

Prediction and analysis of state exams via the use of decision trees





Simon
Cardenas
Villada



Juan Pablo
Yepes García



Miguel
Correa



Mauricio
Toro

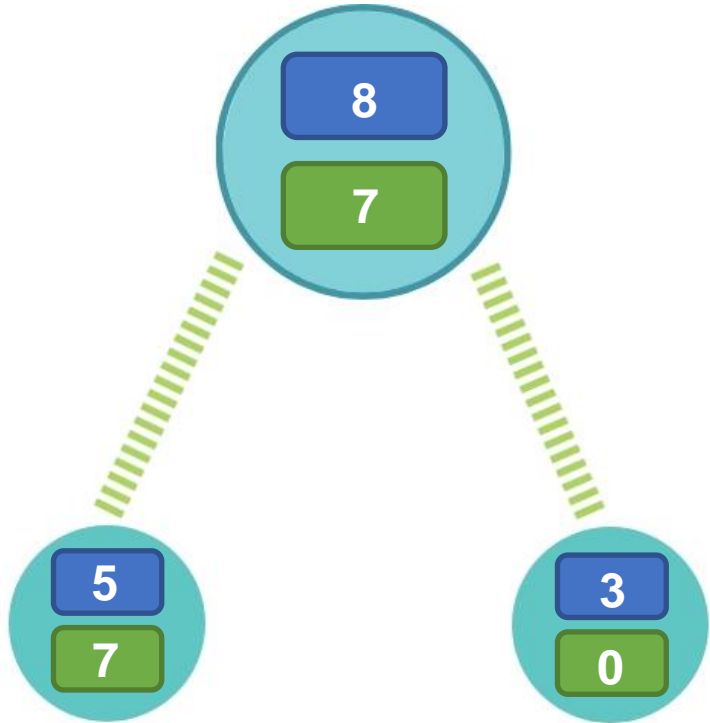


[http://github.com/](http://github.com/Jpyepes/proyecto/) Jpyepes /proyecto/

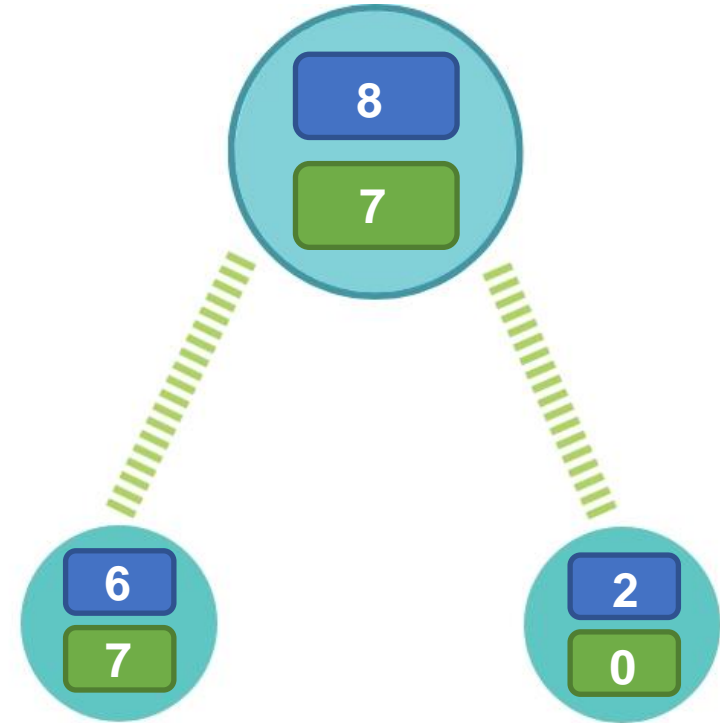


This model represents a hypothetical situation in which 3 conditions approximate the probability of success of the student.

Node Splitting



This node split is based on the condition “fami_nivelsisben == Nivel 2.” For this case, left Gini impurity is 0.474, right Gini impurity is 0.0, and weighted Gini impurity is 0.385. Elements in blue boxes were marked as having success “0”, and elements in green boxes with success “1”.



This node split is based on the condition “fami_estratovivienda.1 == Estrato 1” For this case, left Gini impurity is 0.90, right Gini impurity is 0, and weighted Gini impurity is 0.429. Elements in blue boxes were marked as having success “0”, and elements in green boxes with success “1”.

