Ingeniería del Conocimiento

ID3

José María López Pulido

Índice:

Introducción:

Este documento detalla la memoria de la práctica 2 de Ingeniería del Conocimiento, implementación del algoritmo ID3.

El algoritmo ID3 está catalogado como, un algoritmo de inteligencia artificial, fue desarrollado por J. Ross Quinlan. El algoritmo tiene un sistema de aprendizaje, que genera un árbol de decisión mínimo utilizando la técnica “divide y vencerás”.

ID3 recibe una lista de atributos ejemplos, cada atributo tiene un número de opciones posibles, a los que se les asignará un resultado, positivo o negativo.

Ejemplo:

Atributos:	Tiempo exterior		
Opción:	Soleado	Lluvioso	Nublado
Núm veces positivo	6	3	1
Núm veces negativo	3	1	0

En cada iteración del algoritmo se debe calcular el mérito de cada atributo, escogiendo el atributo con menor mérito, que se convertirá en el nodo actual del árbol. El cálculo del mérito de cada atributo sigue la siguiente fórmula:

Entropía:

$$infor(p,n) = p\ log_2(p)\ n\ log_2(n)$$

Siendo p : el tanto por ciento de casos con solución positiva:

$$p:\ \%ejemplos\ + = \frac{|E+|}{|E+|+|E-|}$$

Siendo n: el tanto por ciento de casos con solución negativa:

$$n:\ \%ejemplos\ - = \frac{|E-|}{|E+|+|E-|}$$

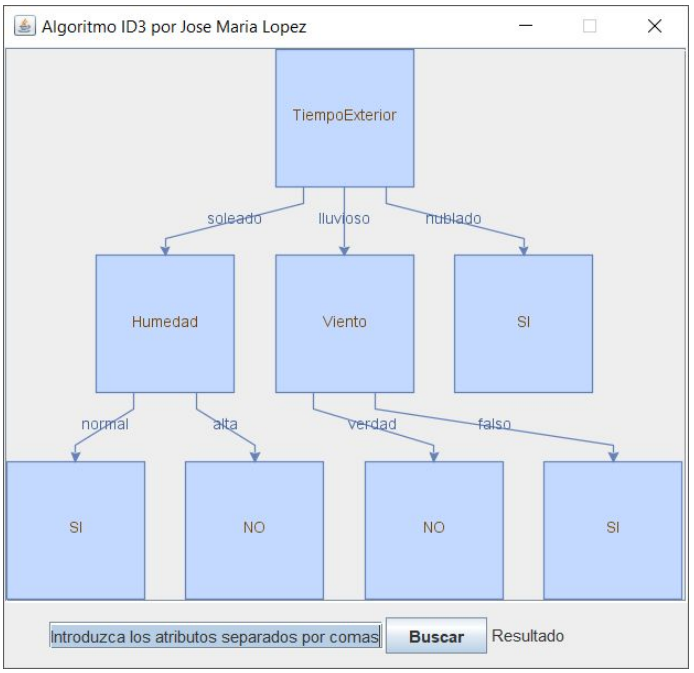
Mérito de un atributo:

$$mérito\ (a_m) = \sum_{i=1}^{n} r_i x\ \ infor(p_i, n_i)$$

Siendo p_i, n_i el número de ejemplos positivos y negativos en el ejemplo actual, y $r_i = a_i / N$ el porcentaje de ejemplos en la rama i, Con N número de ejemplos totales.

Implementación:

El lenguaje utilizado para el desarrollo de la práctica es Java, la interfaz gráfica consta de una sola ventana dividida en dos partes, en la superior podemos encontrar una representación del árbol, y en la segunda una entrada para comprobar un lista de ejemplos nueva.

