PRÁCTICA Nº 4

EL SISTEMA **C4.5** (V.8)

1º) El software correspondiente al C4.5 (realmente el C4.5v8) se encuentra instalado en el ordenador ***lisa.epsig.uniovi.es*** (alternativamente centauro.aulario.uniovi.es). Cada uno trabajará desde su cuenta[1] El algoritmo C4.5 durante su funcionamiento origina ficheros intermedios por lo que será necesario trabajar en un directorio en la que se posea permiso de escritura. Los ficheros de ejemplos que acompañan al software del algoritmo C4.5 se encuentran en el directorio ***/profesores/alquero/examples/*** y deberán copiarse en la cuenta propia.

2º) Antes de la primera ejecución del algoritmo los únicos ficheros relativos a ejemplos que aparecerán tendrán extensiones *.names*¸ *.data* y *.test*. Después de una ejecución con un ***filestem*** (parte principal en el nombre común en todos los ficheros con las extensiones mencionadas y que se refieran a un mismo caso temático: golf, crx, monk1, etc.) aparecerá el fichero denominado *filestem*.***unpruned*** que no transcenderán al usuario y el sistema lo utilizará en posteriores procesos con el mismo *filestem* (generación de reglas, clasificación de ejemplos, etc.).

3º) Es aconsejable estudiar para algunos *filestem* sus correspondientes ficheros con extensiones *data*, que contiene a los ejemplos de entrenamiento, y *names*, que contiene las declaraciones de las clases y de los atributos. Obsérvense detenidamente las estructuras de estos ficheros, en particular las declaraciones de los atributos (para más información consúltese *Un manual breve de C4.5* que se entrega junto con este guión). Se recuerda que los distintos tipos de atributos son:

- Discreto simbólico, su declaración es p.e.: **color_del_pelo: moreno, rubio, pelirrojo.**
- Discreto numérico, su declaración para un caso de 7 valores es p.e.: **día_de_la_semana: discrete 7.**
- Número real, su declaración es p.e.: **peso: continuous.**
- Si se desea ignorar un atributo será suficiente indicarlo en su definición, p.e.: **profesión: ignore.**

4º) Ejecútese el comando **c4.5** para un *filestem* de los que se proponen con ejemplos, g*olf, monk1*, *monk2*, etc., utilizando como único parámetro ***-f.*** El resto de las opciones las tomará el programa por defecto. Si no se redirecciona la salida hacia un fichero, para su impresión o estudio posterior, el resultado saldrá por pantalla directamente y su visión completa y detenida dependerá de las opciones que se tengan en el terminal (se recomienda aumentar el tamaño del *buffer* del terminal hasta 500 o 1000); p.e.:

c4.5 -f golf

(ó, si desea almacenarse el resultado, aunque no es necesario en la clase práctica)

c4.5 -f golf > golf.arb

En este último caso, se sugiere utilizar siempre la misma extensión para las salidas con objeto de facilitar el borrado.

5º) Estúdiese e interprétese el árbol generado con el *filestem* **golf**.

[1] Se necesitará añadir al fichero propio .bash_*profile* (.profile en centauro) el siguiente *path*, si no lo tuviese incorporado ya: ***/usr/local/bin***. Después de este añadido forzar la ejecución del nuevo *.bash_profile (.profile)* **.**

6º) Invóquese el comando ***c4.5rules*** para el *filestem* **golf**, cuyo árbol acabamos de obtener para generar las reglas. Utilizar como único parámetro *-f* y si se cree conveniente redirecciónese nuevamente la salida hacia un fichero; p.e.:

c4.5rules -f golf ***ó*** ***c4.5rules -f golf > golf.req***

Recuérdese siempre que para ejecutar este comando que obtiene las reglas para un *filestem,* se necesita haber construido previamente el árbol para el mismo *filestem* mediante la ejecución del comando c4.5.

7º) Realizar los mismos procesos con algún otro *filestem*, sin añadir y añadiendo diferentes parámetros para comprobar su repercusión en los resultados. Con ciertos parámetros las diferencias serán más notorias dependiendo del mayor o menor número de ejemplos que contenga el fichero de entrenamiento. Algunos parámetros interesantes para manejar son:

-m *número* Influye sobre el grado de expansión del árbol. Pruébese con 1. Por defecto es 2. A menor valor la expansión es mayor. **m** es el número mínimo de ejemplos cubiertos por al menos cada uno de dos de los nodos hijos del nodo en expansión.

-c *número* (*) Influye sobre la poda. Corresponde al nivel de confianza y es un porcentaje, por lo que su valor está entre 0 y 100. Por defecto es 25. A mayor valor menor poda.
(**c** es el nivel de confianza (CF) que se utilizará para hallar el límite superior de un intervalo de confianza que, a su vez, se identifica con la probabilidad de error del nodo/ regla/ atributo cuando se clasifican ejemplos *no vistos*).

-u (*) Se incorporará al comando básico si se desea, a la vez que se generan el árbol o las reglas, ensayarlos sobre un fichero de extensión *.test*. Para utilizarlo elíjase un *filestem* de entre los conjuntos de ejemplos que posean un fichero con la extensión *.test*; cualquiera de los ficheros *monk's* lo poseen.

