Uczenie maszynowe

Jest konieczne w nieznanym środowisku, tj. gdy projektant systemu nie posiada wszechwiedzy

Jest przydatne jako metoda konstrukcji systemu, tj. konfrontuje agenta z rzeczywistością zamiast próby napisania jego programu

Modyfikuje mechanizm decyzyjny agenta, aby poprawić wydajność

Wszystkie typy uczenia mogą byc rozważane jako uczenie reprezentacji funkcji.

Uczenie indukcyjne

Element uczący otrzymuje poprawną wartość funkcji dla poszczególnych danych wejściowych i na tej podstawie modyfikuje reprezentację funkcji tak, aby pasowała do dostarczonej informacji.

Formalnie:
Obiekty: dane opisujące stan lub obiekt; tworzą przestrzeń obiektów X.

Decyzja: Funkcja $dec : X \rightarrow V_{dec}$ przypisująca obiektom ze zbioru X wartość decyzji z ustalonego zbioru V_{dec}.

Zbiór przykładów: Ustalony zbiór obiektów z X z przypisanymi wartościami decyzji: tj. zbiór par $(x_i, dec(x_i))$ dla $i = 1, \ldots, n$.

Zadanie: Z danego zbioru przykładów nauczyć się funkcji (hipotezy) $h : X \to V_{dec}$ aproksymującej decyzję dec tak, aby możliwie *najbardziej poprawnie* przypisywała ją obiektom z przestrzeni X, dla których nieznana jest wartość decyzji dec.

Przykład – kółko i krzyżyk

Uczenie funkcji z przykładów (*tabula rasa)*

f *jest* funkcją docelową

Przykład to para $(x, f(x))$*, tj.,* $\begin{array}{|c|c|c|}\hline O & O & X \\ \hline & X & \\ \hline X & & \\ \hline\end{array}$, $+1$

Problem: znaleźć hipotezę h
taką że $h \approx f$
dla danego zbioru przykładów

Hipoteza h *jest* spójna *na zbiorze*
$\{(x_1, dec(x_1)), \ldots, (x_n, dec(x_n))\}$ *jeśli*

$$h(x_i) = dec(x_i) \text{ dla każdego } 1 \leq i \leq n$$

Rodzaje decyzji

Decyzja może przyjmować wartości:

- rzeczywiste
- dyskretne
- binarne (TRUE, fALSE)

Przestrzeń hipotez

W ogólności może być wiele hipotez dla tego samego zbioru przykładów.

Ile jest różnych hipotez (funkcji) binarnych dla n atrybutów binarnych?

$=$ liczba funkcji binarnych dla dziedziny z 2^n obiektami $= 2^{2^n}$

Przestrzeń hipotez można ograniczyć do ustalonej klasy hipotez.

Z drugiej strony zwiększanie przestrzeni hipotez:

- zwiększa szansę, że funkcja docelowa może być wyrażona
- zwiększa liczbę hipotez zgodnych ze zbiorem przykładów

Empiryczna miara jakości hipotezy

Sposób postępowania:

- Dane X dzielimy na dwa podzbiory – *zbiór treningowy* Z_{trn} i *zbiór testowy* Z_{tst}.

- Hipoteza $h : X \rightarrow V_{dec}$ jest indukowana na podstawie zbioru treningowego Z_{trn}.

- *Skuteczność* hipotezy $Acc(h)$ jest mierzona proporcją poprawnie sklasyfikowanych obiektów ze zbioru testowego do rozmiaru zbioru testowego, tj.

$$Acc(h) = \frac{\mid \{x \in Z_{tst} : h(x) = dec(x)\} \mid}{\mid Z_{tst} \mid}$$

Stosowane podejścia

Zależą od reprezentacji modułu decyzyjnego agenta:

1. drzewa decyzyjne
2. wnioskowanie oparte na podobieństwie - k najbliższych sąsiadów
3. sieci neuronowe
4. sieci bayessowskie
5. systemy regułowe

Drzewa decyzyjne

Wejście: obiekt lub stan opisany przez zbiór własności (atrybutów).

Wyjście: decyzja tak/nie (wartości wyjściowych może być więcej).

Opisują funkcje boolowskie.

Reprezentacja:

Węzły wewnętrzne: Każdy związany z jednym atrybutem - reprezentuje test wartości tego atrybutu.

