Coffee; Machine Learning.

ACM CLASSIFICATION KEYWORDS

Applied computing → Computers in other domains → Agriculture

Applied computing → Life and medical sciences → Bioinformatics

1. INTRODUCTION

Coffee has been one of the most popular drinks since the 1800s, when all of eastern British colonies began growing it in large quantities. Around 1879 coffee farmer in Sri Lanka (formerly called Ceylon) noticed that a new fungus called Hemileia vastatrix was affecting their plantations, but an efficient cure was never found.

The losses caused by this disease made it unsustainable to keep growing coffee in those regions; therefore, plantations of a single healthy strain -Arabica coffee- were moved to the Americas, which quickly became the biggest productors of this good. The crops there were constantly checked, and remained rust-free until 1970, when spores from the fungus were found in Bahia, Brazil. [2]

Since arabica coffee was the most planted strain in America, and due heavy winds and rainy seasons, rust spread around the continent in a matter of years. Countries where the plant had become one of the main tradable products began suffering substantial economic losses.

2. PROBLEM

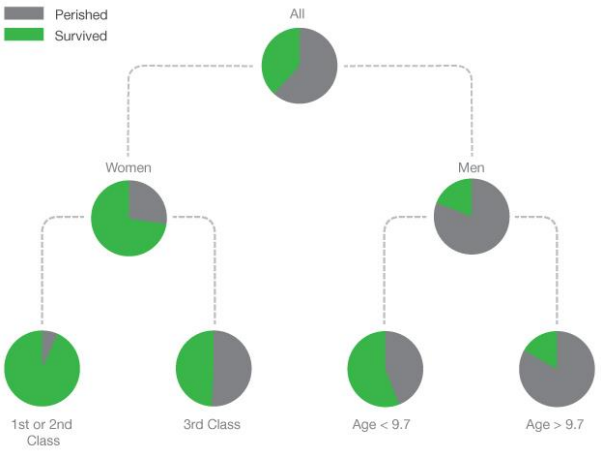
The intent of this project is to create an algorithm capable of determinate whether a coffee bush is vulnerable and suffering from rust or not. These results will be based on data such as soil temperature, weather temperature, soil's PH, illuminance and others; all taken from wireless sensor networks placed in a greenhouse in University EAFIT in Medellín, Colombia.

Said algorithm will be making decisions using methods known as decision trees. This approach uses statistics from the previously classified crops to identify new affected plants, in more technical words, it is a supervised learning method used for classification and regression. [3]

3. RELATED WORK

3.1 CART

CART is an alternative decision tree algorithm. It can handle both classification and regression tasks. This algorithm uses a metric called "Gini Index"; whose objective is to create decision points for classification tasks. CART is similar to C4.5 but there is a notable difference between them: CART constructs the binary trees based on numerical splitting, whereas C4.5 include the step of rule sets. The CART algorithm is structured as a sequence of questions, the answers to which determine what the next question, if any, should be. The result of these questions is a tree like structure where the ends are terminal nodes at which point there are no more questions. [4]



[5]

3.2 CHAID

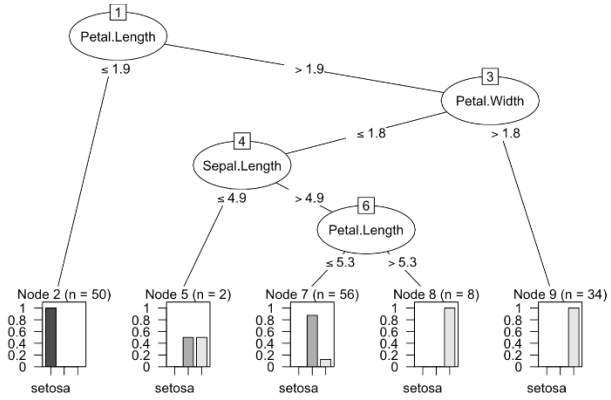
This algorithm, short for Chi-squared Automatic Interaction Detector is one of the oldest (1980) and most popular decision trees. It is a simple set of rules that lets you add more than two tree branches on a single node. The name, as well as some of its principles, comes from the Chi-Squared Statistic that expresses the relationship between two or more non-numerical variables. This technique can easily analyze large amounts of data, but that would also create big amounts of divisions inside the tree. [6]

3.3 ID3

One of the most popular algorithms is the ID3 algorithm, also known as the Iterative Dichotomiser 3. It is used to generate a Decision Tree from a dataset. ID3 is also considered as a precursor to C4.5, as well as many other decision tree algorithms. A big advantage of using ID3 algorithm is that it builds the fastest but also the shortest trees; this makes more understandable the problem. Just as all the algorithms used for building decision trees it has a couple of disadvantages, the principal one being that if you want to implement ID3 algorithm is expensive to train it, meaning you need a lot of data to do so. [7]