Complexity of the algorithm

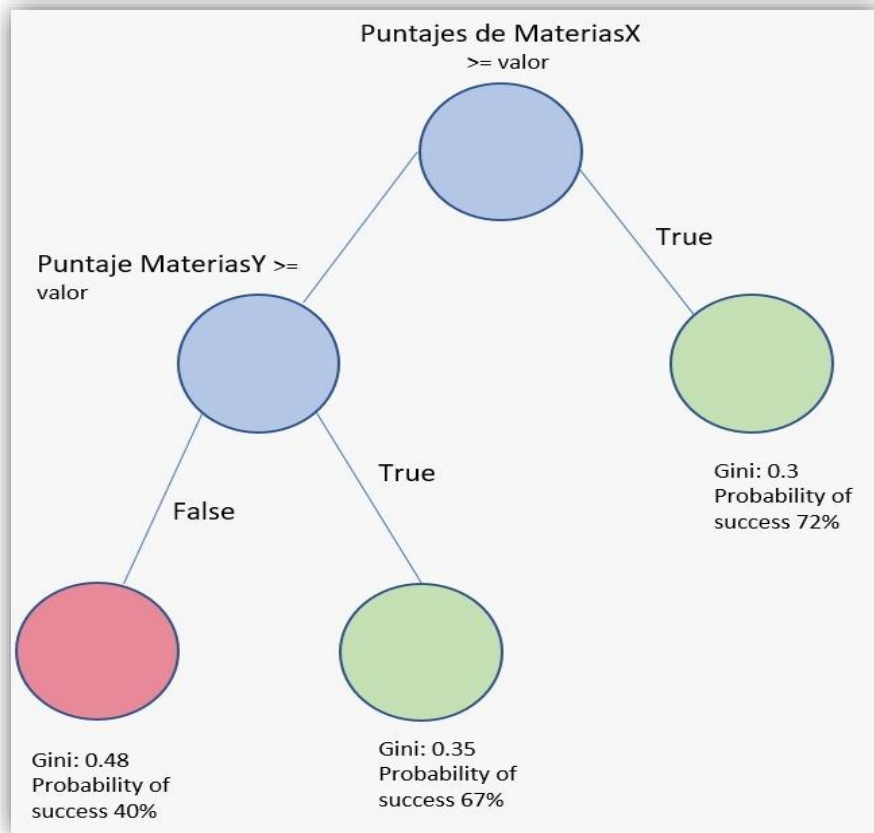


	Time complexity	Space complexity
Training	$O(m*n*2^m)$	$O(m*n)$
Testing	$O(m*n*\log n)$	$O(1)$

Where n is the number of rows and m is the number of columns of the two dimensional array that stores the data sets.



Decision Tree Model



Graph of a tree with possible values

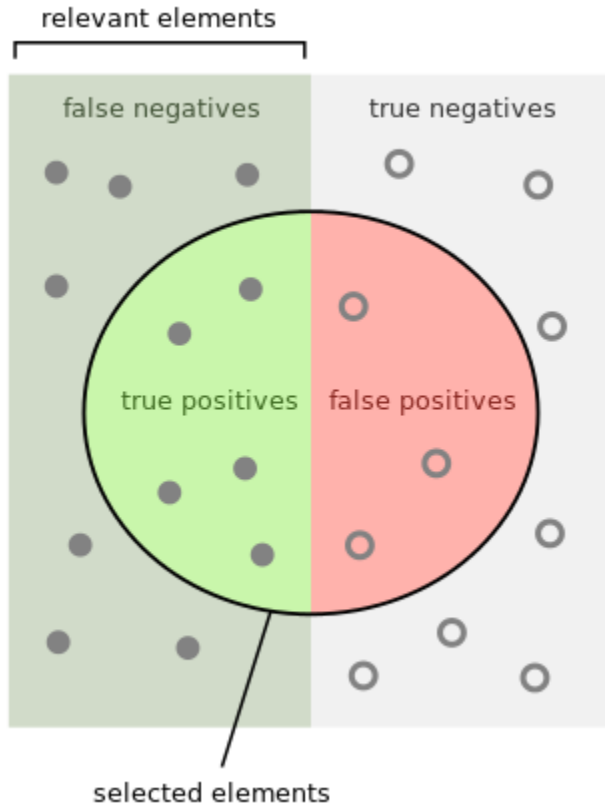
Deciding factors in prediction

Results of the different academic modules the ICFES tests



The tree only analyses the columns that contain the results of the academic modules (Language, mathematics, etc.). This decision was taken because we consider this is the most objective approach and the one that presents less errors in its implementation.

Evaluation Metrics



How many selected items are relevant?

$$\text{Precision} = \frac{\text{true positives}}{\text{true positives} + \text{false positives}}$$

How many relevant items are selected?

$$\text{Recall} = \frac{\text{true positives}}{\text{true positives} + \text{false negatives}}$$

Accuracy: is the number of students who got it right (both good and bad) divided by the total number of students.

