



DECISION TREE





INDICE

- X Objetivos
- X Resumen
- X Casos Base
- X Pseudocodigo
- X Fases de entrenamiento,
- X Fase de Testeo
- X Análisis Resultados
- X Mejoras Algoritmo
- X Problemas
- X Conclusiones
- X GitHub
- X Bibliografía



OBJETIVO

- X Construir un árbol de decisiones para clasificar la clase a la que pertenecen un conjunto de datos. En nuestro caso predecir si un personaje literario esta vivo o muerto.



RESUMEN

El algoritmo construye un árbol binario de decisiones a partir de un conjunto de datos de entrenamiento, basándose en la entropía de la información.

Los datos de entrenamiento es un conjunto $S = \{s_1, s_2, \dots, s_n\}$ que ya están clasificados.

Cada muestra s_i es un vector de dimensión p tal que $(x_{1,i}, x_{2,i}, \dots, x_{p,i})$ donde x_j representa los valores de los atributos y la clase a la que pertenece s_i .

En cada nodo del árbol, el algoritmo escoge el atributo que mejor clasifica el conjunto de muestras. El criterio utilizado para ello es la ganancia de información.



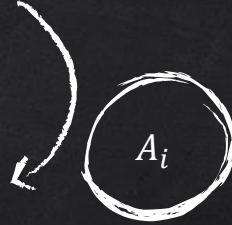
CASOS BASE

1. Todas las muestras de la tabla pertenecen a la misma clase:
 - Se marca el nodo como hoja y se elige la clase
2. La ganancia de información del atributo escogido es menor que un umbral establecido por nosotros:
 - Se marca el nodo como hoja y se elige la clase mas frecuente dentro del conjunto de muestras
3. El conjunto de muestras esta vacío:
 - Se devuelve NULL



CASO BASE 1

I \ J	A_1	A_2	A_3	Clase
Fila 0	1	2	3	X
Fila 3	1	0	1	1
Fila 4	0	1	1	1
Fila 6	0	0	1	1



Atributo: i
Umbral: -1
Clase: 1, Vivo
Hoja: True
Izquierda: NULL
Derecha: NULL

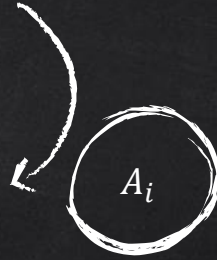


VIVO



CASO BASE 2

$I \setminus J$	A_1	A_2	A_3	Clase
Fila 0	1	2	3	X
Fila 3	1	0	1	1
Fila 4	0	1	1	0
Fila 6	0	0	1	0



Atributo: i
Umbral: -1
Clase: 0, Muerto
Hoja: True
Izquierda: NULL
Derecha: NULL




MUERTO

$$\max(h(A_i)) < \mu$$

$$\max(h(A_i)) = 0.009 < \mu = 0.01$$



CASO BASE 3



I \ J	A_0	A_1	A_2	CLASE
FILA 0	1	1	2	X
FILA 1	1	0	0	1
FILA 2	1	1	0	0
FILA 3	1	1	0	1



I \ J	A_0	A_1	A_2	CLASE
FILA 0	1	1	2	X
FILA 1	1	0	0	1
FILA 2	1	1	0	0
FILA 3	1	1	0	1

I \ J	CLASE
FILA 0	X
FILA 1	1
FILA 2	0
FILA 3	1



Dev **NULL**



PSEUDOCODIGO

1) *Comprobar casos base*

2) *Umbralizar(A_i)*

3) *Para cada atributo a calcular la ganancia de información*

4) *Sea a_best el atributo con mayor ganancia de información*

5) *Si la ganancia de información de a_best es menor que el umbral de ganancia \rightarrow Caso Base 2*

