Departamento de Cs. e Ingeniería de la Computación
Universidad Nacional del Sur

Conceptos de Inteligencia Artificial
&
Sistemas Inteligentes Artificiales

Proyecto: **Aprendizaje Basado en Observaciones**
Árboles de Decisión

Segundo Cuatrimestre de 2020

Objetivo del Proyecto

El objetivo general del presente proyecto consiste en el diseño e implementación en Prolog de un conjunto de predicados que permitan construir un árbol de decisión a partir de un conjunto de ejemplos de entrenamiento dado.

La implementación provista deberá ser genérica, permitiendo la construcción de un árbol de decisión a partir de *cualquier conjunto de entrenamiento* brindado. Es decir, el conjunto de entrenamiento brindado podrá poseer cualquier cantidad de ejemplos de entrenamiento, donde los ejemplos de entrenamiento poseen una cantidad de atributos determinada pero desconocida (incluyendo el atributo de clasificación), y donde cada atributo puede a su vez tomar cualquier cantidad de valores.

Consideraciones Generales

Para la resolución del proyecto podrá asumirse que cada ejemplo de entrenamiento está representado mediante un hecho `ejemplo/3`.

El primer argumento del predicado corresponde a un identificador numérico `ID`. El segundo argumento corresponde a una lista de la forma $[(\mathtt{A_1}, \mathtt{V_{A_1}}), (\mathtt{A_2}, \mathtt{V_{A_2}}), \ldots, (\mathtt{A_n}, \mathtt{V_{A_n}})]$ que modela los atributos y sus valores. Por otra parte, el tercer argumento corresponde a una tupla $(\mathtt{C}, \mathtt{V_C})$ que modela el atributo de clasificación y su valor para el ejemplo en cuestión. Es decir, cada ejemplo del conjunto de entrenamiento estará representado por un hecho:

$$\mathtt{ejemplo}(\mathtt{ID}, [(\mathtt{A_1}, \mathtt{V_{A_1}}), (\mathtt{A_2}, \mathtt{V_{A_2}}), \ldots, (\mathtt{A_n}, \mathtt{V_{A_n}})], (\mathtt{C}, \mathtt{V_C})).$$

Podrá asumirse que el conjunto de ejemplos de entrenamiento brindado es *adecuado*. Recordar que un conjunto de entrenamiento es *adecuado* si no existen dos ejemplos que coinciden en los valores de todos los atributos a excepción del atributo de clasificación.

Asimismo, puede asumirse que no habrá dos ejemplos distintos con el mismo identificador, y que todos los atributos y valores (incluyendo los de clasificación) de los ejemplos se encuentran representados mediante átomos de Prolog.

Por último, para la construcción de los árboles de decisión deberá seguirse el algoritmo de [1], adoptando la estrategia vista en la teoría para la selección del mejor atributo en cada paso (es decir, el atributo que en total clasifique la mayor cantidad de ejemplos). Importante: Si en algún punto no existe un atributo que clasifique al menos un ejemplo para alguno de sus valores, podrá adoptarse cualquier política de selección del mejor atributo.

Ejercicio

Definir un predicado `construirAD/2` que reciba dos argumentos. El *primer argumento* es la ruta completa de un archivo `.pl` que contiene un conjunto de ejemplos de entrenamiento (utilizando la representación descripta anteriormente). El *segundo argumento* es el nombre de un archivo con extensión `.gv` que contendrá el árbol de decisión generado; este archivo deberá ser creado y almacenado en la misma ruta que el archivo recibido como primer argumento. Además, deberá mostrarse un mensaje por pantalla que indique que la creación del árbol fue exitosa, indicando la ruta completa del archivo `.gv` correspondiente.