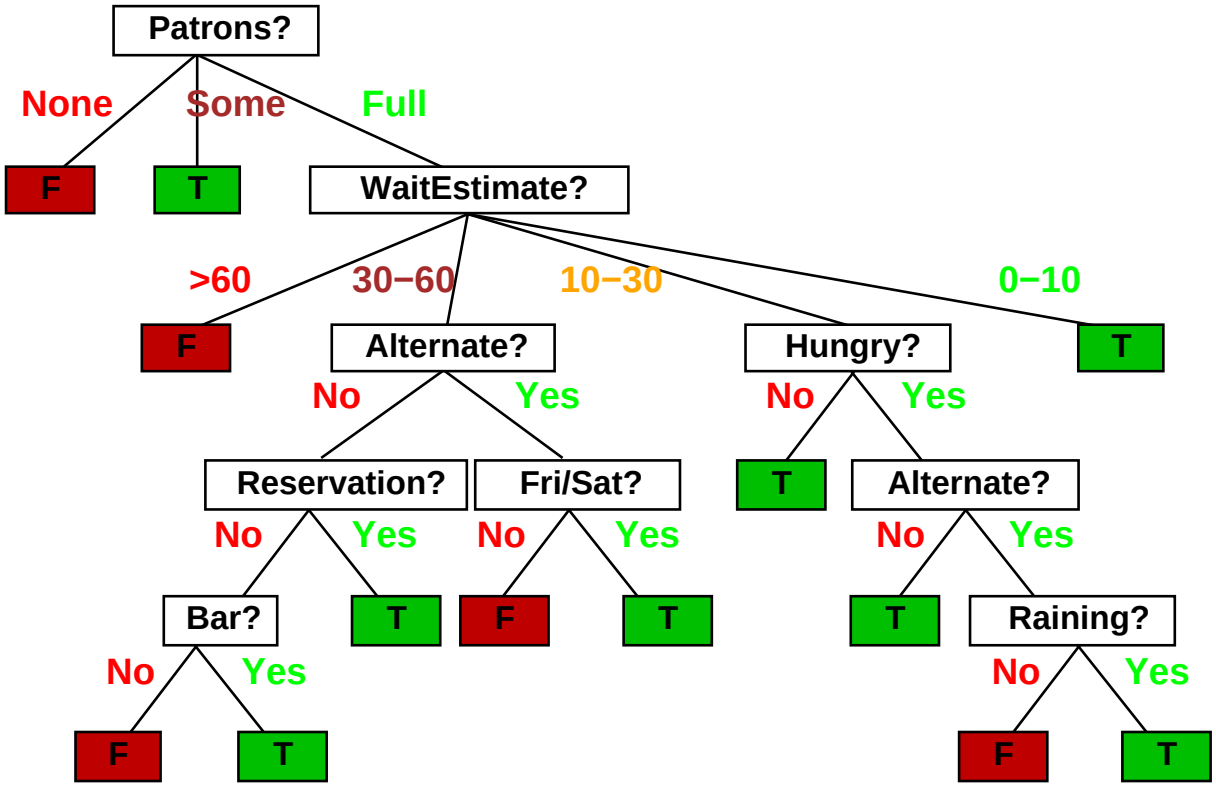
Gałęzie: Każda związana z wartością lub podzbiorem wartości atrybutu węzła, z którego wychodzi - odpowiada obiektom danych z pasującymi wartościami atrybutu.

Liście: Każdy związany z decyzją lub rozkładem decyzji - odpowiada obiektom danych pasującym do ścieżki prowadzącej do danego liścia.

