ALGORITHM FOR DETERMINING THE FACTORS OF ACADEMIC SUCCESS IN THE SABER TESTS

Kevin Alejandro Sossa Chavarria
Universidad Eafit
Colombia
kasossac@eafit.edu.co

Dixon Andres Calderon Ortega
Universidad Eafit
Venezuela
dacalderoo@eafit.edu.co

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

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

ABSTRACT

With this project we seek to learn the keys that makes students succeed in the saber tests, the importance behind this task is that knowing what makes a student succeed we can take actions so the percentage of the students that have a mark above the average become higher. But first of all, we need to achieve the estimate, and then with could use that data to help the education in the country in many ways.

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

1. INTRODUCTION

By having a decision tree-based algorithm that define the keys that leads a student to the academic success, many areas will be profited, starting by the students themselves, and continuing by an increase in the human resources of the country.

Also, by making an estimate of the people that succeed in the tests we can also have and estimate on how many students will access the higher education in the country making us aware of the resources that will be taken for that purpose.

1.1. Problem

The presented problem is to make an algorithm that is capable of recollect the information of the students that take the saber test, determine the variables that affect the succeed rate in the population and by taking that in mind make an estimate in the probability of academic success of a student. To make that happen we have to consider things such as grades for subject, social-economic state and many other variables.

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

We will present 4 articles related to the problem descrived in section 1.1.

3.1 Analysis and Predictions on Students' Behavior Using Decision Trees in Weka Environment

Bresfelean worked is based in the data collected through the surveys from senior undergraduate students at the faculty of economics & Business administration in Cluj-Napoca [1]. Decision tree algorithms in the WEKA tool, ID3 and J48 were applied to predict which students are likely to continue their education with the postgraduate degree. The model was applied on two different specializations students' data and an accuracy of 88.68 % and 71.74 % was achieved with C4.5.

3.2 A Comparative Analysis of Techniques for Predicting Academic Performance

Nghe, Janecek, and Haddawy [2] compared the accuracy of decision tree and Bayesian network algorithms for predicting the academic performance of undergraduate and postgraduate students at two very different academic institutes: Can Tho University (CTU), a large national university in Viet Nam, and the Asian Institute of Technology (AIT), international university in Thailand. It was found that decision trees are 3-12% more accurate than Bayesian Networks.

3.3 Early Prediction of Student Success: Mining Students Enrolment Data

Z. J. Kovacic presented a case study on educational data mining in [3] to identify up to what extent the enrolment data can be used to predict student's success. The algorithms CHAID and CART were applied on student enrolment data of information system students of open polytechnic of New Zealand to get two decision trees classifying successful and unsuccessful students. The accuracy obtained with CHAID was 59.4% and CART was 60.5%.

3.4 A CHAID Based Performance Prediction Model in Educational Data Mining

M. Ramaswami and R. Bhaskaran [4] used the CHAID algorithm to analyze the interrelation between variables to predict the outcome on the performance at higher secondary school education in India. The variables like medium of instruction, marks obtained in secondary education, location of school, living area and type of secondary education were the strongest indicators for the student success in higher secondary education. This CHAID prediction model of student performance was constructed with seven class predictor variables with accuracy 44.69%.

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 Decision-tree algorithm alternatives

In what follows, we present different algorithms to solve to automatically build a binary decision tree.

3.2.1 C4.5 Algorithm:

C4.5 is an algorithm used to generate a decision tree developed by Ross Quinlan, and it's an improvement from his own ID3 algorithm. It uses information to gain ratio selecting the best attribute to avoid features with many values that occurs.

C4.5 works using the top down strategy based on the divide and conquer approach to construct the decision tree induction algorithms. It maps the training set and uses the information gain ratio as a measurement to select splitting attributes and generates nodes from the root to the leaves. Every illustrating path from the root node to the leaf node forms a decision rule to determine which the class of a new instance is.

