DECISION TREES IN ACADEMIC SUCCESS IN SABER PRO

Santiago Ochoa Castaño
University Eafit
Colombia
sochoac1@eafit.edu.co

Miguel Angel Zapata
University Eafit
Colombia
mazapataj@eafit.edu.co

Miguel Correa
Universidad Eafit
Colombia
macorream@eafit.edu.co

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

ABSTRACT

Considering new strategies to predict academic success during Saber 11 would have beneficial effects in search of improving education system in Colombia. Unfortunately, the technological advances with these purposes have not gone far away in the last years. However, nowadays there are a vast variety of tools and widgets able to do estimations and predictions over different variables. In this case, decision trees will be functional to compare sociodemographic and academic variables related to identify success average on students during Saber 11.

Keywords
Decision trees, machine learning, academic success, standardized student scores, test-score prediction

1. INTRODUCTION

Colombia and technology haven´t had a close relation in terms of education. Development of new widgets, based on data study, would improve the way to estimate possible predictions to find out the main reasons of student dropout. Students during the last course of their bachelor's degree are assessed to qualify their knowledge with a test known as Saber Pro. There are many causes involve in having a score above the required average. Through a specific study about this causes, it is possible to predict academic success in bachelor's degrees in the country.

1.1. Problem

The problem to deal with is to implement an algorithm based in a decision tree adjust to different variables defined in a data structure. According to this, variables are related with sociodemographic and academic information such as: age, parents' income, career, Saber 11 previous scores, gender, among others. Besides, for each student there is a variable which consist in their average score in relation with the total score in Saber Pro.

In this way, the objective to achieve is designing different decision trees concluding prediction statements that define academic success as the probability of a student of having a score above their own previous average.

1.2 Solution

In this work, we focused on decision trees because they provide great explainability. We avoid black-box methods such as neural networks, support-vector machines and random forests because they lack explainability.

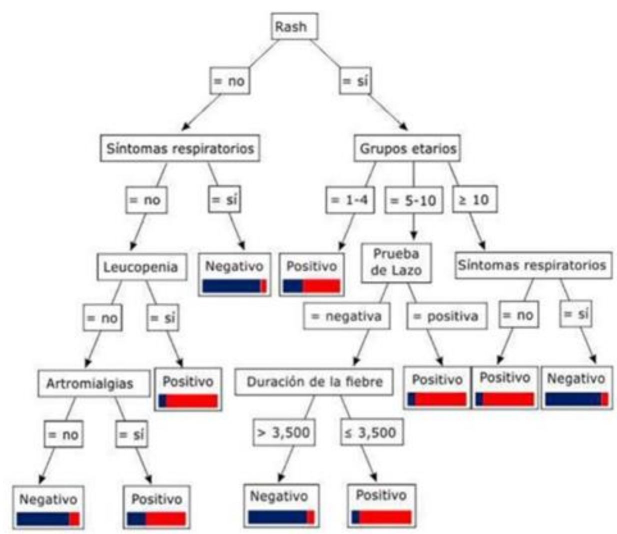
1.3 Article structure

In what follows, in Section 2, we present related work to the problem. Later, in Section 3 we present the datasets and methods used in this research. In Section 4, we present the algorithm design. After, in Section 5, we present the results. Finally, in Section 6, we discuss the results and we propose some future work directions.

2. RELATED WORK

2.1 Decision trees in the dengue diagnosis

Basically, the main purpose was to identify symptoms and signs around a group of patients and concluded they vulnerability to suffer dengue according to decision trees. The study was divided in two kind of variables: 25 numerical and 21 categorical. The accuracy average rounded 80%. Processes were carried out by the software RapidMiner.



Nodes represented a specific rule made up by the antecedent (branches) and the consequent (leaves). Each rule calculated two indicators. The first one meant the percentage of how many times antecedents and consequent were together, and the second pointed out the conditional probability to accomplish a rule.