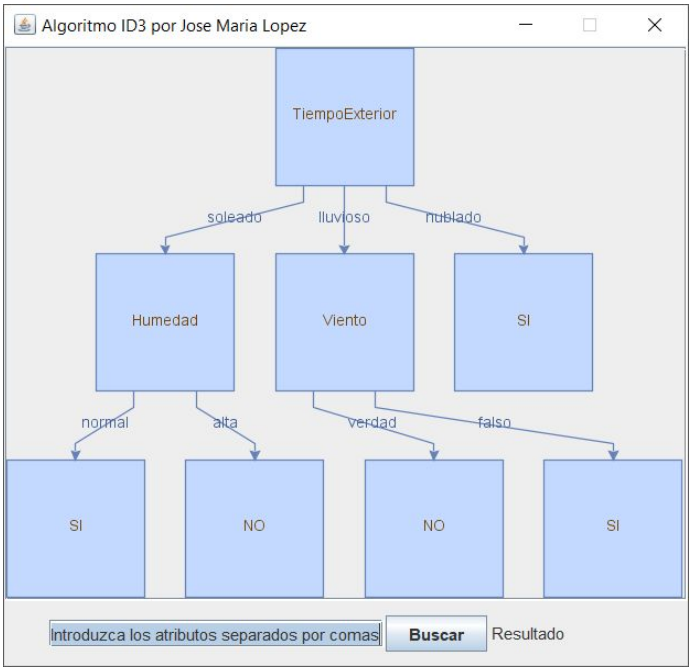


A la hora de la elección de clases para una correcta representación y generación del árbol de decisión, he querido priorizar el manejo, versatilidad y dinamismo en el código, sin que esto supusiera un coste en tiempo de ejecución muy alto. Las clases más importantes en el programa son:

- Clase Atributo con cuatro HashMap, donde la clave para todos es el nombre del ejemplo y usamos cada uno para almacenar diferente información, dos para el conteo del porcentaje de positivos y negativos, otro para el número total de ejemplos y un último con el mérito calculado de cada ejemplo, gracias a esta implementación el añadir nuevos ejemplos a un atributo no supone ningún problema, y al llevar todo calculado, el acceso a datos es más rápido, además atributo también posee un String para el nombre del atributo.
- Clase ID3, que es la clase encargada de leer los dos archivos AtributosJuego.txt, Juego.txt, al leer el primero generamos un ArrayList<Atributo> que almacenará todos los atributos introducidos, de manera que el número de atributos, puede variar. Al leer Juego.txt completaremos la información de cada atributo. A la vez que la información es completada vamos generando una estructura Ejemplos, que contendrá todos los ejemplos leídos del archivo, esta estructura también es dinámica , para facilitar la ampliación, ya sea con más atributos, más ejemplos o con más resultados. Cuando la lectura ha finalizado se procede a la generación del árbol de decisión completo.
- Nodo, esta clase representa un nodo del árbol de decisión, el primer nodo será el nombre del atributo y sus hijos los diferentes ejemplos, para cada nodo hijo, recreamos la estructura de atributo siendo sus nodos hijo, los diferentes ejemplos.
- VistaPrincipal, que implementa la GUI, usando una biblioteca externa jgraphx.jar, para el dibujo del árbol, además tiene un atributo Nodo que contiene el árbol representado, para poder comprobar lo introducido en el buscador.

Funcionamiento de la app:

Al ejecutar la app, mostrará el árbol de decisión completo, en caso de que los archivos no estuvieran en el mismo directorio, mostrará un error.



Funcionamiento del buscador:

El buscador recibe una lista de ejemplos como los del archivo Juego.txt, sin resultado, no importa el orden en el que se introduzca de manera que:

lluvioso,templado,alta,verdad

templado,lluvioso,verdad,alta

serán evaluadas de la misma forma dando el mismo resultado.

Restricciones:

La aplicación está ligada a la estructura de los archivos, cualquier cambio en los archivos no asegura su correcto funcionamiento, dado que la creación de las estructuras está ligada a estos archivos. El primer ejemplo debe de ser un caso falso o negativo, para que los nodos solución Si y No no se visualicen de manera contraria.