One perfect example is a method that defines the class of an animal, either is a mammal, a fish, an amphibian, a reptile or a bird. This method uses the following C4.5 rules:

(Give Birth= No, Can Fly= Yes) -> Birds

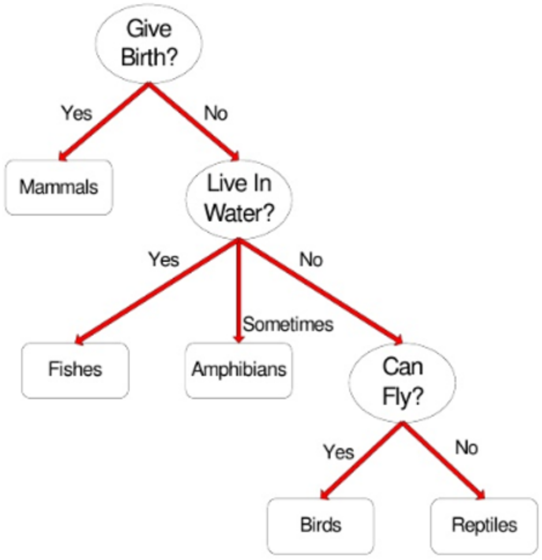
(Give Birth=No, Live in Water=Yes) -> Fishes

(Give Birth=Yes) -> Mammals

(Give Birth=No, Can Fly=No, Live in Water=No) -> Reptiles

() -> amphibians

This way the sequential covering generates a rule by adding tests that maximize rule´s accuracy, and each new test reduces the rule’s coverage.[5]



3.2 CHAID Algorithm:

The uses of the CHAID algorithm allow that with a very good segmentation of the data, they can tell with accuracy the information that they are searching. Also, the display of the information will be good for the users understanding.[6]

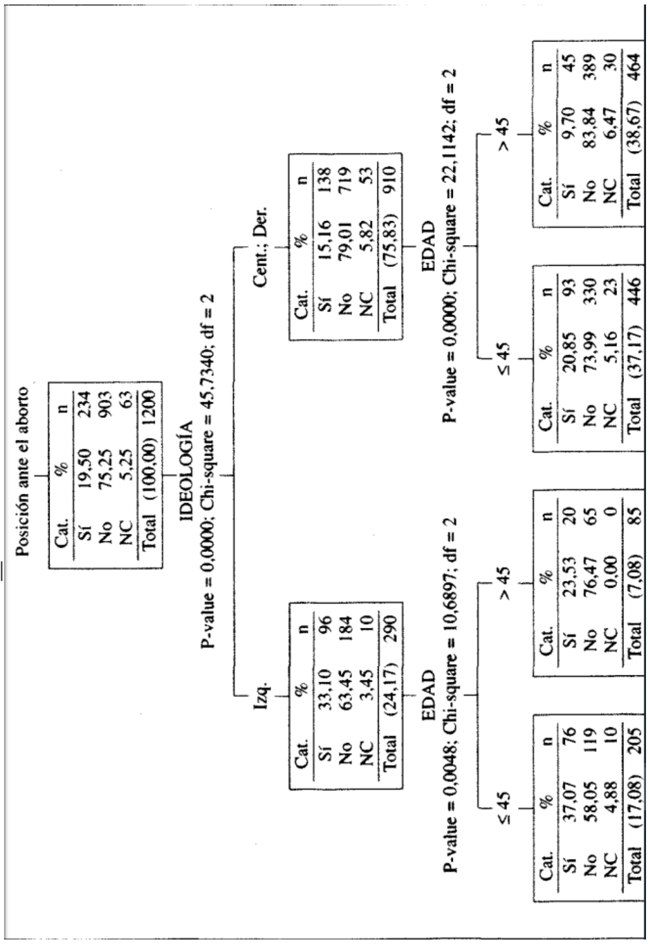
So for what we know, CHAID is mostly used for statics in surveys. More likely, they are good to find common variables that define the stereotypes of the people. In conclusion, CHAID can find groups of thinkers or a conduct in a social group of people, and it’s likely to be used in statistics of population.

Considering this example, in which we want to find what’s the people opinion on abortion, we have three answers: in favor, opposing or does not know. But for the purpose of the CHAID algorithm and the sake of the precision we will also add range of age(more than 45 or lesser) and their position in political thinking(communist, center or capitalist),also center and capitalist we will be considered together to show a different example. So when we have all this data we can form a decision tree like the one below.[6]

3.3 ID3 Algorithm:

ID3 algorithm is based in a decision tree solution. This decision tree learning is a procedure for calculating the target value having a discrete function. The function that has been learned is symbolized by a decision tree. For the inductive inference the decision tree learning is one of the most commonly and broadly used methods which are practical in nature.[7]

Remarkably we can say when this decision tree is formed. With ID3 we also want a very simple solution so we will only