Tabla 1. Indicadores que apoyan las reglas del *Árbol de síntomas y signos*

Árbol de exámenes de laboratorio							
Reglas	Total casos	Soporte (%)	Casos		Soporte interior de clases		Confianza (%)
			Positivos	Negativos	Positivos (%)	Negativos (%)	
Regla 1**	167	20,1	24	143	5,7	34,4	85,6
Regla 2*	163	19,6	151	12	36,4	2,9	92,6
Regla 3**	233	28,0	4	229	0,9	55,1	98,2
Regla 4*	203	24,4	199	4	47,9	0,9	98,0
Indicio 1**	42	5,0	5	37	1,2	8,9	88,0
Indicio 2*	75	9,0	61	14	14,7	3,4	81,3
Indicio 3**	44	5,3	4	40	0,9	9,6	90,9
Indicio 4*	178	21,44	124	54	29,8	13,0	69,6
Indicio 5*	107	12,8	64	43	15,4	10,4	59,8
Indicio 6*	30	3,6	22	8	5,3	1,9	73,3

Regla/indicio positivo*; regla/indicio negativo**.

Finally, results were evaluated by a confusion matrix to show the algorithm performance. Besides, a diagnostic scale justified the algorithm efficiency.

2.2 Decision trees in violence prevention

In summary, different death patterns were discovered by using decision trees. Consequently, the study used the Cross-industry standard process for data mining (CRISP-DM) dividing the problem in six phases: business understanding, data understanding, data preparation, modelling, evaluation, and deployment. A big amount of information was provided by the Colombian Observatory of Organized Crime and by taking advantage of data mining different decision trees were designed. Decisions trees were implemented using an easy algorithm known as J48 by WEKA data mining tool.

Keeping in mind CRISP-DM phases, the study concluded with different estimations for each death pattern between 2003 until 2013.

2.3 Decision tree to predict academic dropout in a Chilean university

The main objective was to give a deep vision of some reasons that cause students dropout. The university could identify the trigger in the dropout using a decision tree. The study implements the optimization (A method to introduce some parameters with a range and a specific result) to get a certain percentage and to get the maximum depth of the tree. In the process they used three tables: The first one describes the academic dropout, the second one describes the attributes used to the analysis and finally the third one shows the parameters used to express the optimization.

Tabla 1: Deserción académica de la muestra

Deserción	*Año 1*	*Año 2*	*Año 3*	*Año 4*	*Total (%)*
No	1.343	1.050	975	821	4.189 (79)
Si	52	221	397	429	1.099 (21)
Total	1.395	1.271	1.372	1.250	5.288 (100)

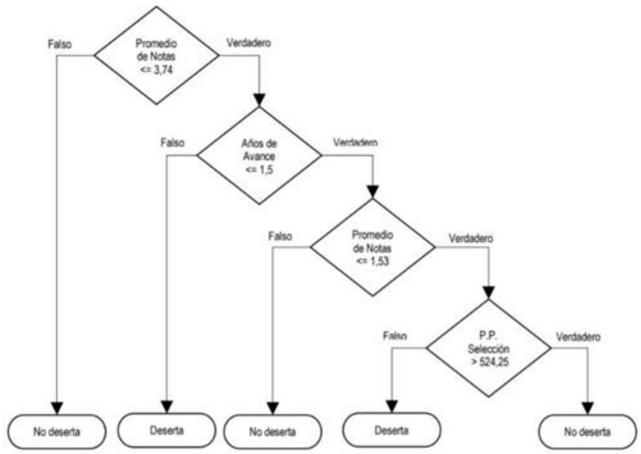
Tabla 2: Atributos para el análisis de la CBAD

Atributo	*Tipo*	*Media*	*Desv. Est.*
Años de Avance	Numérico	2,5	1,1
Edad	Numérico	19,9	2,2
Nivel de Ingreso Familiar (1 a 6)	Numérico	1,4	0,7
Puntaje Prueba de Selección	Numérico	568,9	40,7
Puntaje de Notas Enseñanza Media	Numérico	566,4	85,3
Promedio de Notas	Numérico	4,5	0,9
Desviación Estándar de Notas	Numérico	1,0	0,4
Género		*N*	%
• Femenino	Nominal	2.941	55,6
• Masculino		2.346	44,4
Colegio de Enseñanza Media		*N*	%
• Privado		2.013	38,1
• Público	Nominal	322	6,1
• Subvencionado		2.894	54,7
Deserción		*N*	%
• No	Nominal	4.189	79,2
• Sí		1.099	20,8
Total		5.288	100,0

Tabla 3: Parámetros optimizados

Parámetro	*Rango/Pasos*	*Lista*	*Resultado*
Criterio de selección de atributos para división		Precisión Índice Gini Ratio de Ganancia Ganancia de Información	Índice Gini
Profundidad máxima	De 1 a 20 / 20		16
Nivel de confianza utilizado para el cálculo del error pesimista de la poda	De 0,05 a 0,5 / 9		0,15

The decision tree implemented was based in three factors: Student average, years coursing bachelor's degree and score in the selection test.



