Einführung in die Computerlinguistik
Decision Trees

Hinrich Schütze & Robert Zangenfeind

Centrum für Informations- und Sprachverarbeitung, LMU München

2015-11-23

This lecture is based on
Russell and Norvig's introduction
to artificial intelligence.

Take-away today

Take-away today

- Introduction to decision trees

Take-away today

- Introduction to decision trees
- (Almost) fully understand one complex machine learning model

Take-away today

- Introduction to decision trees
- (Almost) fully understand one complex machine learning model
- Basis for hands-on training and applying this model
  in next practical exercise

Take-away today

- Introduction to decision trees
- (Almost) fully understand one complex machine learning model
- Basis for hands-on training and applying this model
  in next practical exercise
- Exam: questions testing basic understanding of decision trees,
  but no formulas

Overview

Outline

Attributes

Attributes

As an example, we will build a decision tree to decide whether to wait for a table at a restaurant. The aim here is to learn a definition for the **goal predicate** *WillWait*. First we list the attributes that we will consider as part of the input:

1. *Alternate*: whether there is a suitable alternative restaurant nearby.
2. *Bar*: whether the restaurant has a comfortable bar area to wait in.
3. *Fri/Sat*: true on Fridays and Saturdays.
4. *Hungry*: whether we are hungry.
5. *Patrons*: how many people are in the restaurant (values are *None*, *Some*, and *Full*).
6. *Price*: the restaurant's price range ($, $$, $$$).
7. *Raining*: whether it is raining outside.
8. *Reservation*: whether we made a reservation.
9. *Type*: the kind of restaurant (French, Italian, Thai, or burger).
10. *WaitEstimate*: the wait estimated by the host (0–10 minutes, 10–30, 30–60, or >60).

Decision tree for deciding whether to wait

Decision tree for deciding whether to wait

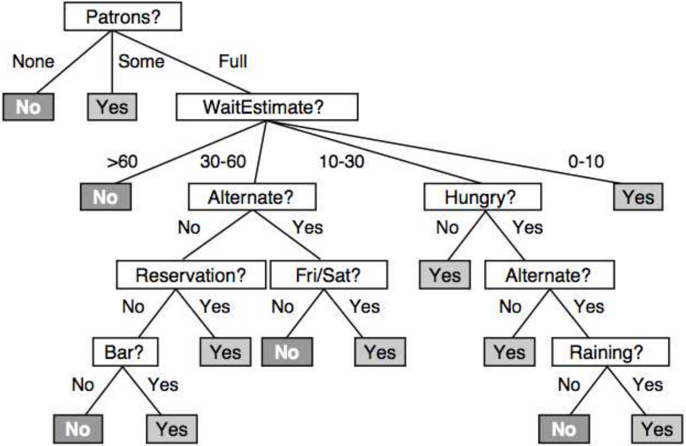


Figure 18.2 A decision tree for deciding whether to wait for a table.

(explain one path)

Expressiveness of decision trees

Expressiveness of decision trees

18.3.2 Expressiveness of decision trees

A Boolean decision tree is equivalent to a logical expression of the form

$$Goal \Leftrightarrow (Path_1 \vee Path_2 \vee \cdots) ,$$

where each $Path_i$ has the form

$$Path = (A_i = a_i \wedge A_j = a_j \wedge \cdots) ,$$

that is, the goal is true if and only if there is a path through the tree that ends in a positive result. Since this is equivalent to disjunctive normal form, that means that any function in propositional logic can be expressed as a decision tree. As an example, the rightmost path in Figure 18.2 is