Przykład

Example	Attributes										Target
	Alt	Bar	Fri	Hun	Pat	$Price$	$Rain$	Res	$Type$	Est	*WillWait*
X_1	*T*	*F*	*F*	*T*	*Some*	*$$$*	*F*	*T*	*French*	*0–10*	*T*
X_2	*T*	*F*	*F*	*T*	*Full*	*$*	*F*	*F*	*Thai*	*30–60*	*F*
X_3	*F*	*T*	*F*	*F*	*Some*	*$*	*F*	*F*	*Burger*	*0–10*	*T*
X_4	*T*	*F*	*T*	*T*	*Full*	*$*	*F*	*F*	*French*	*10–30*	*T*
X_5	*T*	*F*	*T*	*F*	*Full*	*$$$*	*F*	*T*	*Thai*	*>60*	*F*
X_6	*F*	*T*	*F*	*T*	*Some*	*$$*	*T*	*T*	*Italian*	*0–10*	*T*
X_7	*F*	*T*	*F*	*F*	*None*	*$*	*T*	*F*	*Burger*	*0–10*	*F*
X_8	*F*	*F*	*F*	*T*	*Some*	*$$*	*T*	*T*	*Thai*	*0–10*	*T*
X_9	*F*	*T*	*T*	*F*	*Full*	*$*	*T*	*F*	*Burger*	*>60*	*F*
X_{10}	*T*	*T*	*T*	*T*	*Full*	*$$$*	*F*	*T*	*Italian*	*10–30*	*F*
X_{11}	*F*	*F*	*F*	*F*	*None*	*$*	*F*	*F*	*Thai*	*0–10*	*F*
X_{12}	*T*	*T*	*T*	*T*	*Full*	*$*	*F*	*F*	*Burger*	*30–60*	*T*

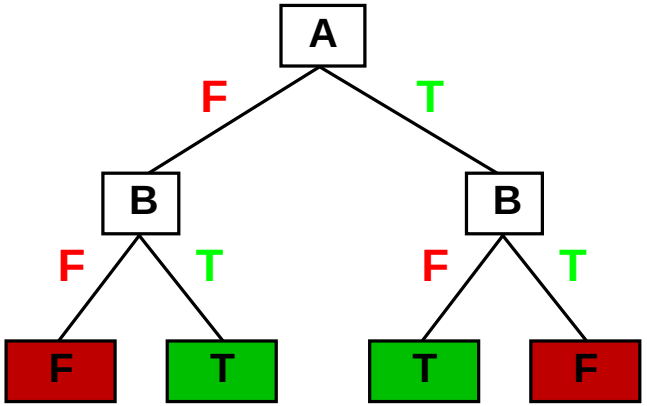
Drzewo decyzyjne



Drzewa decyzyjne– moc wyrażania

Dla każdego zbioru przykładów istnieje spójne drzewo decyzyjne – jedna ścieżka do liścia dla każdego przykładu (chyba, że f jest niedeterministyczna).

A	B	A xor B
F	F	F
F	T	T
T	F	T
T	T	F



Zadanie:
Szukanie bardziej zwartego drzewa decyzyjnego.

Uczenie drzewa decyzyjnego

Cel: znalezienie małego drzewa zgodnego ze zbiorem treningowym.

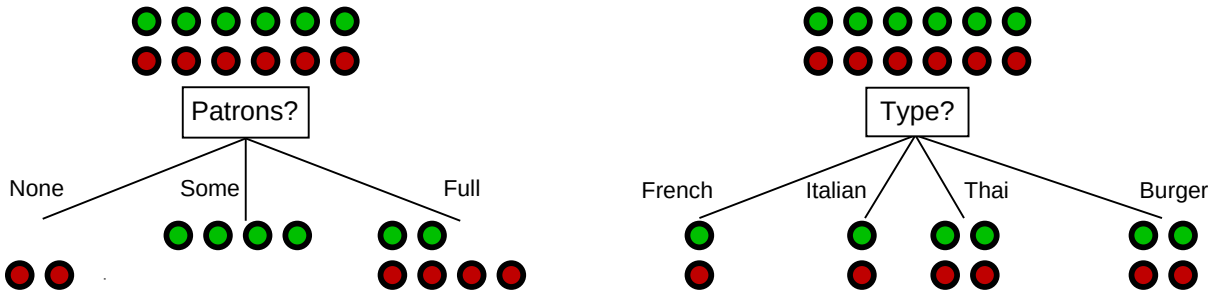
Pomysł: rekurencyjne wybieranie najbardziej znaczącego atrybutu jako korzenia poddrzewa.

```
function DECISION-TREE-LEARNING(examples, attributes, default) returns a decision tree
  inputs: examples, set of examples
          attributes, set of attributes
          default, default value for the goal predicate

  if examples is empty then return default
  else if all examples have the same classification then return the classification
  else if attributes is empty then return MAJORITY-VALUE(examples)
  else
      best ← CHOOSE-ATTRIBUTE(attributes, examples)
      tree ← a new decision tree with root test best
      for each value v_i of best do
          examples_i ← {elements of examples with best = v_i}
          subtree ← DECISION-TREE-LEARNING(examples_i, attributes − best,
                                           MAJORITY-VALUE(examples))
          add a branch to tree with label v_i and subtree subtree
      end
      return tree
```

Wybór atrybutu

Idea: dobry atrybut rozdziela przykłady na podzbiory, których elementy są wszystkie "pozytywne" lub wszystkie "negatywne".