Finally, results were evaluated by a confusion matrix to show the performance and the accuracy average rounded the 87.27%.

Tabla 4: Matriz de confusión para la predicción de deserción

		Predicción de Deserción		
		Sí	*No*	*Total*
Deserción Real	*Sí*	172	44	216
	No	158	1.213	1.371
	Total	330	1.257	1.587

2.4. Decision tree to predict Pruebas saber 11

In search to find factors which have a predictive value associated with academic performance across Pruebas Saber 11, different algorithms based on decision trees were carried out. In this way, to proceed along the study was required the CRISP-DM methodology. Furthermore, variables were related with socio-economic, academic, and institutional information. Decision trees were developed using the WEKA data mining tool to identify patterns associated with academic failure or success.

Tabla 4. Matriz de correlaciones de las competencias de las pruebas Saber 11°.

COMPETENCIAS	CIENCIAS NATURALES	INGLÉS	LECTURA CRÍTICA	MATEMÁTICAS	CIUDADANAS	GLOBAL
Ciencias Naturales	1	0,715	0,790	0,825	0,814	0,929
Inglés		1	0,691	0,694	0,691	0,795
Lectura Crítica			1	0,761	0,809	0,905
Matemáticas				1	0,782	0,919
Ciudadanas					1	0,923
Global						1

In conclusion, decision trees were able to show solid information about the different stablished parameters.

3. MATERIALS AND METHODS

In this section, we explain how the data was collected and processed and, after, different solution alternatives considered to choose a decision-tree algorithm.

3.1 Data Collection and Processing

We collected data from the *Colombian Institute for the Promotion of Higher Education* (ICFES), which is available online at ftp.icfes.gov.co. Such data includes anonymized Saber 11 and Saber Pro results. Saber 11 scores of all Colombian high schools graduated from 2008 to 2014 and Saber Pro scores of all Colombian bachelor-degree graduates from 2012 to 2018 were obtained. There were 864,000 records for Saber 11 and records 430,000 for Saber Pro. Both Saber 11 and Saber Pro, included, not only the scores but also socio-economic data from the students, gathered by ICFES, before the test.

In the next step, both datasets were merged using the unique identifier assigned to each student. Therefore, a new dataset that included students that made both standardized tests was created. The size of this new dataset is 212,010 students. After, the binary predictor variable was defined as follows: Does the student score in Saber Pro is higher than the national average of the period?

It was found out that the datasets were not balanced. There were 95,741 students above average and 101,332 students below average. We performed undersampling to balance the dataset to a 50%-50% ratio. After undersampling, the final dataset had 191,412 students.

Finally, to analyze the efficiency and learning rates of our implementation, we randomly created subsets of the main dataset, as shown in Table 1. The dataset was divided into 70% for training and 30% for testing. Datasets are available at https://github.com/mauriciotoro/ST0245-Eafit/tree/master/proyecto/datasets .

	Dataset 1	Dataset 2	Dataset 3	Dataset 4	Dataset 5
Train	15,000	45,000	75,000	105,000	135,000
Test	5,000	15,000	25,000	35,000	45,000

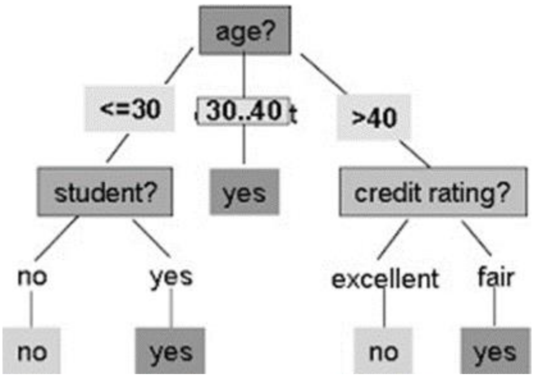
Table 1. Number of students in each dataset used for training and testing.