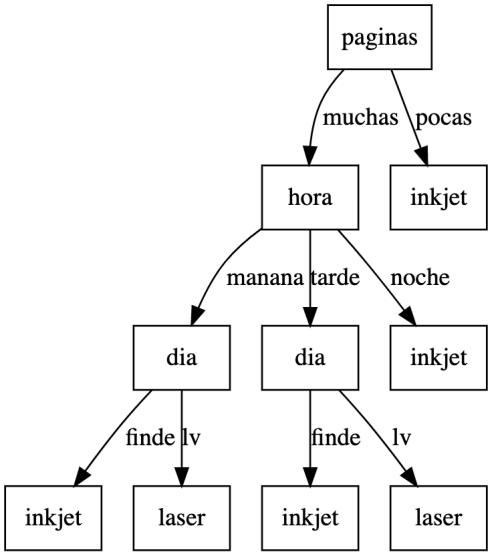
El árbol de decisión creado deberá representarse mediante un *digrafo*, expresado en el lenguaje `DOT`[1].

Ejemplo

Consideremos el ejemplo visto en teoría, que aborda la construcción de un árbol de decisión para un asistente de impresión. Supongamos que el conjunto de ejemplos de entrenamiento para este problema se encuentra representado por el siguiente conjunto de hechos, almacenados en un archivo cuya ruta completa es `'/Downloads/Ejs/ejemplos.pl'`:

```
ejemplo(1, [(dia, lv), (hora, manana), (paginas, pocas), (tipo, pdf), (tamano, grande)], (decision, inkjet)).
ejemplo(2, [(dia, lv), (hora, tarde), (paginas, muchas), (tipo, doc), (tamano, mediano)], (decision, laser)).
ejemplo(3, [(dia, lv), (hora, tarde), (paginas, muchas), (tipo, xls), (tamano, chico)], (decision, laser)).
ejemplo(4, [(dia, lv), (hora, noche), (paginas, muchas), (tipo, doc), (tamano, mediano)], (decision, inkjet)).
ejemplo(5, [(dia, lv), (hora, manana), (paginas, muchas), (tipo, pdf), (tamano, grande)], (decision, laser)).
ejemplo(6, [(dia, lv), (hora, tarde), (paginas, pocas), (tipo, xls), (tamano, chico)], (decision, inkjet)).
ejemplo(7, [(dia, lv), (hora, noche), (paginas, muchas), (tipo, xls), (tamano, chico)], (decision, inkjet)).
ejemplo(8, [(dia, lv), (hora, tarde), (paginas, muchas), (tipo, doc), (tamano, chico)], (decision, laser)).
ejemplo(9, [(dia, lv), (hora, tarde), (paginas, pocas), (tipo, doc), (tamano, chico)], (decision, inkjet)).
ejemplo(10, [(dia, lv), (hora, noche), (paginas, muchas), (tipo, pdf), (tamano, grande)], (decision, inkjet)).
ejemplo(11, [(dia, lv), (hora, manana), (paginas, muchas), (tipo, xls), (tamano, chico)], (decision, laser)).
ejemplo(12, [(dia, finde), (hora, manana), (paginas, muchas), (tipo, pdf), (tamano, grande)], (decision, inkjet)).
ejemplo(13, [(dia, finde), (hora, tarde), (paginas, muchas), (tipo, doc), (tamano, chico)], (decision, inkjet)).
ejemplo(14, [(dia, finde), (hora, manana), (paginas, pocas), (tipo, doc), (tamano, grande)], (decision, inkjet)).
```

Aplicando el algoritmo de construcción de árboles de decisión de [1], y siguiendo la estrategia de selección del mejor atributo vista en la teoría, podemos obtener el siguiente árbol de decisión:



[1] `https://es.wikipedia.org/wiki/DOT`

Teniendo en cuenta esto, podemos realizar la siguiente consulta, obteniendo el resultado indicado a continuación:

```
?- construirAD('/Downloads/Ejs/ejemplos.pl', 'arbol.gv').
Se construyo exitosamente el AD en el archivo: /Downloads/Ejs/arbol.gv
true.
```

donde el contenido del archivo `arbol.gv` puede ser el siguiente:

```
digraph AD {
    node [shape = box]
    paginas1 [label = "paginas"]
    hora1 [label = "hora"]
    dia1 [label = "dia"]
    dia2 [label = "dia"]
    inkjet1 [label = "inkjet"]
    inkjet2 [label = "inkjet"]
    inkjet3 [label = "inkjet"]
    inkjet4 [label = "inkjet"]
    laser1 [label = "laser"]
    laser2 [label = "laser"]
    paginas1 -> inkjet1 [label = "pocas"]
    paginas1 -> hora1 [label = "muchas"]
    hora1 -> dia1 [label = "manana"]
    hora1 -> dia2 [label = "tarde"]
    hora1 -> inkjet2 [label = "noche"]
    dia1 -> laser1 [label = "lv"]
    dia1 -> inkjet3[label = "finde"]
    dia2 -> laser2 [label = "lv"]
    dia2 -> inkjet4[label = "finde"]
}
```