$Patrons?$ jest lepszym wyborem – daje **informację** o klasyfikacji.

Informacja

Rozwiązaniem jest informacja.

Im większa ignorancja w kwestii odpowiedzi na początku, tym więcej informacji zawiera odpowiedź.

Skala: 1bit = odpowiedź na pytanie boolowskie z prawdopodobieństwem $\langle 0.5, 0.5 \rangle$

Entropia

Dany jest rozkład prawdopodobieństwa

$$\langle P_1, \ldots, P_n \rangle.$$

Miara informacji (*entropia prawdopodobieństwa*) wyznacza ile informacji niesie ten rozkład

$$H(\langle P_1, \ldots, P_n \rangle) = \Sigma_{i=1}^{n} - P_i \log_2 P_i$$

S –zbiór danych
S_d –zbiór obiektów z decyzją d

$$H(S) = \Sigma_{d \in V_{dec}} - \frac{\mid S_d \mid}{\mid S \mid} log_2 \frac{\mid S_d \mid}{\mid S \mid}$$

Entropia = średnia liczba bitów potrzebna do zakodowania decyzji d dla losowo wybranego obiektu ze zbioru S. Optymalne kodowanie przydziela $-log_2 p$ bitów do decyzji występującej z prawdopodobieństwem p.

Entropia: przypadek 2 decyzji

Dane są dwie decyzje: pozytywna (+) i negatywna (-)

$p_+ = \frac{|S_+|}{|S|}$ – częstość obiektów z decyzją pozytywną w S

$p_- = \frac{|S_-|}{|S|}$ – częstość obiektów z decyzją negatywną w S

$$H(S) = -p_+ log_2 p_+ - p_- log_2 p_-$$

Zysk informacji

Zysk informacji $G(S, a)$=

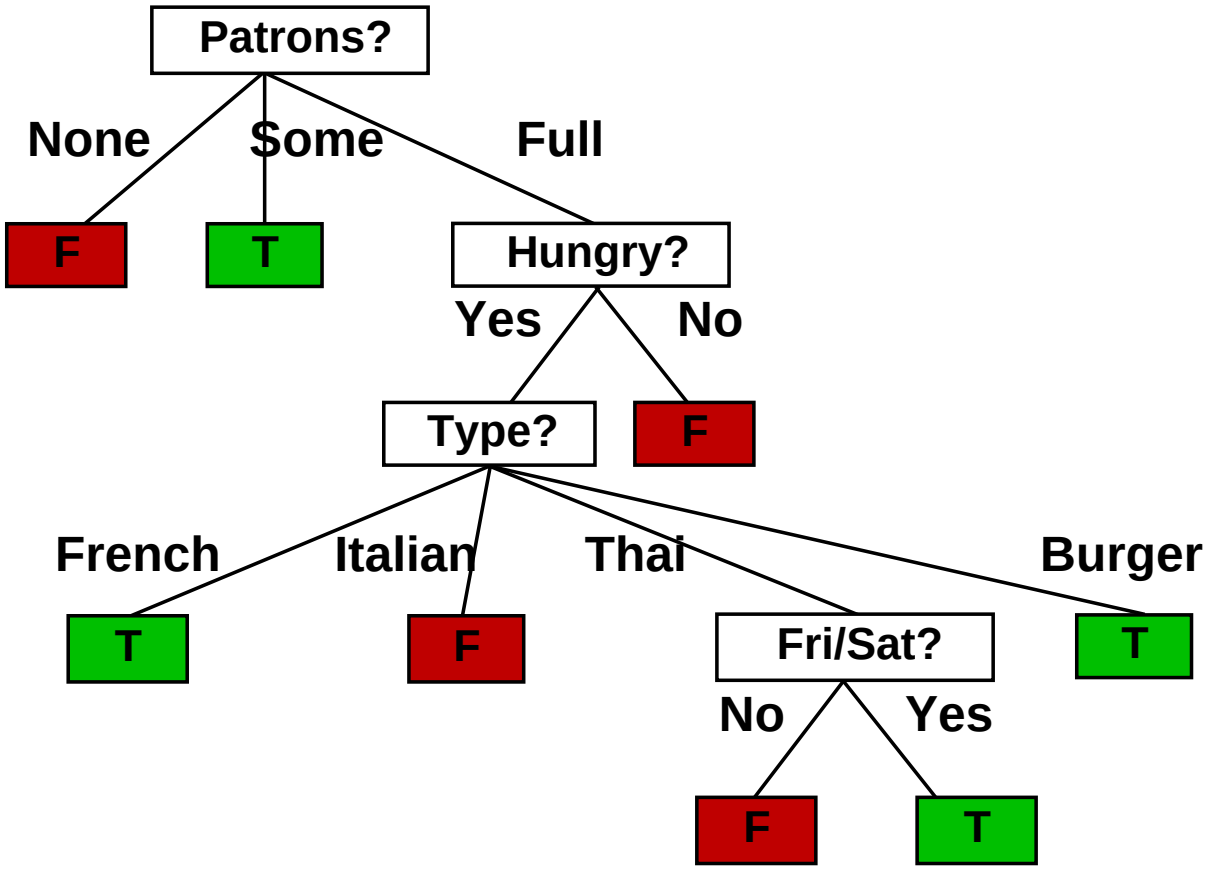
redukcja entropii przy podziale zbioru względem atrybutu a.