3.2.1 ID3 Algorithm

It is the most used algorithm in decision trees at present. The ID3 algorithm is simple and powerful at the same time. Its main idea is to approximate discrete values.

The algorithm starts with the first node (root) and keeps generating nodes without using the attribute of the set and determine the entropy (information gain) of the attribute. It continues the process until is no more instance there. The require choosing the attribute is based on choose the largest value of the information gain between the attributes left.

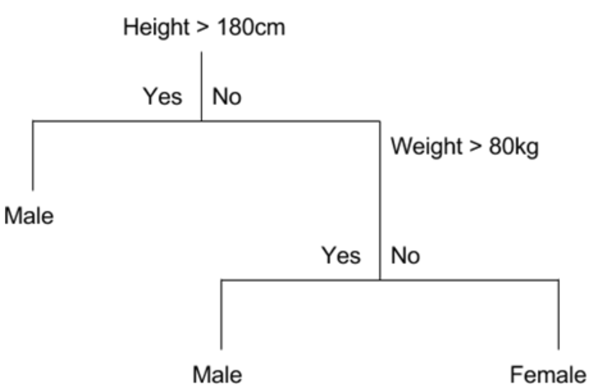
The set need to be order in series, where each of the values are categorized as attributes, being this the objective. The classification is a binary decision (yes or no).



3.2.2 CART (Classification And Regression Trees)

This model allows to have input and output variables such as nominal, ordinal and continuous. This algorithm is used to predict categories of objects and to predict continuous values. At the time of implement the decision tree, the nodes can only be divided in two groups since this is the restriction that binary method imputes. CART uses the Gini index as a measure to identify the node with the greater reduction of impurity and it is used to split the node records.

CART only accept two values: number or categories.



3.2.3. C4.5 Algorithm

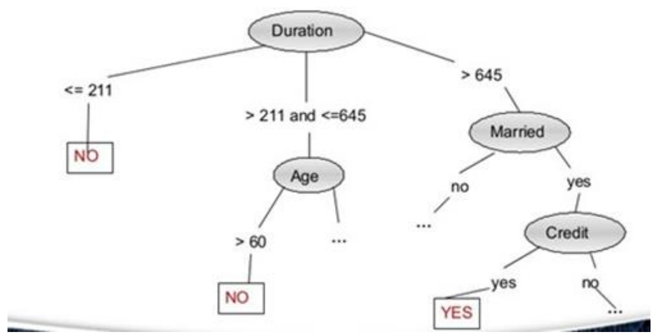
This algorithm is an improved version from the ID3. A decision tree generated (The tree is based in the depth-first strategy) by the C4.5 algorithm is used to for classification (Frequently known as a statistic classifier). The algorithm starts with data distribution performed recursively.

In the process it selects the all samples and identify which can be the most efficiently at the time of divides the samples set in a sort of subsets uses to take decision, basically in each node the system need to decide which sample take to do a data distribution. The entropy criterion (The attribute with more information gain) is used to generate new nodes as ID3 algorithm does. Finally, the attribute with the highest information gain is stablished as the decision parameter.

The C4.5 algorithm propose two tentative tests:

- Standard test: For discrete variables with a specific result and branches for each value.

- Complex test: As the standard test use discrete variables but the result is just for a specific group where the variables are assigned.



3.2.3 CN2 algorithm

It is an algorithm based in the induction system (method uses to classify the variables present in the problem). It collects ideas of ID3 and AQ algorithms. The CN2 algorithm was designed to work with problems where are lack of information and in some time a poor description of the language implemented, basically when the “training data” is imperfect.

The processes begin when it identifies a set of examples and started to do a complex search to stablish the classification rules with the objective of organize all the variables that are present. In the classification the program can adding new conjunctive terms or removing a disjunctive element in one of its selectors. During the learning process the algorithm must take some decisions to evaluate the functions that have given the condition set.