$$Path = (Patrons = Full \wedge WaitEstimate = 0\text{–}10) .$$

Training set

Training set

Example	Input Attributes										Goal
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*WillWait*
$\mathbf{x}_1$	*Yes*	*No*	*No*	*Yes*	*Some*	*$$$*	*No*	*Yes*	*French*	*0–10*	$y_1 = Yes$
$\mathbf{x}_2$	*Yes*	*No*	*No*	*Yes*	*Full*	*$*	*No*	*No*	*Thai*	*30–60*	$y_2 = No$
$\mathbf{x}_3$	*No*	*Yes*	*No*	*No*	*Some*	*$*	*No*	*No*	*Burger*	*0–10*	$y_3 = Yes$
$\mathbf{x}_4$	*Yes*	*No*	*Yes*	*Yes*	*Full*	*$*	*Yes*	*No*	*Thai*	*10–30*	$y_4 = Yes$
$\mathbf{x}_5$	*Yes*	*No*	*Yes*	*No*	*Full*	*$$$*	*No*	*Yes*	*French*	*>60*	$y_5 = No$
$\mathbf{x}_6$	*No*	*Yes*	*No*	*Yes*	*Some*	*$$*	*Yes*	*Yes*	*Italian*	*0–10*	$y_6 = Yes$
$\mathbf{x}_7$	*No*	*Yes*	*No*	*No*	*None*	*$*	*Yes*	*No*	*Burger*	*0–10*	$y_7 = No$
$\mathbf{x}_8$	*No*	*No*	*No*	*Yes*	*Some*	*$$*	*Yes*	*Yes*	*Thai*	*0–10*	$y_8 = Yes$
$\mathbf{x}_9$	*No*	*Yes*	*Yes*	*No*	*Full*	*$*	*Yes*	*No*	*Burger*	*>60*	$y_9 = No$
$\mathbf{x}_{10}$	*Yes*	*Yes*	*Yes*	*Yes*	*Full*	*$$$*	*No*	*Yes*	*Italian*	*10–30*	$y_{10} = No$
$\mathbf{x}_{11}$	*No*	*No*	*No*	*No*	*None*	*$*	*No*	*No*	*Thai*	*0–10*	$y_{11} = No$
$\mathbf{x}_{12}$	*Yes*	*Yes*	*Yes*	*Yes*	*Full*	*$*	*No*	*No*	*Burger*	*30–60*	$y_{12} = Yes$

Usefulness of attributes

Usefulness of attributes

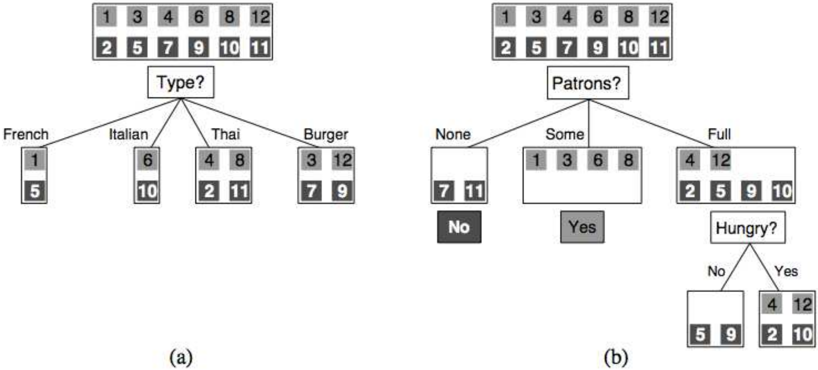


Figure 18.4 Splitting the examples by testing on attributes. At each node we show the positive (light boxes) and negative (dark boxes) examples remaining. (a) Splitting on *Type* brings us no nearer to distinguishing between positive and negative examples. (b) Splitting on *Patrons* does a good job of separating positive and negative examples. After splitting on *Patrons*, *Hungry* is a fairly good second test.

Decision tree learning

Decision tree learning

function DECISION-TREE-LEARNING(*examples*, *attributes*, *parent_examples*) **returns** a tree

  if *examples* is empty **then return** PLURALITY-VALUE(*parent_examples*)
  else if all *examples* have the same classification **then return** the classification
  else if *attributes* is empty **then return** PLURALITY-VALUE(*examples*)
  else
    $attr \leftarrow \operatorname{argmax}_{a \in attributes}$ IMPORTANCE(a, *examples*)
    tree ← a new decision tree with root test *attr*
    for each value v_i of *attr* **do**
      $exs \leftarrow \{e : e \in examples$ **and** $e.attr = v_i\}$
      subtree ← DECISION-TREE-LEARNING(*exs*, *attributes* − *attr*, *examples*)
      add a branch to *tree* with label ($attr = v_i$) and subtree *subtree*
    return *tree*

Figure 18.5 The decision-tree learning algorithm. The function IMPORTANCE is described in Section 18.3.4. The function PLURALITY-VALUE selects the most common output value among a set of examples, breaking ties randomly.