S_v – zbiór obiektów w S z wartością atrybutu $a = v$

$$G(S, a) = H(S) - \Sigma_{v \in Value(a)} \frac{\mid S_v \mid}{\mid S \mid} H(S_v)$$

Przykład cd.

Drzewo decyzyjne wyuczone z 12 przykładów



Atrybut numeryczny

W przypadku atrybutu numerycznego zbiór jego wartości dzielimy na dwa podzbiory poprzez wykonanie *cięcia*.

Zysk informacji obliczamy względem wybranego cięcia.

Zysk informacji $G(S, a, c)$=
redukcja entropii względem cięcia binarnego c na atrybucie a.

c – wartość cięcia
$S_{a<c}$ – zbiór obiektów z wartościami atrybutu a poniżej cięcia
$S_{a\geq c}$ – zbiór obiektów z wartościami atrybutu a powyżej cięcia

$$G(S,a) = H(S) - \frac{\mid S_{a<c} \mid}{\mid S \mid} H(S_{a<c}) - \frac{\mid S_{a\geq c} \mid}{\mid S \mid} H(S_{a\geq c})$$

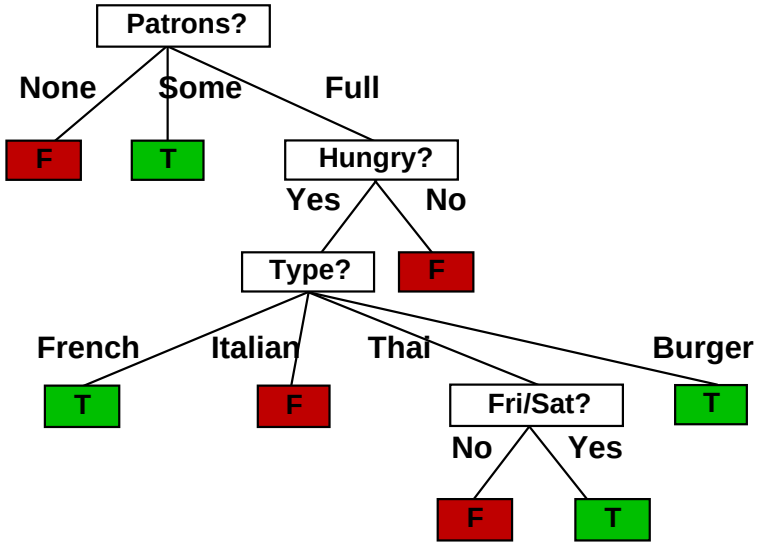
Wybór cięcia

Drzewo decyzyjne wybiera atrybut rozpatrując **najlepsze cięcia** dla atrybutów numerycznych.

Ten sam atrybut **numeryczny** może być wybrany **kilkakrotnie** na jednej ścieżce od korzenia do liścia.

Ale każdy atrybut **symboliczny** może być wybrany **conajwyżej raz**!