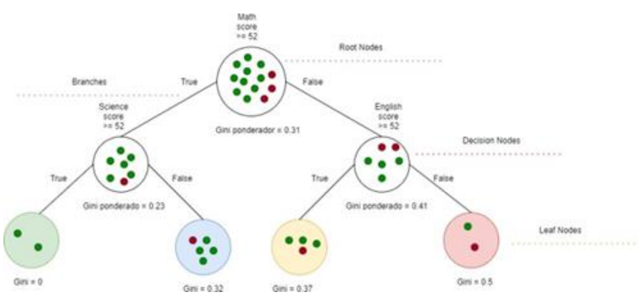
```
procedure CN2unordered(allexamples, classes):
let ruleset = {}
for each class in classes:
   generate rules by CN2ForOneClass(allexamples,class)
   add rules to ruleset
return ruleset.

procedure CN2ForOneClass(examples,class):
let rules = {}
repeat
   call FindBestCondition(examples, class) to find bestcond
   if     bestcond is not null
   then add the rule 'if bestcond then predict class' to rules
      & remove from examples all exs in class covered by bestcond
until bestcond is null
return rules
```

4. ALGORITHM DESIGN AND IMPLEMENTATION

In what follows, we explain the data structure and the algorithms used in this work. The implementation of the data structure and algorithm is available at Github[1].

4.1 Data Structure

The data structure chosen to make predictions about academic success related to Saber Pro were decision trees. This data structure is a method to predict decisions which take advantage of the information provided by the data. In this way, this data structure is capable to learn different decision rules to create an automatic model suitable to analyze options and determine steps in a process. For this reason, these mechanisms are considered a type of algorithm for predictive modeling machine learning used for classification and regression.



The green color indicates a high probability to success in Saber Pro; the blue one shows a lower probability than green, but it is still good; the yellow one represents a lower chance to success, and finally, the read one is considered the worst case.

4.2 Algorithms

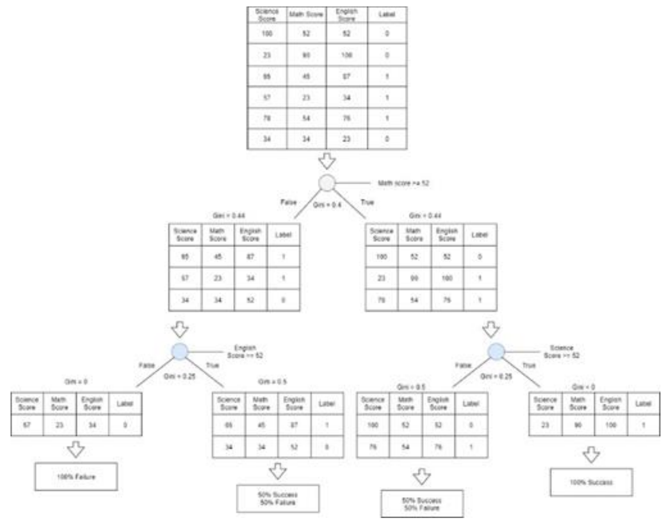
At the hour to understand how is designed the decision tree, there are different concepts to keep in mind. In summary, decision trees are made of nodes. Each of them contains an attribute linked to a series of events. Nodes are classified by:

Root Node: The attribute that provides the greatest amount of information. From there, the tree will begin to develop. In terms of code, it returns a reference to the hold tree.

Branches: Marks the trajectory of events from the root node to the leaf nodes. Throughout the tra- jectory, it takes two paths: True or False.

Internal nodes or decision nodes: Declared attributes with a studied value where data is split by asking a question. In terms of code, it holds a refer- ence to a question, and to two new nodes known as child nodes.

Leaf nodes: Are the most distant nodes and contains the final decisions.



The previous graph allows to figure out the deci- sion tree designed. The idea is to look over from top to bottom since in this way the tree was built. The root node contains the entire set of data, and from there by asking a question which leads to true or false, the set of data will be split in two nodes. Eventually, for each node, there is a new question repeat- ing process in a recursive way. At the time there are not more questions to ask, the next node created will be a leaf node. It rep- resents the final decision predicting the percentage to get each label. In this case, labels mean having success or not in Saber Pro 11.