Induced decision tree

Induced decision tree

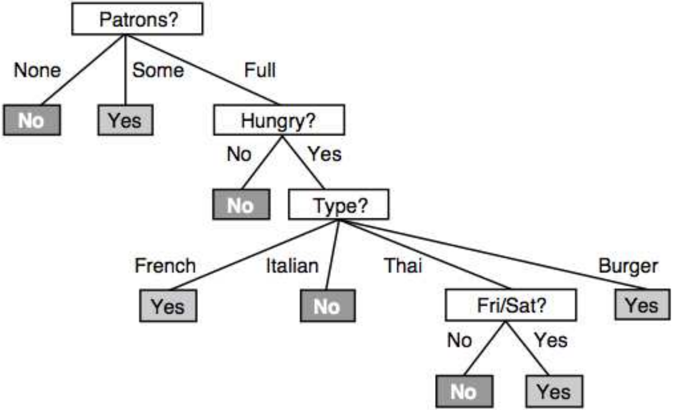


Figure 18.6 The decision tree induced from the 12-example training set.

Hand-designed vs. induced trees

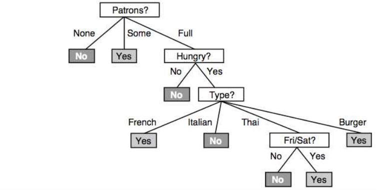


Figure 18.6 The decision tree induced from the 12-example training set.

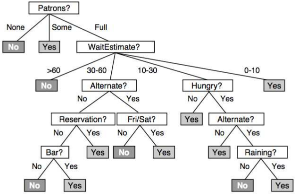


Figure 18.2 A decision tree for deciding whether to wait for a table.

Decision tree learning: Importance?

function DECISION-TREE-LEARNING(*examples*, *attributes*, *parent_examples*) **returns** a tree

if *examples* is empty **then return** PLURALITY-VALUE(*parent_examples*)
else if all *examples* have the same classification **then return** the classification
else if *attributes* is empty **then return** PLURALITY-VALUE(*examples*)
else
  $attr \leftarrow \text{argmax}_{a \in attributes}$ IMPORTANCE(a, *examples*)
  tree ← a new decision tree with root test *attr*
  for each value v_i of *attr* **do**
    $exs \leftarrow \{e : e \in examples$ **and** $e.attr = v_i\}$
    subtree ← DECISION-TREE-LEARNING(*exs*, *attributes* − *attr*, *examples*)
    add a branch to *tree* with label ($attr = v_i$) and subtree *subtree*
  return *tree*

Figure 18.5 The decision-tree learning algorithm. The function IMPORTANCE is described in Section 18.3.4. The function PLURALITY-VALUE selects the most common output value among a set of examples, breaking ties randomly.

Entropy

Entropy

$$
\begin{aligned}
H(V) &= \sum_i P(v_i) \log_2 \frac{1}{P(v_i)} \\
&= -\sum_i P(v_i) \log_2 P(v_i)
\end{aligned}
$$

Entropy: Restaurant example

$$
\begin{aligned}
H(V) &= \sum_i P(v_i) \log_2 \frac{1}{P(v_i)} \\
&= -\sum_i P(v_i) \log_2 P(v_i)
\end{aligned}
$$

$$
H(\text{Goal}) = -\left(\frac{p}{p+n} \log_2 \frac{p}{p+n} + \frac{n}{p+n} \log_2 \frac{n}{p+n}\right)
$$

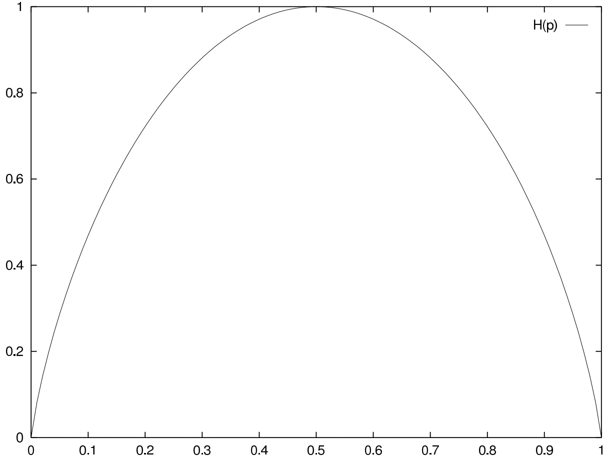
p is number of positive examples ("will wait"),
n is number of negative examples ("will not wait")

