En la clase ID3 es donde implentamos el algoritmo como tal.

El método id3 recibe una 2 parámetros, mData que contiene los ejemplos de los atributos y restoAttributos que contiene los atributos sobre los que se va a iterar.

Lo primero que nos encontramos en el método son los casos base, es decir, los casos que al cumplirse terminan esa rama recursiva y “regresa” desde donde se ha realizado la llamada.

Caso base1: La lista restoAttributos está vacía, en este caso “regresamos”.

Caso base2: Todos los ejemplos de mData son del mismo signo, en este caso “regresamos.

En caso de no cumplir ninguno de los casos base, seguimos.

1º Obtenemos el elemento que minimice el mérito, esto lo hacemos llamando al método getBestAttribute. El método gestAttribute queda definido de la siguiente forma:

getBestAttribute(List<String> restoAttributos, List<List<String>> mData)

El método getBestAttribute itera sobre cada uno de los atributos. Por cada atributo itera obteniendo el mérito y comparando quedándose en la variable bestAttribute el atributo con mejor mérito. Una vez ha terminado de iterar sobre todos los atributos, devuelve el mejor atributo obtenido.

2º Obtenemos y guardamos en una lista (noRepeatedValues) los atributos del mejor atributo elegido en el paso anterior.

3º En otra lista, filtramos el resto de atributos de la lista de atributos.

4ºEn un Map<String, List<List<String>> partitionDataMaxValue, guardamos las filas de los ejemplos del mejor atributo elegido en el paso 1. Creamos un Mapa<String, Nodes> (nodes) que contendrá los nodos del árbol.

5º Iteramos sobre los valores de la lista noRepeatedValues. En este bucle hacemos la llamada recursiva que nos devolverá el subárbol (conjunto de nodos), de la rama de este atributo. El nodo devuelto en la llamada recursiva lo añadiremos al mapa Nodes creado en el punto 4.

6º Finalmente regresamos devolviendo el Nodo compuesto del mejor atributo y el el mapa nodes que contiene el resto de nodos elegidos en esa rama.

Siguiendo estos pasos conseguimos la recursividad completa del algoritmo.