6) *Nodo \rightarrow atributo = a_best*

7) *nuevoNodoIzquierda*

7.1) $T_1 = \text{filtrarTabla}(C, a_best, 0)$

7.2) *nodo \rightarrow izq = construirArbol(T_1 , nuevoNodoIzquierda)*

8) *nuevoNodoDerecha*

8.1) $T_2 = \text{filtrarTabla}(C, a_best, 1)$

8.2) *nodo \rightarrow der = construirArbol(T_2 , nuevoNodoDerecha)*



FASES ÁRBOL DE DECISIÓN



FASE ENTRENAMIENTO

X Input: datos de entrenamiento clasificados

X Output: árbol de decisiones



INFORMACIÓN NODO

A_i

- Atributo: contiene el atributo por el que se va a dividir
- Umbral del atributo: guarda el umbral de los atributos continuos
- Clase: clase a la que pertenece el nodo
- Hoja: indica que es hoja y por tanto, se ha tomado una decisión
- Nodo Izquierda: dirección de memoria del nodo izquierdo
- Nodo Derecha: dirección de memoria del nodo derecho

ALMACENAMIENTO DE DATOS

Numero de filas + 1

I \ J	A ₀	A ₁	A ₂	A ₃	CLASE
FILA 0	0	1	2	3	X
FILA 1	1	0	0	1	1
FILA 2	1	1	0	0	0
FILA 3	0	1	0	1	1
FILA 4	0	0	1	1	1
FILA 5	1	1	1	0	0
FILA 6	0	0	0	1	1
FILA 7	1	1	1	1	1

Numero de filas = 7
Numero de atributos = 4

Numero de Atributos + 1

ATRIBUTOS DISCRETOS



I \ J	A_0	A_1	A_2	A_3	CLASE
FILA 0	0	1	2	3	X
FILA 1	1	0	0	1	1
FILA 2	1	1	0	0	0
FILA 3	0	1	0	1	1
FILA 4	0	0	1	1	1
FILA 5	1	1	1	0	0
FILA 6	0	0	0	1	1
FILA 7	1	1	1	1	1



I \ J	A_1	A_2	A_3	Clase
Fila 0	1	2	3	X
Fila 3	1	0	1	1
Fila 4	0	1	1	1
Fila 6	0	0	1	1

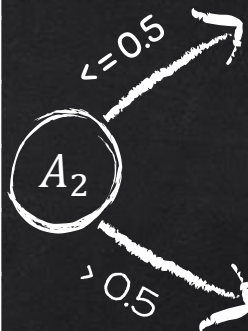
I \ J	A_1	A_2	A_3	Clase
Fila 0	1	2	3	X
Fila 1	0	0	1	1
Fila 2	1	0	0	0
Fila 5	1	1	0	0
Fila 7	1	1	1	1

Se eliminan las filas que no coincidan con el valor del atributo y la columna del atributo. De esta forma, evitamos que el atributo pueda volver a ser utilizado.

ATRIBUTOS CONTINUOS

Umbral = 0.5

I \ J	A ₀	A ₁	A ₂	A ₃	CLASE
FILA 0	0	1	2	3	X
FILA 1	1	0	0.37	1	1
FILA 2	1	1	0.73	0	0
FILA 3	0	1	0.00	1	1
FILA 4	0	0	0.9	1	1
FILA 5	1	1	0.6	0	0
FILA 6	0	0	0.12	1	1
FILA 7	1	1	0.45	1	1



I \ J	A ₀	A ₁	A ₂	A ₃	CLASE
FILA 0	0	1	2	3	X
FILA 1	1	0	0.37	1	1
FILA 3	0	1	0.00	1	1
FILA 6	0	0	0.12	1	1
FILA 7	1	1	0.45	1	1

I \ J	A ₀	A ₁	A ₂	A ₃	CLASE
FILA 0	0	1	2	3	X
FILA 2	1	1	0.73	0	0
FILA 4	0	0	0.9	1	1
FILA 5	1	1	0.6	0	0

Se eliminan las filas que estén por debajo / encima del umbral de el atributo continuo. Por tanto, el atributo puede volver a ser utilizado.