Posición ante el aborto
IDEOLOGÍA
P-value = 0,0000; Chi-square = 45,7340; df = 2
Izq.
Cent.; Der.
EDAD
P-value = 0,0048; Chi-square = 10,6897; df = 2
EDAD
P-value = 0,0000; Chi-square = 22,1142; df = 2
≤ 45
> 45
≤ 45
> 45

Cat.	%	n
Sí	19,50	234
No	75,25	903
NC	5,25	63
Total	(100,00)	1200

Cat.	%	n
Sí	33,10	96
No	63,45	184
NC	3,45	10
Total	(24,17)	290

Cat.	%	n
Sí	15,16	138
No	79,01	719
NC	5,82	53
Total	(75,83)	910

Cat.	%	n
Sí	37,07	76
No	58,05	119
NC	4,88	10
Total	(17,08)	205

Cat.	%	n
Sí	23,53	20
No	76,47	65
NC	0,00	0
Total	(7,08)	85

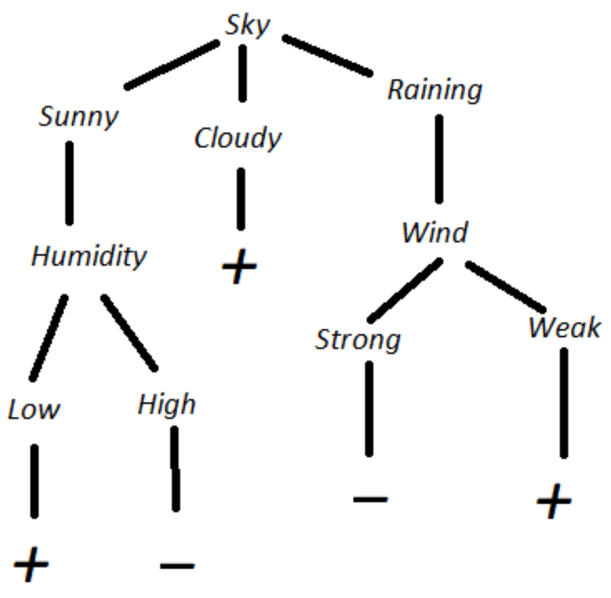
Cat.	%	n
Sí	20,85	93
No	73,99	330
NC	5,16	23
Total	(37,17)	446

Cat.	%	n
Sí	9,70	45
No	83,84	389
NC	6,47	30
Total	(38,67)	464

find a tuple of values, like yes or no, at the end of a decision tree formed with this procedure. but this type of procedure will help, most likely, to yes or no data, or when the information that we want to give is very precise.

We will consider this example. We want to play tennis outside various days in the weekend, but the sky has a very strange behaviour, so you decide to recompile some data to know which days are good to go and play tennis(+) or what days are not so good(-). So, the first thing to consider is how the sky looks. If it is sunny, cloudy or rainy. if the day is sunny we have to consider the humidity, if it’s low we’ll go(+), if it’s high we’ll go other day(-). If the day is cloudy we’ll go no matter what(+). Finally, if the day is rainy, we’ll consider the wind, if it’s strong we’ll go other day(-), but if it’s low we’ll give it a try(+).

With all the cases now exposed to simple variables and tuples of values we can implement ID3 algorithm as expected and we will have something like the decision tree below, that will help us sort the data that we were given to how likely are we going to play tennis.



3.4 CN2 Algorithm:

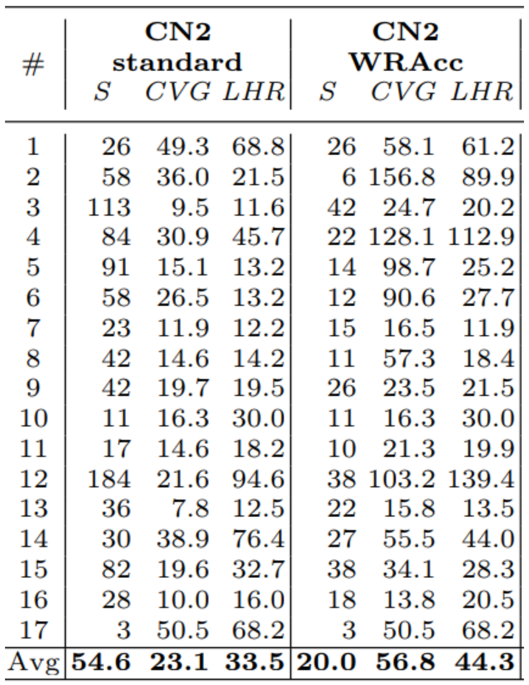
The CN2 algorithm is based in a machine learning pattern of rule induction, that is when the rules are extracted from the visualization of the machine observations that it does of himself. Also, this algorithm has the ability to work even when the data is imperfect.