3.4 C5.0

C5.0 algorithm is the successor of C4.5, developed by Ross Quinlan in 1993. It splits decisions from the one that gives more information to the ones that can easily be removed or ignored. When the test data comes through, it predicts a single category to which it belongs.; these classes into which the data fall can be discrete attributes divided into a set of intervals. C5.0 trees can be very useful when data is missing from datasets, because it works relatively well without it. The main differences with its processor are the speed, memory usage and size of the trees. [3][8]



[9]

3.RED-BLACK TREE

In computer science a Red-Black Tree is an abstract type of data, specifically is a BST(Binary search tree), there are many kinds of BST but most of them are used for predicting selections or samples of whatever, this is not that case, the Red-Black tree is a different kind of BST, this one has the objective to store data, the best part is that this kind of BST also search, insert and delete data in O(log(n)) complexity, where the letter "n" is the quantity of elements in the tree.

If you want to work with a Red-Black tree you must consider this is a kind of tree in which every node has a color attribute, there are two options: Black or Red, also you need to remember that leaves of any Red-Black tree do not contain relevant data in general, besides you must follow some properties or some rules when you are implementing this kind of BST, those important rules are the following ones:

- As we mentioned before, every node is going to be black or red, but not both or another color.
- All the null leaves must be black.
- Every red node must have two more black tree children.
- Is mandatory the tree root to be black.

4. RED-BLACK TREE

Binary search trees are really efficient when you care about organization; their creation and modifications are not complicated and follow a simple logic: right if the data is bigger, left otherwise. Additionally, new data will cause a "reorganization" for the properties of this tree to stay the same.

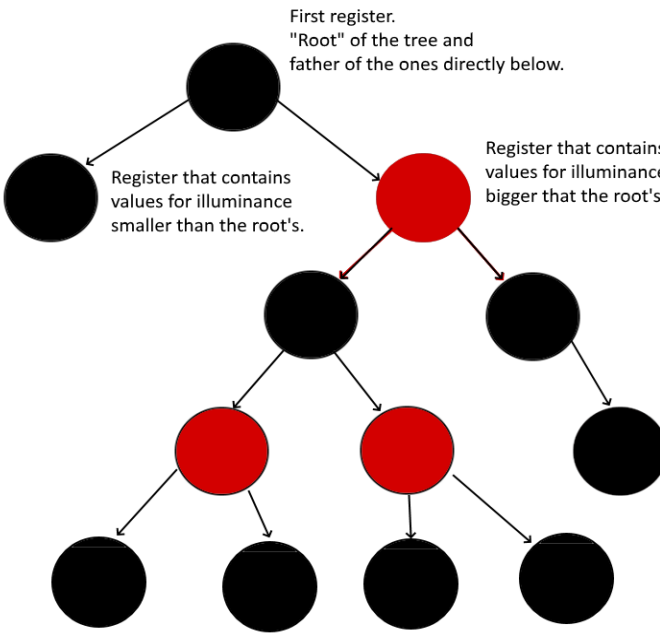


Figure 1: Red-Black tree structure.

4.1 OPERATIONS OF THE DATA STRUCTURE

4.1.1 Insertion of a register to the tree:

When inserting new data to the tree the structure of it may vary a little bit so it stays balanced; but it doesn't change what it does in principle, search for the place of a new value based on the data that is smaller or bigger.

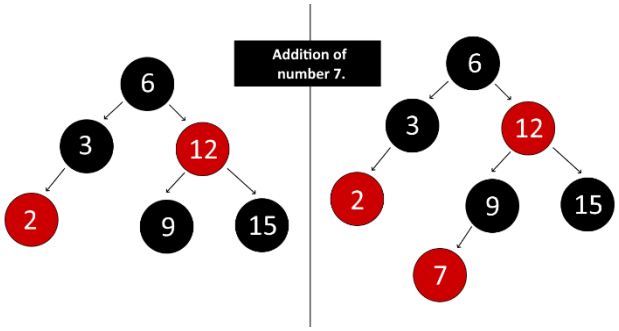


Figure 2: Addition operation pictured with simple numeric data.