HEURÍSTICA

Entropía

$$E = - \sum_i^{Clases} p_i \log_2(p_i)$$

Ganancia Informacion

$$Gain(A) = E(C) - E(A)$$

$$E(C) = - \sum_{j=1}^{Clases} \frac{n_j}{n} \log_2 \left(\frac{n_j}{n} \right)$$

$$E(A) = \sum_{i=1}^{Part} \frac{n_i}{n} E(nodo_i)$$

Ratio Ganancia

No ha sido utiliza puesto que se ha utilizado un árbol de decisiones binario, es decir, siempre dividimos en dos ramas y por tanto, el ratio de ganancia no aporta información relevante a la hora de tomar decisiones



DISCRETIZACIÓN

Atributo	11	5	0	2	7	15	14	9	4
Clase	0	1	0	0	1	1	1	0	1

Para ordenar sea ha elegido utilizar Quicksort debido a que ofrece ventajas en cuestiones de memoria respecto a Mergesort además de que la complejidad $O(N^2)$ es fácilmente evitable si se escoge el pivote Correctamente. La implementación utilizada ha sido cogida de la pagina Geeksforgeeks(bibliografía) puesto que la implementación es profesional y que el objetivo del trabajo era otro. Dicha implementación ha sido adecuada por nosotros para integrada de forma correcta a nuestro trabajo.



QUICKSORT

Index	0	1	2	3	4	5	6	7	8
Atributo	0	2	4	5	7	9	11	14	15
Clase	0	0	1	1	1	0	0	1	1

$$- E(4) = \frac{2}{9} * \left(\frac{2}{2} * \log_2 \left(\frac{2}{2} \right) + \frac{0}{2} * \log_2 \left(\frac{0}{2} \right) \right) + \frac{7}{9} * \left(\frac{2}{7} * \log_2 \left(\frac{7}{2} \right) + \frac{5}{7} * \log_2 \left(\frac{7}{5} \right) \right) = 0.671$$

$$- E(9) = \frac{5}{9} * \left(\frac{2}{5} * \log_2 \left(\frac{5}{2} \right) + \frac{3}{5} * \log_2 \left(\frac{5}{3} \right) \right) + \frac{4}{9} * \left(\frac{2}{4} * \log_2 \left(\frac{4}{2} \right) + \frac{2}{4} * \log_2 \left(\frac{4}{2} \right) \right) = 0.978$$

$$- E(14) = \frac{7}{9} * \left(\frac{4}{7} * \log_2 \left(\frac{7}{4} \right) + \frac{3}{7} * \log_2 \left(\frac{7}{3} \right) \right) + \frac{2}{9} * \left(\frac{0}{2} * \log_2 \left(\frac{2}{0} \right) + \frac{2}{2} * \log_2 \left(\frac{2}{2} \right) \right) = 0.765$$