To train the CN2 algorithm first we have to set a list of instructions to tell what we want to show for the final display. This will generate classification rules, and then the computer will decide whether or not they can be applied, obtaining a result for the final classification and finally this creates the decision tree.

Finally, we have to say that this algorithm does not display the data with a decision tree like we see in other kind of examples. However, it can manage the same amount of information like the other algorithms. Also, the display that it uses is similar to the AQ Algorithm.

Remarkably, the advantage of the proposed approach is that each rule with high weighted accuracy represents a 'chunk' of knowledge about the problem, due to the appropriate tradeoff between accuracy and coverage, achieved through the use of the weighted relative accuracy heuristic.[8]

The example that we are going to explain is to know the weight of some objects. Also we want to know the average between other objects and the height with the interpretation of the mass they have, so the algorithm shows the number of data that they acquired with all this people and also shows the differences between the other processes of the CN2 algorithm that exist and can be used to get better results.[8]



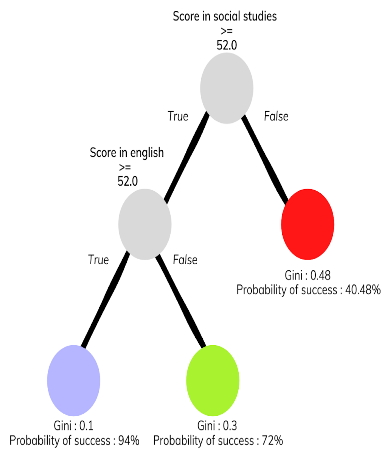
#	CN2 standard			CN2 WRAcc		
	S	*CVG*	*LHR*	*S*	*CVG*	*LHR*
1	26	49.3	68.8	26	58.1	61.2
2	58	36.0	21.5	6	156.8	89.9
3	113	9.5	11.6	42	24.7	20.2
4	84	30.9	45.7	22	128.1	112.9
5	91	15.1	13.2	14	98.7	25.2
6	58	26.5	13.2	12	90.6	27.7
7	23	11.9	12.2	15	16.5	11.9
8	42	14.6	14.2	11	57.3	18.4
9	42	19.7	19.5	26	23.5	21.5
10	11	16.3	30.0	11	16.3	30.0
11	17	14.6	18.2	10	21.3	19.9
12	184	21.6	94.6	38	103.2	139.4
13	36	7.8	12.5	22	15.8	13.5
14	30	38.9	76.4	27	55.5	44.0
15	82	19.6	32.7	38	34.1	28.3
16	28	10.0	16.0	18	13.8	20.5
17	3	50.5	68.2	3	50.5	68.2
Avg	**54.6**	**23.1**	**33.5**	**20.0**	**56.8**	**44.3**

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 Binary Decision Tree

A binary decision tree is a data structure based on a sequential decision process starting from the root being this divided in two nodes, repeating that process until you finish the Booleans operation and come out with an answer.



[1]http://www.github.com/ ????????? /proyecto/

Figure 1: A binary decision tree to predict Saber Pro based on the results of Saber 11. Violet nodes represent those with a high probability of success, green medium probability and red a low probability of success.

4.2 Algorithms

Our algorithm takes the existent data base of students that have presented the test in the past, dividing it in two by looking into which student achieve academic success. By doing that we can start to compare the variables that affect the succeed by calculating the Gini impurity. So then when the tree is done, we can know which variables are making the general impurity higher so we can optimize the three.

4.2.1 Training the model

To train our model we put all the information we have disponible so the algorithm can start to, step by step, make the binary decision three. By making that our algorithm is training our model. While we put data in the algorithm our model is becoming more precise, because it is calculating the impurity of every variable disponible so we can optimize it, training it to be better.