4.2.1 Training the model

Formulas:

Average Gini:

ni = Number of students in the left

node nD = Number of students in

the right node Ii = Gini of the left

node

ID = Gini e right node

Gini impurity:

$$I_G = 1 - ({P_0}^2 + {P_1}^2)$$

P0 = Student proportion with the 0 label

P1 = Student proportion with the 1 label

Labels:

0 = Indicate that the student is below average

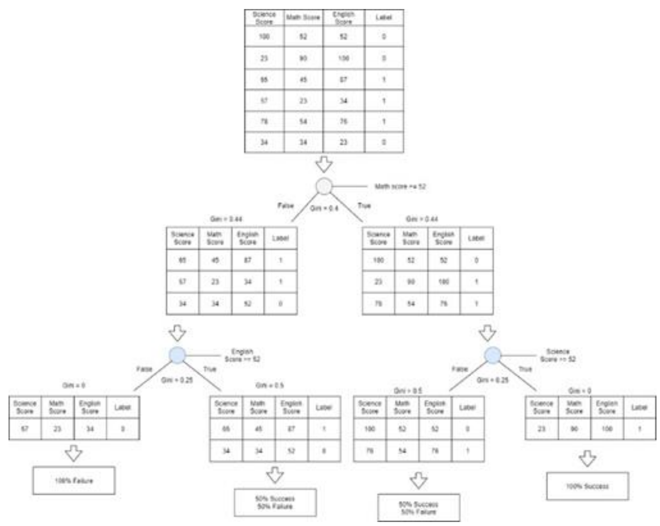
1 = Indicate that the student is above average

The most important thing to train the tree is the way to take approach of data given by the training data set. In this case, different methods and mathematical concepts were used to simplify this task.

Step by step:

In the first place, there is a function in charge of building the tree. It receives the entire training set as input, and as output it will return a reference to the root node of our tree. Then, it starts by adding the entire training set to an initial node (root node), and from there, the idea is to find the best question to ask at this node. The best question is the one that reduces our uncertainty the most. To do that, it is ap- plied two calculations known as Gini impurity and Infor- mation gain. The first one, allows quantify how much un- certainty there is at a node. The second one, allows quantify how much a question reduces that. Let's work on impurity first. It is a metric that ranges between 0 and 1 where lower values indicate less uncertainty, or mixing, at node. The main proposed is to quantify the chance of being incorrect by assigning a randomly label to an example in the same set. If there are not mixing, the impurity is zero. Now, in relation with the information gain, it lets find the question that reduces the uncertainty the most, and it is just a number that describes how much a question helps to unmix the labels at a node. In order to do that, the algorithm begins by calculating the uncertainty of the starting set. Then, for each possible question to ask for, the idea is to try partitioning the data and calculating the uncertainty of the child nodes that result. After that, using the previous uncertainty of the two child nodes must be taken a weighted average of both. Eventually, this result will subtract from the starting uncer- tainty, and that what information gain means. As the algo- rithm goes, it will keep track of the question that produces the most gain, and that will be the best one to ask at this node. Once more the tree is split, the function will be called again to add new child nodes. When there are not more further questions to ask, the information gain will be zero and the new node becomes a leaf. It will predict that an example is either a label or other. This process will be re- peat it again and again until not finding more questions to ask for splitting the data.

4.2.2 Testing algorithm



For giving a brief explanation of how this algorithm predict academic success in Saber Pro by asking different ques- tions which leads to a label, two randomly students from the training data set were chosen in order to compare how differ the label given by the algorithm and the stablished one.

Example of a student that will get a score above the average:

Science Score	Math Score	English Score	Succes
52	52	52	1

After the process the algorithm determinates success as 1. matching with the training data set.

punt_matem	punt_biologi	punt_quimic	punt_fisica	punt_ciencia	punt_filosofi	punt_ingles	desemp_ingl	profundiza	puntaje_pro	desemp_pro	exito
47.0	45.0	41.0	37.0	46.0	40.0	56.0	A2	PUNT_PROFU	6.0	II	0
35.0	41.0	46.0	47.0	40.0	34.0	42.0	A-	PUNT_INTER	41.0		0
49.0	46.0	53.0	41.0	44.0	45.0	45.0	A1	PUNT_INTER	51.0		0
46.0	49.0	35.0	42.0	36.0	41.0	43.0	A-	PUNT_INTER	51.0		0
52.0	55.0	50.0	46.0	52.0	44.0	52.0	A1	PUNT_PROFU	6.0	II	1