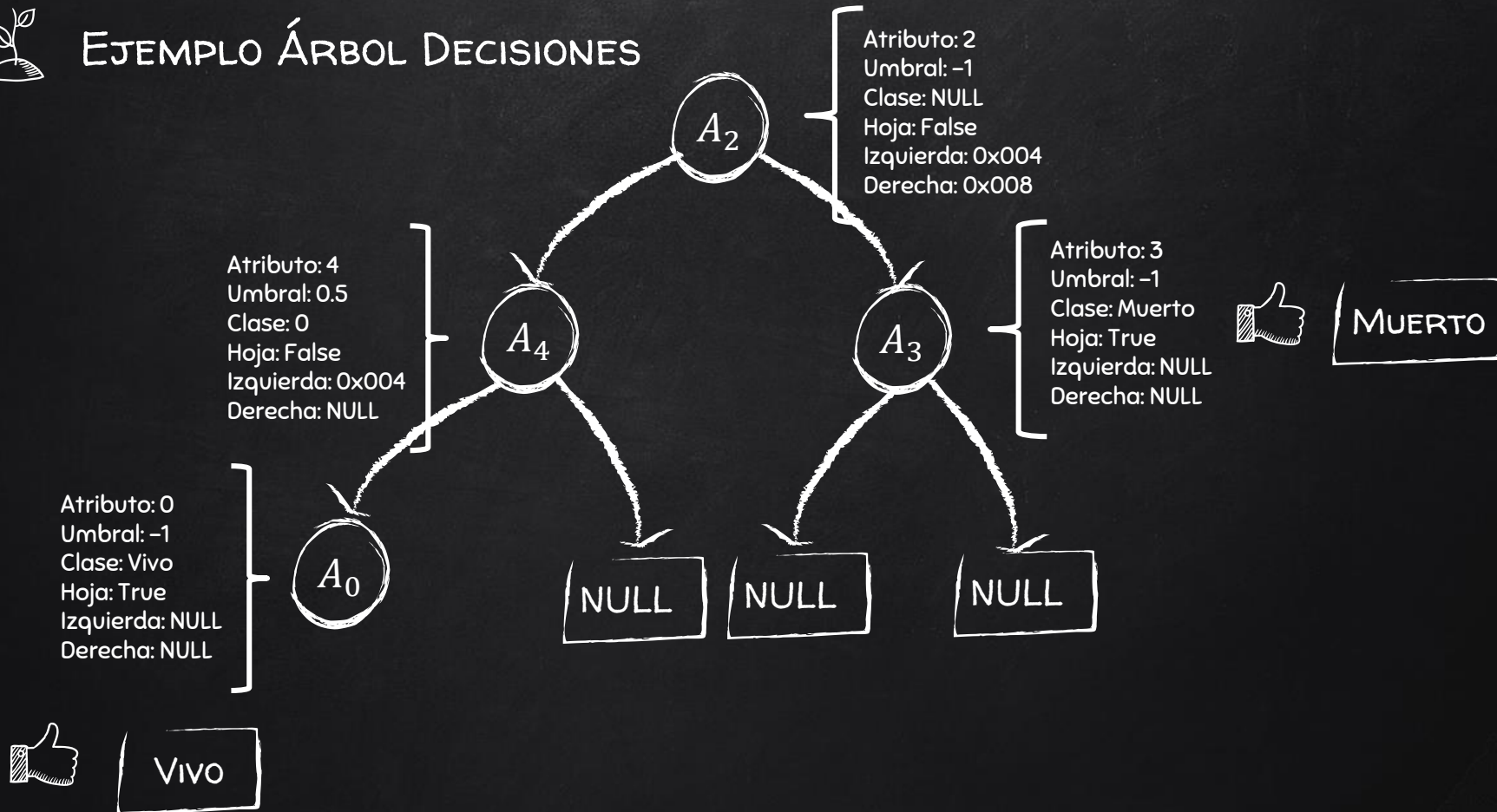
$$\text{MAX} \left[\begin{array}{|c|c|c|c|} \hline \text{Index} & 2 & 5 & 7 \\ \hline \text{Entropía} & E(4) & E(9) & E(14) \\ \hline \end{array} \right] = \begin{array}{|c|} \hline \\ \hline \end{array} E(4) = 0.671$$

Index	0	1	2	3	4	5	6	7	8
Atributo	0	2	4	5	7	9	11	14	15
Clase	0	0	1	1	1	0	0	1	1

$$\text{Umbral}(A_i) = \frac{4+2}{2} = 3$$



EJEMPLO ÁRBOL DECISIONES





FASE TESTEO

X Input: datos de testeo

X Output: la clase a la que pertenece cada muestra de los datos de testeo

Datos de Testeo



Árbol Decisiones



Pertenencia Datos Testeo



Atributo	A_0	A_1	A_2	A_3	Clase Predicha
F_1	1	1	0	0.3	Vivo
F_2	1	0	2	0.7	Muerto

Ejemplo 1:

1 - $F_1(A_2) = 0 \rightarrow$ Izquierda

2 - $F_1(A_3) = 0.3 < \text{Umbral} \rightarrow$ Izquierda

3 - $G_{DT}(A_0) = \text{Es Hoja} \rightarrow F_1(x) = \text{Vivo}$

Ejemplo 2:

1 - $F_2(A_2) = 1 \rightarrow$ Derecha

2 - $G_{DT}(A_3) = \text{Es Hoja} \rightarrow F_2(x) = \text{Vivo}$

Atributo: 4
Umbral: 0.5
Clase: 0
Hoja: False
Izquierda: 0x004
Derecha: NULL

Atributo: 0
Umbral: -1
Clase: Vivo
Hoja: True
Izquierda: NULL
Derecha: NULL

Atributo: 2
Umbral: -1
Clase: NULL
Hoja: False
Izquierda: 0x004
Derecha: 0x008

Atributo: 3
Umbral: -1
Clase: Muerto
Hoja: True
Izquierda: NULL
Derecha: NULL



Vivo



ANÁLISIS RESULTADOS

- X Se analizan los resultados obtenidos
- X Se calcula la tasa de acierto en base al numero de datos predichos correctamente