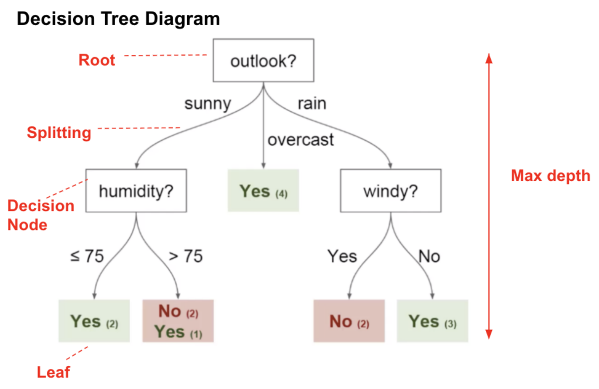


Figure 2: In this example, we show a model to predict whether or not to play Golf, according to weather.

4.2.2 Testing algorithm

After the three is build and we put new data (the information of a student who will present the Saber Test) in, the algorithm takes that data so it can do its own path in the binary conditions pre-seted in the three and optimized beforehand. By doing that the algorithm not only making itself more accurate but giving an answer to the question, which are my possibilities of academic success in the test?

4.3 Complexity analysis of the algorithms

In this part of the report we are going to take a look

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

Table 2: Time Complexity of the training and testing algorithms. *N is number of students, M number of rows, 2^M is number of nodes.*

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

Table 3: Memory Complexity of the training and testing algorithms. *N is number of students, M number of rows, 2^M is number of nodes.*

4.4 Design criteria of the algorithm

First of all we take on count the fact that most decision trees are pretty deep, so we thought that if we make not very deep, but very wide (many nodes) will be a very good way to avoid time complexity, furthermore we still do not want to overload it, so instead of giving the nodes a value, like, if English grade is equal to 3, we decide to manage the variables by ranges, so we can have a wide tree which we can take advantage of while not being counterproductive.

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. By doing that we also prioritize, time running over memory usage.

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 n***
Accuracy	0.99	0.99	0.99
Precision	0.99	0.99	0.99
Recall	0.98	0.98	0.98

Table 3. Model evaluation on the training datasets.

5.1.2 Evaluation on test datasets

In the next section we present the evaluation metrics for the test datasets in Table 4.

	Dataset 1	***Dataset 2***	***Dataset 3***
Accuracy	0.79	0.79	0.8
Precision	0.77	0.78	0.79
Recall	0.8	0.8	0.81

Table 4. Model evaluation on the test datasets.

5.2 Execution times

Compute execution time for each dataset in github. Measure execution time 100 times for each dataset and report average execution time for each dataset.

	Dataset 1	***Dataset 2***	***Dataset 3***
Training time	4 s	6 s	8 s
Testing time	4 s	6 s	9 s

Table 5: Execution time of the *CART* 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 n***
Memory consumption	480 MB	669 MB	862MB

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

6. DISCUSSION OF THE RESULTS

We achieve a very good results, the precisions and sensibility are appropriate for this problem. We can see many uses of this decision three, we think mainly it could help the people who are not very sure if they have been working on having a good grade enough. We think that giving up a little in memory was worth it considering the time of execution we get.

6.1 Future work

In the future we would like to improve our memory usage and still having such a good execution time. Another goal we have to achieve is to make the algorithm have a better precisión so it can be used for more formal things.

REFERENCES

1. V. P. Bresfelean, "Analysis and Predictions on Students' Behavior Using Decision Trees in Weka Environment", Proceedings of the ITI 2007 29th Int. Conf. on Information Technology Interfaces, June 25-28, 2007.

2. N. Thai Nghe, P. Janecek, and P. Haddawy, "A Comparative Analysis of Techniques for Predicting Academic Performance", 37th ASEE/IEEE Frontiers in Education Conference, October 2007.

3. Z. J. Kovacic, "Early prediction of student success: Mining student enrollment data", Proceedings of Informing Science & IT Education Conference (InSITE) 2010.

4. M. Ramaswami and R. Bhaskaran, "A CHAID based performance prediction model in educational data mining", IJCSI International Journal of Computer Science Issues, Vol. 7, Issue 1, No. 1, January 2010.

5. Lanzi, P. 2007. Machine Learning and Data Mining: 12 Classification Rules.

6. Escobar, M. 1998. Las aplicaciones del análisis de segmentación: el procedimiento CHAID.

7. BHARDWAJ, Rupali; VATTA, Sonia. Implementation of ID3 algorithm. International Journal of Advanced Research in Computer Science and Software Engineering, 2013, vol. 3, no 6.

8. LAVRAC, Nada, et al. Rule induction for subgroup discovery with CN2-SD. En Proceedings of the 2nd international workshop on integration and collaboration aspects of data mining, decision support and meta-learning. 2002. p. 77-87.