Klasyfikacja obiektu



Example	Attributes										Target
	Alt	Bar	Fri	Hun	Pat	$Price$	$Rain$	Res	$Type$	Est	*WillWait*
X	*T*	*F*	*F*	*T*	*Full*	*$$$*	*F*	*T*	*French*	*0–10*	*??*

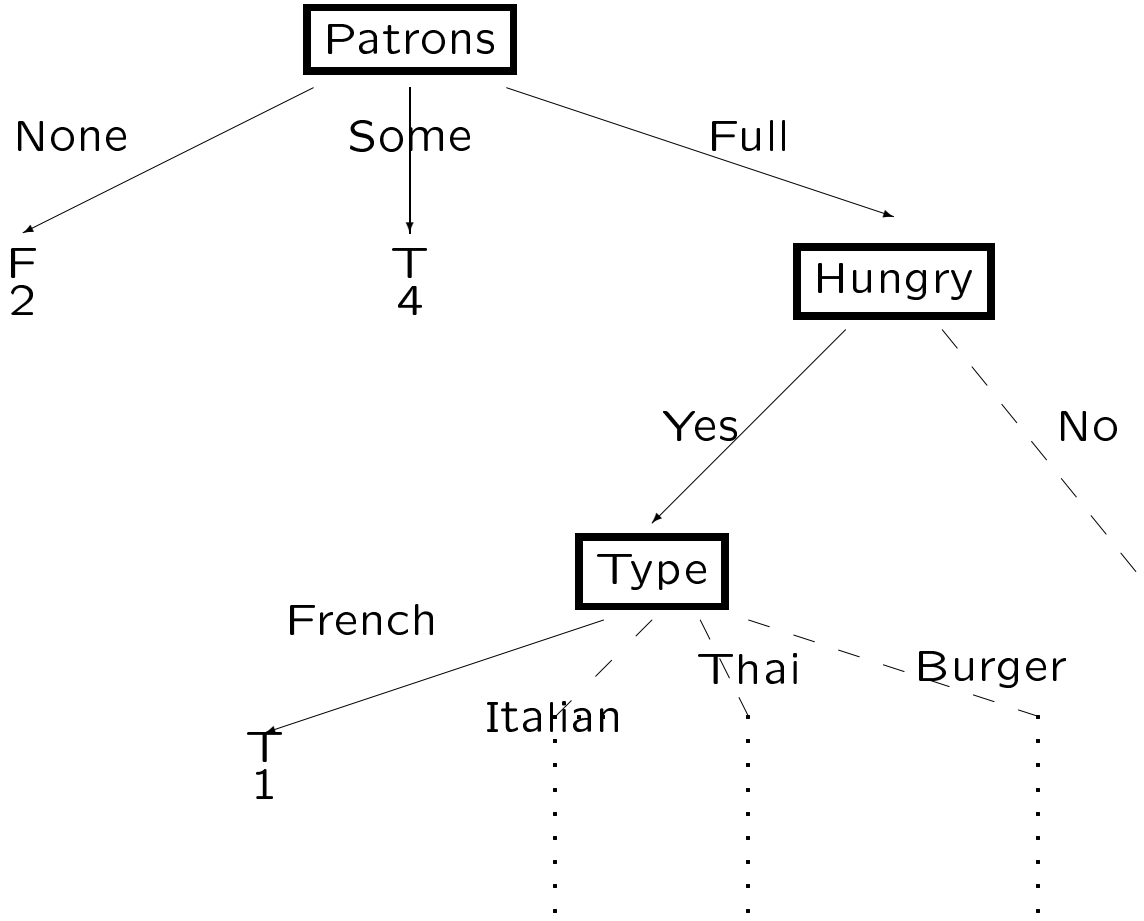
$$Patrons = Full \rightarrow Hungry = Yes \rightarrow Type = French \rightarrow WillWait = T$$

Klasyfikacja obiektu – brakujące wartości

Co robić, gdy informacja o klasyfikowanym obiekcie jest niepełna, np.

Example	Attributes										Target
	Alt	Bar	Fri	Hun	Pat	$Price$	$Rain$	Res	$Type$	Est	*WillWait*
X	*T*	*F*	*F*	*T*	*??*	*$$$*	*F*	*T*	*French*	*0–10*	*??*

Pomysł: zejście wszystkimi ścieżkami przy atrybutach z nieustaloną wartością



Odpowiedź:
Maksimum z sumy rozkładów decyzji obiektów uczących w osiągniętych liściach:

$$2 \times F \wedge (4 + 1) \times T \rightarrow WillWait = T$$

Przycinanie drzewa.