Pertenencia
Datos Testeo



Análisis
Resultados



Tasa Acierto

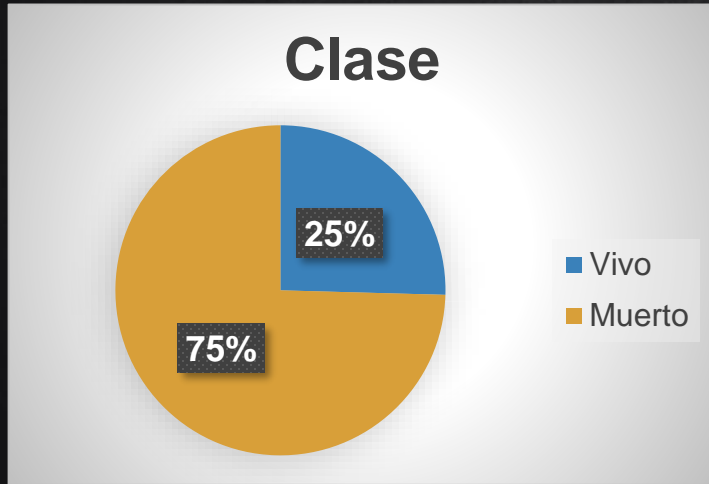




ANÁLISIS RESULTADOS

Atributos Discretos: 0/1

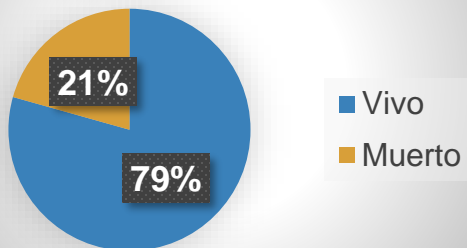
Atributos Continuos: (0, 1)



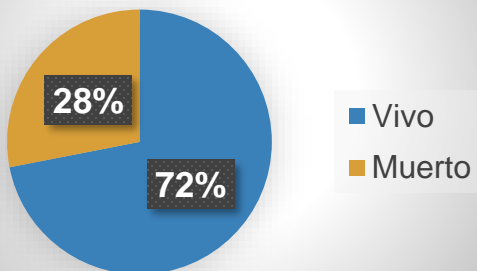
- X Masculino
- X Libro1
- X Libro2
- X Libro3
- X Libro4
- X Libro5
- X Matrimonio
- X Nobleza

- X Muertes Relacionadas
- X Popularidad

Clase Train

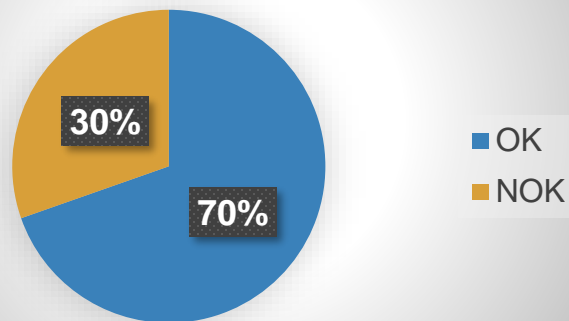


Clase Test



NUMERO ACIERTOS: 660
NUMERO DESACIERTOS: 288

Tasa Acierto



0.696203%



POSIBLES MEJORAS

- X Parametrizar el algoritmo para que trabaje tanto con atributos continuos como con atributos discretos
- X Admitir valores nulos en los datos de entrada, y que no sean utilizados en el calculo de la entropía y ganancia de información
- X Soporte para atributos con diferentes importancias
- X Bagging y boosting



PROBLEMAS

1. Nuestro problema contenía atributos continuos y discretos, por tanto, a la hora de tratar los datos hay que diferenciar entre atributos continuos y discretos
2. Ramas infinitas: para evitarlas se ha establecido un umbral de ganancia de información.
3. Almacenamiento de datos
4. Eliminar filas de la tabla por un atributo