Precision: is the number of predictions that were successful divided by the number of predictions that were successful plus the false positives

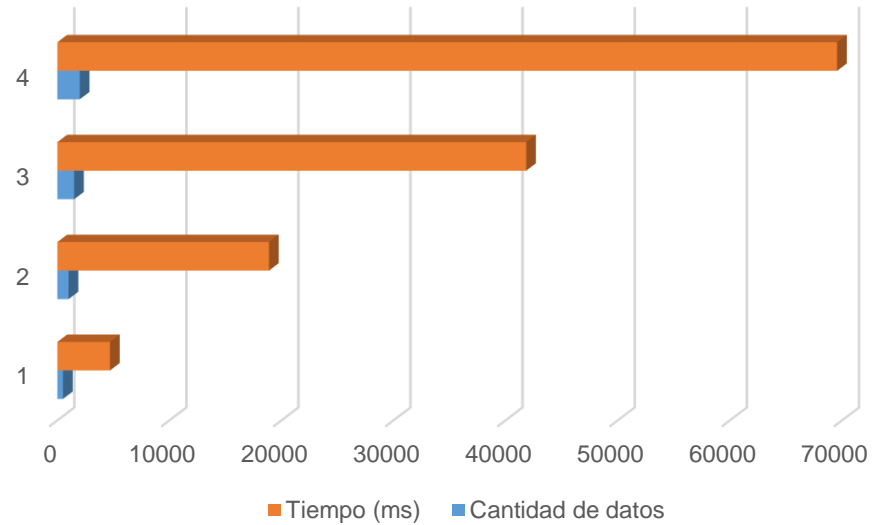
Recall: is the number of predictions that were successful divided by the number of predictions that were successful plus the false negatives

	Training Data	Testing Data
Accuracy	75%	69%
Precision	50.1%	83.55%
Recall	52.5%	52.5%

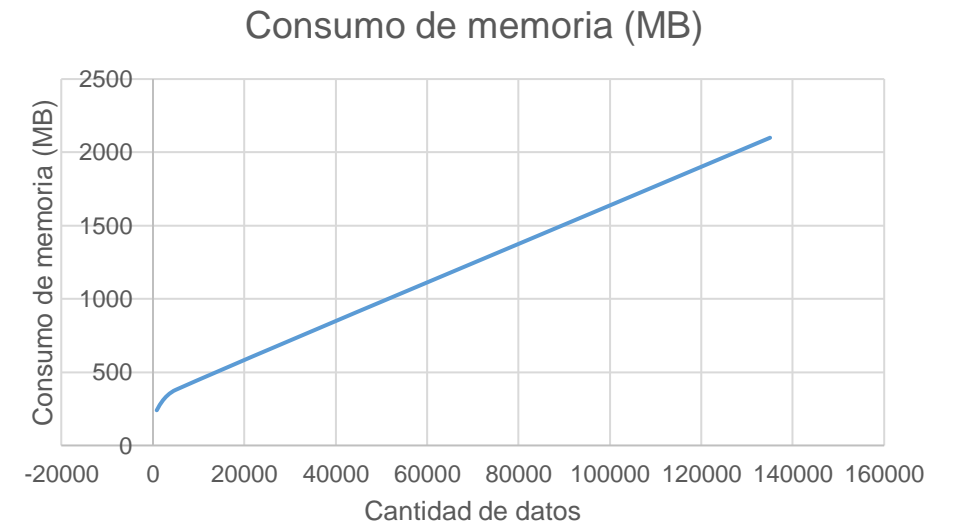
Evaluation metrics using a training dataset of 135,000 students and test dataset of 45,000 students.



Time and Memory Consumption



 Time Consumption



 Memory Consumption



PACKAGE **CLASS** TREE INDEX HELP

ALL CLASSES

SEARCH:

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

Class ElArbolito

java.lang.Object
ElArbolito

```
public class ElArbolito
extends Object
```

Esta clase esta basada en el video "Let's Write a Decision Tree Classifier from Scratch": <https://www.youtube.com/watch?v=LDRbO9a6XPU>

Constructor Summary

Constructors

Constructor	Description
ElArbolito(String[][] raiz)	ElArbolito es un constructor de la clase que recibe una matriz y utiliza las diferente clases de este proyecto para generar el Arbol

Method Summary

All Methods	Instance Methods	Concrete Methods
Modifier and Type	Method	Description
Nodo	CrearArbol(String[][] matriz)	El metodo crearArbol utiliza un llamado recursivo con las dos ramas para asi crear el arbol, sin emabrgo, antes de este llamado hay diferentes condiciones para asegurarse de que no haya error alguno
Nodo	getRaiz()	
int	predecir(String[] estudiante, Nodo nodo)	El metodo predecir por medio de la recursion intenta hacer la prediccion de un estudiante con base en los puntajes logrados
boolean	tieneHoja(Nodo nodo)	

THANKS
AGAIN!