Example of a student that will get a score below the average:

Science Score	Math Score	English Score	Succes
46	47	56	0

After the process the algorithm determinates a probability of 50% failure and 50% of success in the test.

punt_matem	punt_biologi	punt_quimic	punt_fisica	punt_ciencia	punt_filosofi	punt_ingles	desemp_ingl	profundiza	puntaje_pro	desemp_pro	exito
47.0	45.0	41.0	37.0	46.0	40.0	56.0	A2	PUNT_PROFU	6.0	II	0
35.0	41.0	46.0	47.0	40.0	34.0	42.0	A-	PUNT_INTER	41.0		0
49.0	46.0	53.0	41.0	44.0	45.0	45.0	A1	PUNT_INTER	51.0		0
46.0	49.0	35.0	42.0	36.0	41.0	43.0	A-	PUNT_INTER	51.0		0

After the procees the algothm determinates success as 0. Matching with the training data set, but has the posisibility of getting the incorrect labe, since according to the algotithm there is a possibility of 50% to get the right label and 50% to get the incorrect label.

4.3 Complexity analysis of the algorithms

To calculate complexities of the training and testing algorithms for the worst-case using O notation, it´s important to understand that the variable N represents the number of rows and M represents the number of columns of a matrix which contains the training dataset to build the tree. The worst case would be start by iterating M columns in order to find which variable separates the data obtaining the highest information gain. In this way, for each value in each column the information gain would be calculated, that is, it iterates N rows assuming that all the values are different. Internally, to calculate the information gain, the matrix must be separated in two, which implies iterating again N rows. According to this, each iteration would be nested with another concluding N^2*M.

In other hand, the algorithm is based on a binary tree, that is, it is divided in two. Thus, in the worst case, the number of subdivisions that it would have will be M columns. This involves 2^M operations. For this reason, O notation to train the decision tree would be $O(N^2 * M*2^M)$. In another case, to calculates the complexity of validating the decision tree for the worst case, it would be to go through N rows M times using the decision tree created, to check the result that each person will have within the dataset.

Algorithm	Time Complexity
Train the decision tree	$O(N^2 * M*2^M)$
Test the decision tree	O(N*M)

Table 2: Time Complexity of the training and testing algorithms. The variable N represents the number of rows and M represents the number of columns of a matrix which contains the training dataset to build the tree.

Memory complexity of the training algorithms is $O(N*M* 2^M)$. This is because to the worst case where each division would be creating two new matrixes with the size of the input matrix, that is, N*M. Besides, the algorithm is based on a binary tree, that is, it is divided in two. Thus, in the worst case, the number of subdivisions that it would have will be M columns. This involves 2^M operations. For this reason, O notation to train the decision tree in terms of memory would be $O(N * M*2^M)$.

In other hand, to calculates the memory complexity of testing for the worst case would be O(1). That means the decision tree and the required elements to validate have been already created.

Algorithm	Memory Complexity
Train the decision tree	$O(N*M* 2^M)$
Test the decision tree	O(1)

Table 3: Memory Complexity of the training and testing algorithms. The variable N represents the number of rows and M represents the number of columns of a matrix which contains the training dataset to build the tree.

4.4 Design criteria of the algorithm

The algorithm CART was designed in that way because it gives a clear procedure to take decisions, and it allows to know which question to ask and when to produce the purest possible distribution of the labels at each node. Thanks to the different precise metrics such as Gini impurity, which quantify the amount of uncertainty at single node, or information gain that quantify how much a question reduces that uncertainty, it´s possible to select the best question to ask at each point. And given that question, the tree will be built recursively.

5. RESULTS

5.1 Model evaluation

In this section, we present some metrics to evaluate the model. Accuracy is the ratio of number of correct predictions to the total number of input samples. Precision. is the ratio of successful students identified correctly by the model to successful students identified by the model. Finally, Recall is the ratio of successful students identified correctly by the model to successful students in the dataset.

5.1.1 Evaluation on training datasets

In what follows, we present the evaluation metrics for the training datasets in Table 3.

	Dataset 1	***Dataset 2***	***Dataset 5***
Accuracy	0.77	0.78	0.78
Precision	0.76	0.75	0.75
Recall	0.78	0.79	0.8

Table 3. Model evaluation on the training datasets.

