Comp 6721 - Artificial Intelligence - Project 2 project report

Federico O'Reilly Regueiro - 40012304

December 7th, 2016

1

Data: this text
Result: how to write algorithm with LaTeX2e
initialization;
while *not at end of this document* **do**
    read current;
    if *understand* **then**
        go to next section;
        current section becomes this one;
    else
        go back to the beginning of current section;
    end
end

Algorithm 1: How to write algorithms

2 Context free grammars for English

2.a sentences parsed by the given grammar

For the proposed grammar, a noun can be composed in two ways and is included twice in a sentence. Thus, the given grammar could parse/generate $2 \times 2 = 4$ sentences:

- the computer crashes the computer
- the computer crashes the program
- the program crashes the computer
- the program crashes the program

2.b enhance the grammar to parses/generates NPs with modifiers

By modifying rules 1 and 2, the grammar could parse sentences such as *the bad program that crashes the computer*. The necessary modifications are listed below.

i	sentence	$\longrightarrow$ np vp | np compl vp
ii	np	$\longrightarrow$ det noun | det adj noun
iii	vp	$\longrightarrow$ verb np
iv	noun	$\longrightarrow$ computer | program
v	verb	$\longrightarrow$ crashes
vi	det	$\longrightarrow$ the
vii	adj	$\longrightarrow$ fast | bad
viii	compl	$\longrightarrow$ that

The series of parsed/generated sentences grows considerably, since we can now generate sentences in two different ways and nouns in $2 \times 3 = 6$ ways. Since we have two nouns in the sentence then we have $2 \times 2 \times 3 \times 2 \times 3 = 72$ sentences:

the computer crashes the computer
the computer crashes the program
the program crashes the computer
the program crashes the program

the computer that crashes the computer
the computer that crashes the program
the program that crashes the computer
the program that crashes the program

the fast computer crashes the computer
the fast computer crashes the program
the fast program crashes the computer
the fast program crashes the program

the fast computer that crashes the computer
the fast computer that crashes the program
the fast program that crashes the computer
the fast program that crashes the program

the bad computer crashes the computer
the bad computer crashes the program
the bad program crashes the computer
the bad program crashes the program

the bad computer that crashes the computer
the bad computer that crashes the program
the bad program that crashes the computer
the bad program that crashes the program

the computer crashes the fast computer
the computer crashes the fast program
the program crashes the fast computer
the program crashes the fast program

the computer that crashes the fast computer
the computer that crashes the fast program
the program that crashes the fast computer
the program that crashes the fast program

the fast computer crashes the fast computer
the fast computer crashes the fast program
the fast program crashes the fast computer
the fast program crashes the fast program

the fast computer that crashes the fast computer
the fast computer that crashes the fast program
the fast program that crashes the fast computer
the fast program that crashes the fast program

the bad computer crashes the fast computer
the bad computer crashes the fast program
the bad program crashes the fast computer
the bad program crashes the fast program

the bad computer that crashes the fast computer
the bad computer that crashes the fast program
the bad program that crashes the fast computer
the bad program that crashes the fast program

the computer crashes the bad computer
the computer crashes the bad program
the program crashes the bad computer
the program crashes the bad program

the computer that crashes the bad computer
the computer that crashes the bad program
the program that crashes the bad computer
the program that crashes the bad program

the fast computer crashes the bad computer
the fast computer crashes the bad program
the fast program crashes the bad computer
the fast program crashes the bad program

the fast computer that crashes the bad computer
the fast computer that crashes the bad program
the fast program that crashes the bad computer
the fast program that crashes the bad program

the bad computer crashes the bad computer
the bad computer crashes the bad program
the bad program crashes the bad computer
the bad program crashes the bad program

the bad computer that crashes the bad computer
the bad computer that crashes the bad program
the bad program that crashes the bad computer
the bad program that crashes the bad program

TODO now list what should be avoided or what should be added.

3

4 Decission tree

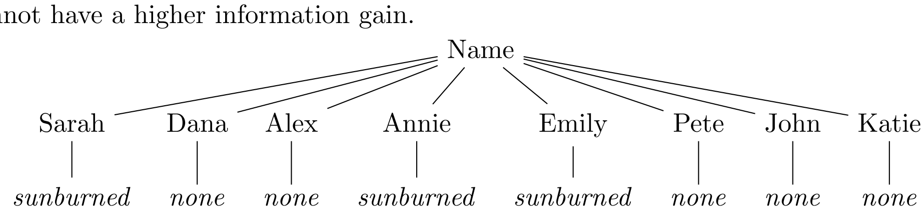
From the table we are given, we can derive the entropy of our observations for the two possible outcomes $sunburnt = \{0, 1\}$.

$$H[sunburnt] = -\frac{3}{8}log_2(\frac{3}{8}) - \frac{5}{8}log_2(\frac{5}{8}) = 0.954434002924965$$

Information gain, $IG(x, y) = H[x] - \sum_y p(y)H[x|y]$ requires calculating conditional entropies given each one of the features. For names, since we have no repeated names, each name is associated with a single outcome, which implies that the entropy of sunburnt *given* a certain name will be 0 for these observations.

$$\begin{aligned} H[sunburnt|Name] &= \sum_n p(sunburnt|Name = n)H[sunburnt|Name = n] \\ &= \sum_n \frac{1}{8} \cdot 0 \\ IG(sunburnt, Name) &= H[sunburnt] - 0 = 0.954434002924965 \end{aligned}$$

Which would make $Name$ an obvious choice for the tree given the sole IG criterion for deciding since we cannot have a higher information gain.



It must be noted, however, that yielding one leaf per observation is generally due to a poor choice of feature leading to overfitting, and representative of the high variance typical of decision trees. This decision tree does not generalize well.

5 Genetic Algorithms

5.a defining a gene representation

Use a string of 5 hexadecimal digits, a sign and an exponent. Placing the exponent on one side and the sign on the other would give these two elements some positional independence

5.b fitness function

5.c crossover and mutation - 2 generations for a small initial population of 3

5.d explain the state space - convergence?

5.e how might GA's solve this? Preferable to brute force search?