CONCLUSIONES

- X Los resultados no parecen estar del todo mal ($\sim 0,70$)
- X Se ha podido comprobar como los resultados están condicionados por los datos de entrada y de entrenamiento
- X Normalmente cuanto mayor sea la cantidad de datos entrenamiento mejores serán los resultados que se obtendrán
- X La calidad de los datos de entrenamiento también pueden llegar a condicionar los resultados. Es importante que los datos de entrenamiento no presenten sesgos



rabiiix / DecisionTree Private

Unwatch 1 Star 0 Fork 0

Code Issues 0 Pull requests 0 Actions Projects 1 Security Insights Settings

Algoritmo Voraz utilizando arboles de decison. [Manage topics](#) [Edit](#)

49 commits 2 branches 0 packages 1 release

Branch: master New pull request Create new file Upload files Find file Clone or download

rabiiix version totalmente funcional v4 Latest commit e417f4e 2 days ago

.github/workflows	Create cpp.yml	16 days ago
Dataset	lkjds	16 days ago
Otros	comienzo version 2	12 days ago
Test	primera version compila y ejecuta	14 days ago
Versiones	version totalmente funcional v4	2 days ago
.gitignore	mod .gitignore	3 days ago
README.md	Update README.md	2 days ago
Trabajo-1.pdf	leer dataser test	25 days ago
decisionTreeMain.c	programa finalizado	7 days ago
grupos trabajo.pdf	leer dataser test	25 days ago

Se decidió desarrollar el trabajo en GitHub debido a que ofrece un magnifico control de versiones y una buena gestión de desarrollo multiusuario



PROGRESO GITHUB

Arbol de Decsion C

Updated 20 hours ago

Filter cards

+ Add cards

Fullscreen

Me

13 Done

Establecer umbral de heuristica para evitar posibles ramas infinitas

Added by rabiix

Ahora mismo una vez se utiliza un atributo, directamente se elimina de la tabla. Esto hace que la profundidad maxima del arbol sea como mucho igual al numero de atributos. Una posible mejora seria, hacer que si el atributo es continuo se no se elimina hasta que la entropia sea menor que un umbral establecido por nosotros de forma arbitraria pero sensata.

Added by rabiix

a la hora de realizar el la implemntacion del proyecto se puede implementar el algoritmo de la media de la

4 Bugs

CVE-2019-4: si un atributo continuo habia sido eliminado de la tabla, no se comprobaba si habia sido eliminado y se calculaba su umbral. Esto resultaba en una violacion de segmento.

Added by rabiix

CVE-2019-3: Fase de Testeo (testDatosDT()): Cuando se recorren los atributos de las filas a la hora de decidir a que clase pertenece dicha fila, se produce un bucle infinito. Como la mayoria de atributos son discretos se tomo la decision de crear un arbol binario donde los atributos con valor 0 fueran en la rama izquierda y los de valor 1 a la derecha. Cuando se recorren los atributos de las filas existen dos atributos continuos y por tanto, se queda en un bucle infinito debido a

2 Bugs Corregidos

CVE-2019-2: Si la division daba 0, directamente la entropia se pone a 0, si no, se realiza el calculo de forma canonica.

Added by rabiix

CVE-2019-1 Los calculos eran echos mediante divisiones de enteros. Para corregirlo, se ha realizado un casteo en todas aquellas opearaciones que incluian numeros enteros.

Added by rabiix

2 Dificultades

2- Tenemos atributos de dos tipos, lo cual, dificulta la implementacion debido a que dependiendo del atributo hay que tratarlo de una manera o de otra.

Added by rabiix

1- Este dataset tiene aproximadamente 2000 datos y dos de los atributo dos continuos, por tanto, que se creen ramas infinitas es mas que probable.

Added by rabiix

Activar Windows

Ve a Configuración para activar Windows.



BIBLIOGRAFÍA

- X [Código Quicksort](#)
- X [Quicksort vs Mergesort](#)
- X [Greedy Algorithm for Construction of Decision Trees for Tables with Many-Valued Decisions](#)
- X [Gini Index](#)
- X [Decision Tree Introduction GFG](#)
- X [Decision Tree Hackermoon](#)
- X [Decision Tree 1: how it works – YT](#)



GRACIAS!

Preguntas?

Ruben Cherif Narvaez – <https://github.com/rabiixx/>

Jhonny Fabricio Chicaiza Palomo

Daniel Ameztoy Zúñiga



THE QUIETER YOU BECOME, THE MORE YOU ARE ABLE TO HEAR