5.1.2 Evaluation on test datasets

In what follows, we present the evaluation metrics for the test datasets in Table 4.

	Dataset 1	*Dataset 2*	*Dataset 5*
Accuracy	0.78	0.78	0.77
Precision	0.77	0.76	0.76
Recall	0.77	0.8	0.8

Table 4. Model evaluation on the test datasets.

5.2 Execution times

	Dataset 1	*Dataset 2*	*...Dataset n*
Training time	2.2 mins	6.3 mins	15.1
Testing time	47 s	2.5 mins	7.8 mins

Table 5: Execution time of the *(Please write the name of the algorithm, C4.5, ID3)* algorithm for different datasets.

5.3 Memory consumption

We present memory consumption of the binary decision tree, for different datasets, in Table 6.

	Dataset 1	*Dataset 2*	*Dataset 5*
Memory consumption	20 MB	56MB	150 MB

Table 6: Memory consumption of the binary decision tree for different datasets.

6. DISCUSSION OF THE RESULTS

Explain the results obtained. Is precision, accuracy and sensibility appropriate for this problem? Is the model overfitting? Is memory consumption and time consumption appropriate? *(In this semester, according to the results, can this be applied to give scholarships or to help students with low probability of success? For which one is better?)*

In relation with precision, accuracy and sensibility the algorithm proved to be effective. The margin of error is not considerably high, and it is close to the related works. In this way, the problem could be solved successfully. In terms of overfitting neither the training datasets nor the testing datasets had this problem because both kept the same statistics. Furthermore, the average execution time calculated for each dataset proved to be moderately efficient, although it can be improved by avoiding not having to divide (left / right) for each possible value of the treeset which causes O (N ^ 2). On the other hand, memory consumption of the binary decision tree was higher because average execution was essential in the project.

6.1 Future works

The algorithm presented could have some improvements and some of them are the following:

The algorithm needs to be optimized, providing a better time complexity since internally it performs repetitive processes causing in increase time execution, but it can be simpler providing a better time execution.

Other point that needs to be improved is the Memory complexity since this can cause several disadvantages at the time of run the program.

ACKNOWLEDGEMENTS

The acknowledgements are directed to the teacher of *Data structure and algorithms* I of EAFIT, that helps us in every step throughout the semester, solving questions that were essential to be able to develop the algorithm. He guided us in the best way.

We appreciate the job of Santiago Ochoa Castaño and Miguel Angel Zapata since the project was done with great effort and the results achieved were optimal. Given a good idea of what the algorithm purpose was.

REFERENCES

1. Acosta, J., Oller, L., Sokol, N., Balado, Sardiñas, J., Montero, D., Balado, Sansón, R. and Sardiñas,

M.E. Revista Cubana de Pediatría, 88(4). 441-453. http://scielo.sld.cu/scielo.php?script=sci_arttext&pid=S0034-75312016000400005&lng=en&tlng=en

2. Clark. P and Niblett. T. 1989 The CN2 induction algorithm. The Turing Institute, Kluwer Academic Publishers, United Kingdom. https://link.springer.com/content/pdf/10.1023/A:1022641700528.pdf

3.Timarán, R., Calderón, A. and Hidalgo, A. Universidad y Salud. 19(3). 359-365.

DOI: http://dx.doi.org/10.22267/rus.171903.101

4. Timarán, R., Calcedo, J. and Hidalgo,A. Rev.investig.desarrollo.innov.9(2).363-378.

DOI: 10.19053/20278306.v9.n2.2019.9184

5. Martinez, Z. and Menendez, J. 2004. Predicción de crisis empresariales en seguros no-vida. una aplicación del algoritmo See5. Universidad Complutense. Spain.

https://eprints.ucm.es/6834/1/04010.pdf

6. Ramírez, P.E. and Grandón, E.E. Formación Universitaria. 11(3), 3-10.

https://scielo.conicyt.cl/scielo.php?script=sci_arttext&pid=S0718-50062018000300003&lng=en&tlng=en

7. Wikipedia. 2019. ID3 algortihm. (22 May 2019). https://es.wikipedia.org/wiki/Algoritmo_ID3