-g Se añadirá este parámetro si se desea utilizar el viejo criterio de ganancia del ID3 en la elección de atributos al expandir un nodo.

-s Controla la agrupación de valores de un atributo discreto. Por defecto no los agrupa, lo que ocasiona una rama para cada valor del atributo discreto en expansión.

-v *nivel* (*) Determina el nivel de información que el proceso devuelve con el resultado. Por defecto es 0. El máximo nivel es 5, aunque se recomienda no solicitar mayor nivel que 2 por la cantidad de información que genera.

-t *árboles*, **-w** *tamaño***,** y **-i** *incremento* son parámetros que hacen referencia al ***sistema de ventanas*** (véase capítulo 9, pag, 86 de [QUINLAN 93]), sistema heredado del ID3 que se utilizaba cuando el número de ejemplos de entrenamiento era muy elevado. Este *sistema de ventanas* consiste en partir de una *primera ventana* con un subconjunto de los ejemplos de entrenamiento y las reglas que resulten ir depurándolas para sean coherentes con el resto de los ejemplos que se van incorporando gradualmente en *sucesivas ventanas*. Al ser escogidos aleatoriamente los ejemplos de partida de la primera de ventana, las familias de árboles/reglas resultantes pueden ser distintas en ejecuciones sucesivas, motivo principal del interés en este método de funcionamiento del sistema.

Un ejemplo:

c4.5 -f monk1 -c 50 -g -u -m 1

generará un árbol con menor poda de la proporcionada por defecto (*-c 50*) y utilizará el criterio del ID3 en la elección de atributo (*-g*), sesgando la elección hacia atributos con muchos valores frente a los de pocos. Realizará un valoración del árbol de decisión resultante sobre el fichero *monk1.test* (*-u*). El valor uno para *m* (*-m 1*) supondrá que existan en cada expansión al menos dos nodos hijo con un número de casos cubiertos por cada una mayor o igual que uno. El árbol obtenido -antes de la poda- con esta ejecución será el más desarrollado de todos los posibles .

(*) Se utilizan tanto con c4.5 como también con ***c4.5rules***

8º) Edítense ficheros originales con extensiones *names* y *data*, sin que necesariamente sean de gran tamaño. Solamente se trata de afianzarse con la sintaxis que exigen tales ficheros.

9º) Con alguno de los árboles o reglas anteriores utilícense los comandos *consult* y *consultr* respectivamente para clasificar nuevos ejemplos, que se introducirán por teclado uno a uno bajo demanda del sistema de los valores de atributos que los integran. La sintaxis de estos comandos es sencilla:

***consult -f** filestem [-t]* y ***consultr -f** filestem [-t]*.

El parámetro opcional -*t* supone la edición, añadida a la respuesta, del árbol o de las reglas consultadas. En respuesta a los valores de los atributos que demanda el sistema para el nuevo ejemplo que se clasifica, se puede contestar:

? que significa valor desconocido;

un único valor, simbólico o numérico del dominio del los valores del atributo demandado;

un conjunto de posibles valores de atributos discretos bajo la forma: $V_1{:}P_1\ V_2{:}P_2\ \ldots\ V_k{:}P_k$ donde V_i es un valor de los posibles y P_i la correspondiente probabilidad de ocurrencia con la condición de coherencia: $\sum_{i=1}^{k} P_i = 1$; p.e. 3:0.3 5:0.2 4:0.5;

un intervalo bajo la forma ***extremo_inferior-extremo_superior*** para el caso de atributos continuos; p.e. 34-45.

Cada contestación irá seguida de un *retorno.*

BIBLIOGRAFÍA

[MURPHY, 94] MURPHY, P., AHA, D.W.**,** ***UCI Repository of Machine Learning Databases -a machine-readable data repository***. Maintained at the Department of Information and Computer Science, University od California at Irvine. Anonymous ***ftp*** from **ics.uci.edu** in the directory **pub/machine-learning-databases**, (1994).

[QUINLAN, 93] QUINLAN, J.R., *C4.5*: *Programs for Machine Learning. Morgan Kaufmann Publishers*, San Mateo California, 1993.

[THRUN et al., 91]. *The MONK's Problems. A performance Comparison of Different Learning Algorithms.* THRUN, S.B., BALA,J., BLOEDORN, E., BRATKO, I., CESTNIK, B., CHENG, J., DE JONG, J., DZEROSKI, S., FAHLMAN, S.E., FISHER, D., HAMANN, R., KAUFMAN, K., KELLER, S., KONONENKO, I., KREUZIGER, R.S., MICHALSKI, R.S., MITCHELL, T., PACHOWICZ, P., REICH H. VAFAIE, Y., VAN DE WELDE, W., WENZEL, W., WNEK, J., ZHANG, J.. Technical Report CMU-CS-91-197, Carnegie Mellon University, December 1991.

GIJÓN, 11 DE ENERO DE 2010