4.1.2. Removal of a register from the tree:

This might be trickier than adding a new element but mainly it is the same thing. Once it deletes the desired value, the data will reposition itself to

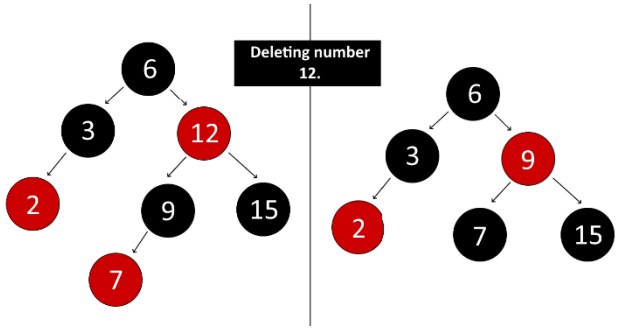
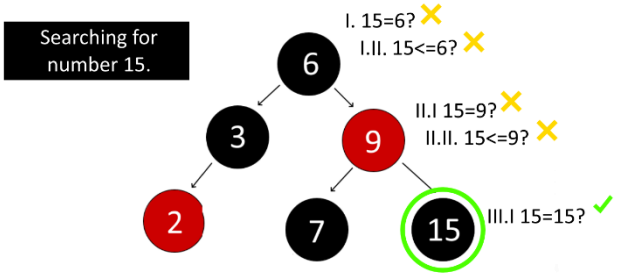


Figure 3: Deletion operation pictured with simple numeric data.

4.1.3. Searching for a value inside the structure:

Once the structure is known, checking if a value is inside the tree is fairly simple. Following the truth/false statements about the size of a value one is able to find a value.



4.2 DESIGN CRITERIA OF THE DATA STRUCTURE

Red-Black Trees are efficient when thinking about time and space. Two things that are really valuable due their shortage and importance. It is also a really understandable data structure that, when show graphically, can make anyone understand the basis of how it works.

4.3 COMPLEXITY ANALYSIS

METHOD	*COMPLEXITY*
Search	O(log(n))
Insertion	O(log(n))
Removal	O(log(n))

Table 1: Table to report complexity analysis

4.4 EXECUTION TIME

OPERATION	TIME FOR TRAINING DATASET	TIME FOR TESTING DATASET
Creation	23.5 ms	17.3 ms
Insertion	9.8 ms	10.6 ms
Removal	4.5 ms	6.4 ms
Search	3.2 ms	5.4 ms

Table 2: Execution time of the operations of the data structure for each data set.

4.5 MEMORY USED

	MEMORY USAGE FOR TRAINING DATASET	MEMORY USAGE FOR TESTING DATASET
Creation	84.2MB	82.7MB

Table 3: Memory used for each dataset.

4.6 RESULT ANALYSIS

The data structure used for the project works as fast as man can hope for, with a complexity lower than others. The organization is easy to understand, as well as useful. Memory usage could be improved, but the results weren't bad. A Red-Black tree continues to be one of the most efficient data structures.

EFFICIENCY	FOR 100 REGISTERS	FOR 150 REGISTERS
TIME	23.5 s	11.2 s
MEMORY	84.2 MB	84.4 MB

Table 4: Result Analysis for used data structure

COMPLEXITY	STACK	LINKED LIST	HASH TABLE	RED-BLACK TREE
SEARCH	O(n)	O(n)	O(n)	O(log(n))
INSERT	O(1)	O(1)	O(n)	O(log(n))
DELETE	O(1)	O(1)	O(n)	O(log(n))

Table 5: Complexity of different data structures.

REFERENCES

1. Goldschein, E. 11 Incredible Facts About the Global Coffee Industry. *Business Insider*. 2011. https://www.businessinsider.com/facts-about-the-coffee-industry-2011-11?op=1#according-to-the-world-wildlife-fund-of-the-50-countries-with-the-highest-deforestation-rates-from-1990-to-1995-37-were-coffee-producers-11

2. Arneson, P.A. Coffee rust. *The Plant Health Instructor*. 2000. https://www.apsnet.org/edcenter/disandpath/fungalbasidio/pdlessons/Pages/CoffeeRust.aspx

3. Scikit-learn Documentation. Decision Trees. https://scikit-learn.org/stable/modules/tree.html

4. Brownlee, J. Classification And Regression Trees for Machine Learning. *Machine Learning Mastery*. https://machinelearningmastery.com/classification-and-regression-trees-for-machine-learning/

5. The Fact Machine. http://www.thefactmachine.com/wp-content/uploads/2014/05/CART-TreeMap6Node.jpg

6. Statsoft. Popular Decision Tree: CHAID Analysis, Automatic Interaction Detection. http://www.statsoft.com/Textbook/CHAID-Analysis

7. Howe, B. Building Trees: ID3 Algorithm. Coursera. https://es.coursera.org/lecture/predictive-analytics/building-trees-id3-algorithm-7PNNn

8. IBM. C5.0 Node. https://www.ibm.com/support/knowledgecenter/en/SS3RA7_15.0.0/com.ibm.spss.modeler.help/c50node_general.htm

9. Kuhn, M; Ross, Q. Plot a decision tree. https://topepo.github.io/C5.0/reference/plot.C5.0.html