MSc Knowledge Engineering
First Assessed Practical

Decision Trees vs. Ontologies: Final Showdown

February 4, 2005

In this practical, you will be asked to explore the possibilities of using decision tree learning for ontological engineering. Use the `submit` procedure to submit your solution by executing the command

```
 submit msc ke 1 <your-filename>
```

on any DICE machine. The deadline for submission is **Friday 25th February at 4pm**.

Your solution should consist of a single text document containing answers to questions Q1 to Q6 below. The only accepted format for submissions is PDF. Conversion tools to create PDF are available for most word processor and document typesetting software. You will be asked to draw tree representations of ontologies which should be included in this document. A simple and powerful tool for drawing such diagrams that is available on the DICE machines is `xfig` (but you may use any other suitable tool).

Please be brief with textual answers. Apart from lists of rules or diagrams where required, none of the questions should take more than one paragraph to answer.

1 Introduction

So far, we have dealt with decision trees (DTs) as a method for inductive learning. In this assignment, you are going to use DTs for the automated generation of ontologies descriptions of categories that are used just like learning examples in ordinary DT learning. Using a set of typical examples for categories of objects described in terms of attributes, a hierarchical concept tree can be generated without human intervention. You are going to analyse how suitable this approach is, how well it lends it self to human post-processing, how new categories can be integrated into an existing tree, and evaluate the behaviour of the resulting ontology with respect to default reasoning.

2 Building ontologies using decision trees

In this exercise, you are given the following descriptions for different categories of vehicles in terms of a set of attribute values for a typical example of each category labelled with the respective category name:

	A_1: Price	A_2: Environment	A_3: Doors	A_4: Speed	A_5: Passengers	A_6: Name
1	$	GroundLevel	None	+	One	Bicycle
2	$$$$	AboveGround	Few	++++	Many	Plane
3	$$	GroundLevel	Many	+++	Many	HighSpeedTrain
4	$$$	GroundLevel	None	+	Few	CargoShip
5	$$	BelowGround	Many	++	Many	UndergroundTrain
6	$	GroundLevel	Few	++	Few	Car
7	$$$$	AboveGround	Few	++++	Few	Spacecraft
8	$$$$	BelowGround	One	++	Many	Submarine
9	$	GroundLevel	None	++	One	Motorbike
10	$$	AboveGround	Few	+++	Few	Helicopter

Each category is described by attribute values taken from a set V_i for the corresponding attributes A_1 to A_5 that are supposed to be the values of the attributes for a typical member of the category given by the value of A_6. For example, A_3 describes the number of doors the vehicle typically has (where $V_3 = \{None, One, Few, Many\}$), and a car typically has "a few" (but not many) doors.

2.1 Building a taxonomic hierarchy

The first task is to build a taxonomy of vehicle types using the decision-tree learning algorithm discussed in the lectures. Since there are no positive and negative examples, however, and each example is taken to represent an entire class of vehicles, we need to define an appropriate criterion for attribute selection.

The criterion we are going to use is the following: If, at any point in time, the example set you want to split using some attribute is E, choose the *minimal-entropy* attribute

$$A^* = \arg\min_{A_i} \sum_{E_j, |E_j|>0} -\frac{|E_j|}{|E|} \log_2 \frac{|E_j|}{|E|}$$

where $1 \leq i \leq 5$, and E_j is the subset of E that is obtained by selecting all examples in E for which the value of A_i is V_{ij}. Also note that no attribute A_i should be selected twice along the same branch of the tree. Ties between attributes with equally low entropy are broken by preferring attributes with smaller index (e.g. A_1 would be preferred over A_4, A_2 would be preferred over A_3 and A_5 etc.).