Observación: La definición de nodos y arcos no necesariamente tiene que efectuarse en forma agrupada. El orden de las sentencias que definen nodos o arcos del digrafo es indistinto.

Consideraciones Adicionales para la Implementación

Para la consulta, creación y escritura de archivos en Prolog les recomendamos explorar el manual de referencia de SWI-Prolog. En particular, las secciones 4.3 (Loading Prolog Files), 4.36 (File System Interaction) y 4.17.2 (ISO Input and Output Streams) pueden resultarles útiles.

Si lo desean, podrán adoptar una representación intermedia de los árboles en Prolog, antes de obtener la representación final en el lenguaje `DOT`.

Podrán optar por diferentes alternativas de representación utilizando el lenguaje `DOT`, pero siempre deberán representar los árboles mediante *digrafos*. Las variantes podrán comprender la convención de nombres para la representación de los atributos, las formas y colores adoptadas para los nodos y arcos del árbol, etc.

Por último, si lo desean, pueden hacer uso de herramientas de visualización gráfica de los árboles para validar sus resultados, como por ejemplo `http://www.webgraphviz.com`

Consideraciones de Entrega y Evaluación

1. El proyecto deberá ser resuelto en forma *individual*.
2. La fecha límite para la entrega de la implementación del proyecto es el día **Lunes 23 de Noviembre a la medianoche (23:59 hs)**.
3. La entrega de la implementación del proyecto será a través del aula virtual de Moodle de la materia. Deberá completarse la tarea correspondiente al proyecto, entregando un archivo `.zip` que contenga lo siguiente:
   - Archivos Prolog (`.pl`) correspondientes a la implementación de los predicados requeridos para la resolución del proyecto. En caso de utilizar más de un archivo `.pl`, deberá indicarse cuál es el archivo "principal".
   - Archivos Prolog (`.pl`) correspondientes a los casos de prueba utilizados (conjuntos de entrenamiento), con sus correspondiente archivos `DOT` (`.gv`) de salida. Deberán brindarse al menos 3 casos de prueba adicionales al proporcionado en el enunciado del proyecto. Se ponderará la creatividad en la creación de los casos de prueba.
   - Un archivo `.pdf` en el que se muestren las trazas de los casos de prueba (sin incluir el del enunciado), explicando también los motivos por los que fueron elegidos. Para las trazas deberá utilizarse el formato visto en la teoría, construyendo tablas para la selección de atributos en cada paso, e indicando el árbol parcialmente construido hasta ese momento.
4. La evaluación del proyecto se complementará con la realización y entrega de un video:
   - La fecha límite para la entrega del video es el día **Viernes 27 de Noviembre a la medianoche (23:59 hs)**.
   - La entrega del video será a través del aula virtual de Moodle de la materia. Allí deberá proveerse un link de visualización o descarga del mismo.

   En la siguiente sección se indican algunas pautas para la realización del video.

Guías para la confección del Video

- Deberá explicarse la estrategia adoptada para la resolución del ejercicio, describiendo los principales predicados implementados para tal fin.
- Deberá explicarse toda decisión de diseño adoptada. En particular, deberán describirse detalladamente las convenciones adoptadas para la representación de los árboles en el lenguaje `DOT`, y para la representación intermedia en Prolog (en caso de haberla utilizado).
- Deberá mostrarse el funcionamiento del proyecto para los casos de prueba adoptados.

  Importante: Para el video deberá utilizarse la versión del proyecto entregada.

Referencias

[1] Stuart Russel and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 2nd edition, 2002.