Entropy: Restaurant example

$$
\begin{aligned}
H(\text{Goal}) &= -\left(\frac{p}{p+n}\log_2\frac{p}{p+n} + \frac{n}{p+n}\log_2\frac{n}{p+n}\right) \\
&= -\left(\frac{6}{6+6}\log_2\frac{6}{6+6} + \frac{6}{6+6}\log_2\frac{6}{6+6}\right) \\
&= -2\frac{6}{6+6}\log_2\frac{6}{6+6} \\
&= \log_2 2 \\
&= 1
\end{aligned}
$$

Plot of entropy

Plot of entropy



Notation

$$H(T(q)) = -q \log_2 q - (1-q) \log_2 (1-q)$$

Remainder = Remaining uncertainty

$$\text{Remainder}(A) = \sum_{i=1}^{|v(A)|} \frac{p_i + n_i}{p + n} H(T(\frac{p_i}{p_i + n_i}))$$

p_i is the number of positive examples that have attribute value v_i ($A = v_i$) and n_i is the number of negative examples that have attribute value v_i ($A = v_i$).

Information gain

Information gain

$$\text{Gain}(A) = H(T(\frac{p}{p+n})) - \text{Remainder}(A)$$

Information gain for Pt = Patrons and Tp = Type

$$
\begin{aligned}
\text{Gain(Tp)} &= 1 - [\frac{2}{12}H(T(\frac{1}{2})) + \frac{2}{12}H(T(\frac{1}{2})) + \frac{4}{12}H(T(\frac{2}{4})) + \frac{4}{12}H(T(\frac{2}{4}))] \\
&= 1 - [\frac{2}{12} + \frac{2}{12} + \frac{4}{12} + \frac{4}{12}] \\
&= 0 \text{ bits} \\
\text{Gain(Pt)} &= 1 - [\frac{2}{12}H(T(\frac{0}{2})) + \frac{4}{12}H(T(\frac{4}{4})) + \frac{6}{12}H(T(\frac{2}{6}))] \\
&= 1 - [0 + 0 + \frac{1}{2}H(T(\frac{1}{3}))] \\
&\approx 0.541 \text{ bits}
\end{aligned}
$$

Entropy exercise (goal)

example	decision	type	day of week	colleague?
x_1	yes	french	saturday	"let's stay"
x_2	no	thai	friday	"let's go"
x_3	yes	burger	saturday	"let's stay"
x_4	yes	thai	sunday	"let's stay"
x_5	no	french	friday	"let's go"
x_6	yes	italian	sunday	"let's stay"
x_7	no	burger	friday	"let's go"
x_8	yes	thai	sunday	"let's stay"
x_9	no	burger	friday	"not sure"
x_{10}	no	italian	friday	"not sure"
x_{11}	no	thai	friday	"not sure"
x_{12}	yes	burger	sunday	"not sure"

Decision tree learning: Importance?

function DECISION-TREE-LEARNING(*examples*, *attributes*, *parent_examples*) **returns** a tree

```
if examples is empty then return PLURALITY-VALUE(parent_examples)
else if all examples have the same classification then return the classification
else if attributes is empty then return PLURALITY-VALUE(examples)
else
    attr ← argmax_{a ∈ attributes} IMPORTANCE(a, examples)
    tree ← a new decision tree with root test attr
    for each value v_i of attr do
        exs ← {e : e ∈ examples and e.attr = v_i}
        subtree ← DECISION-TREE-LEARNING(exs, attributes − attr, examples)
        add a branch to tree with label (attr = v_i) and subtree subtree
    return tree
```

Figure 18.5 The decision-tree learning algorithm. The function IMPORTANCE is described in Section 18.3.4. The function PLURALITY-VALUE selects the most common output value among a set of examples, breaking ties randomly.

Outline

NLTK decision tree demo

Take-away today

- Introduction to decision trees
- (Almost) fully understand one complex machine learning model
- Basis for hands-on training and applying this model in next practical exercise
- Exam: questions testing basic understanding of decision trees, but no formulas