Q1: Automated generation of the ontology (25%)
Perform the decision-tree learning algorithm on the above example data using the above attribute selection criterion. Draw the resulting tree by applying while observing the following instructions:

- Attach the label "Vehicle" to the root node
- The child nodes of a node are obtained by performing an attribute test after selecting the minimal entropy attribute and creating new child nodes for all values of that attribute
- Label the resulting nodes with the corresponding attribute-value pair (e.g. "Price=$$")

- When you run out of attributes, create one leaf node for each example still in the example set, using the category name "Name" of the example as a label (e.g. "Name=Car")
- Attribute values for which no examples are available will be leaf nodes in your tree
- Give an account of the attribute selection steps and justify your choices
- List the steps in which you had to break ties between several candidate attributes and note what the result was

Q2: Minimising entropy (10%)
The minimal-entropy criterion prefers attributes that clearly split the example set into as few attribute values as possible while at the same time favouring attributes for which as many examples as possible have the same value.

Discuss what kind of ontologies will usually result when applying this criterion. What are the advantages and disadvantages of this approach? It may be useful to consider extreme kinds of attributes for this purpose, e.g. ones with very many/very few values and ones with a very uniform/biased value distribution among examples.

2.2 Manual post-processing of automatically generated ontologies

Assume you are able to modify the resulting ontology in certain ways as a human knowledge engineer. You may perform the following operations:

- You may rename non-leaf node labels to introduce suitable category names
- If a parent has a single child, you can merge the two nodes (this operation can be extended to entire paths along which no node has more than one child)
- In the cases in which the simple index comparison rule was used to break ties in the above process, you may revise your choice of attribute
- You can combine child nodes of the same parent to generalise among the corresponding attribute values
- You may remove leaf nodes that are not labelled with "Name" values

Q3: Improving the ontology (15%)
Transform the previous tree in accordance with these rules to obtain a "better" ontology. Explain in which way the improved tree is more suitable than the one generated automatically.

Are you finding it hard to give reasonable names to nodes that result from merging a parent node with its immediate descendant? Give reasons.

3 Importing descriptions from other ontologies

In many modern day applications (e.g. the Semantic Web), different people use different ontologies. Assume that you have to extend the ontology developed in Q1 by additional categories given by the following set of descriptions provided by someone else:

	A_1	A_2	A_3	A_4	A_5	A_7: Material	A_6: Name
11	$$	GroundLevel	Few	++	Few		RollsRoyce
12	$$$$		Few	++++	One		FighterJet
13	$$	GroundLevel	Two	++	Few		Truck
14	$$$	GroundLevel	None	+	Few	Rubber	RubberBoat

Note that example 12 has missing attribute information, while example 14 includes an unknown attribute. Also, example 13 introduces a new attribute value.

Q4: Integrating new categories (20%)
Classify examples 11-14 in the tree obtained from Q1 (note that you are not allowed to reprocess examples 1-10 but must work with your existing taxonomy). Which problems can be observed? Suggest methods for dealing with them in an automated way where possible and indicate where manual post-processing would be necessary (and what it should look like).

Draw the modified ontology tree that results from Q1 after inclusion of the new category descriptions and execution of your post-processing procedures.

Q5: Default rules (20%)
Imagine that you do not want to treat the new categories "RollsRoyce", "FighterJet" etc. in the same way as those defined in the improved ontology you created in Q3, but rather that you want to treat these as exceptions to the default attributes of the categories your own ontology describes.

Use default logic to define a set of rules of the format

$$P : J_1, \ldots J_n / C$$

to refine the definition of those classes whose definitions have been affected by the exceptions raised by examples 11 to 14.

As an example, you will need to define a rule that describes that anything that moves at ground level and is quite expensive is a high speed train, unless it has few doors in which case it is a Rolls Royce.

Q6: The impossibility of completeness (%10)
For each of the default rules, find a counterexample of a concrete vehicle that would be classified wrongly despite the use of your ontology and the default rules. (Please do not use vehicle types that do not exist.)

Why can we not use "overriding" as a method for default reasoning in the suggested ontology construction method?