Problem:
Rzadkie wyjątki lub błędy w przykładach uczących mogą powodować niepotrzebne rozwinięcie gałęzi drzewa.

Rozwiązanie:
Dodanie fazy walidacji do procesu uczenia. Węzły rozdzielające, które nie potwierdzą swojej przydatności są zamieniane na liście.

Dokładniej:

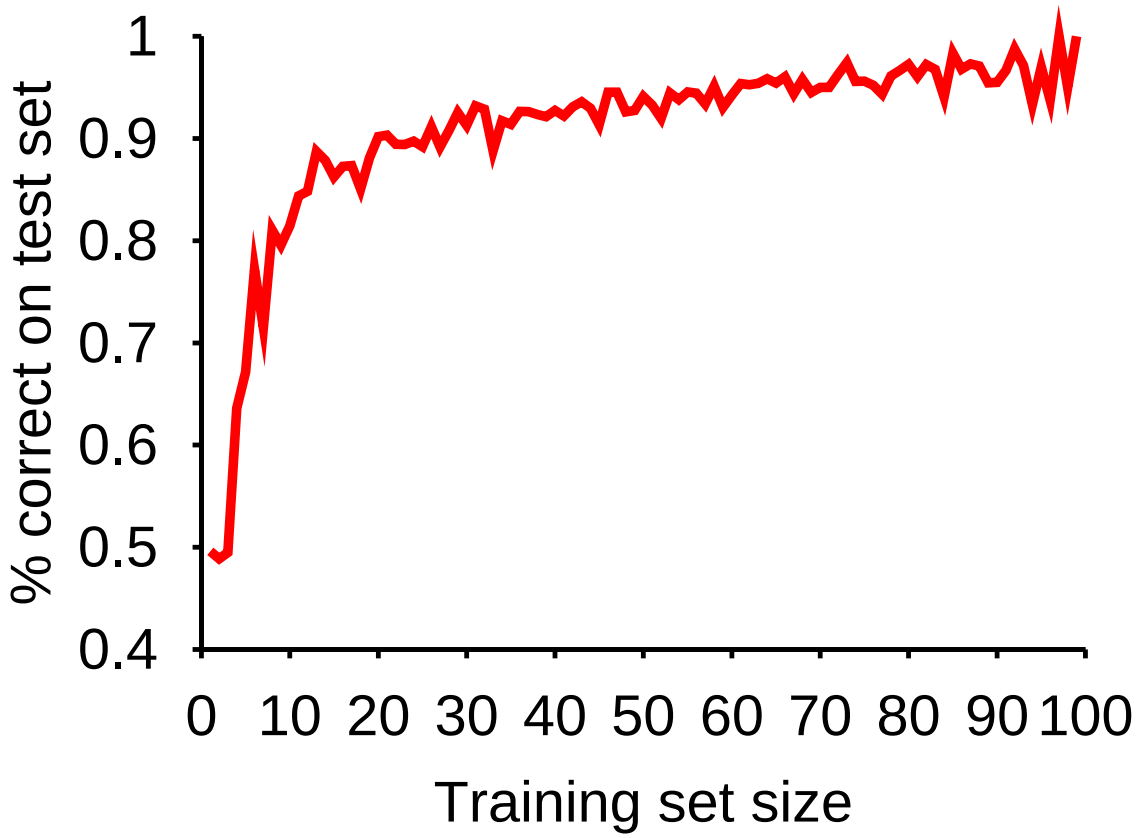
1. Wybieramy węzeł *cand*, którego wszystkie następniki są liśćmi.
2. Niech d_{cand} – najczęstsza decyzja przypisana obiektom budującym ten węzeł.
3. Jeśli zastąpienie węzła *cand* przez decyzję d_{cand} nie pogorszy skuteczności na zbiorze testowym, to zastąp podrzewo o wierzchołku *cand* przez liść z decyzją d_{cand}.
4. Powtarzaj 1.-3. dopóki zbiór kandydatów jest niepusty.

Miara efektywności.

Skąd wiadomo, że $h \approx f$

1) Teoria obliczeń/ statystyka
2) Testowanie funkcji h na nowym zbiorze testowym

Krzywa uczenia = % poprawności na zbiorze testowym jako funkcja jego rozmiaru

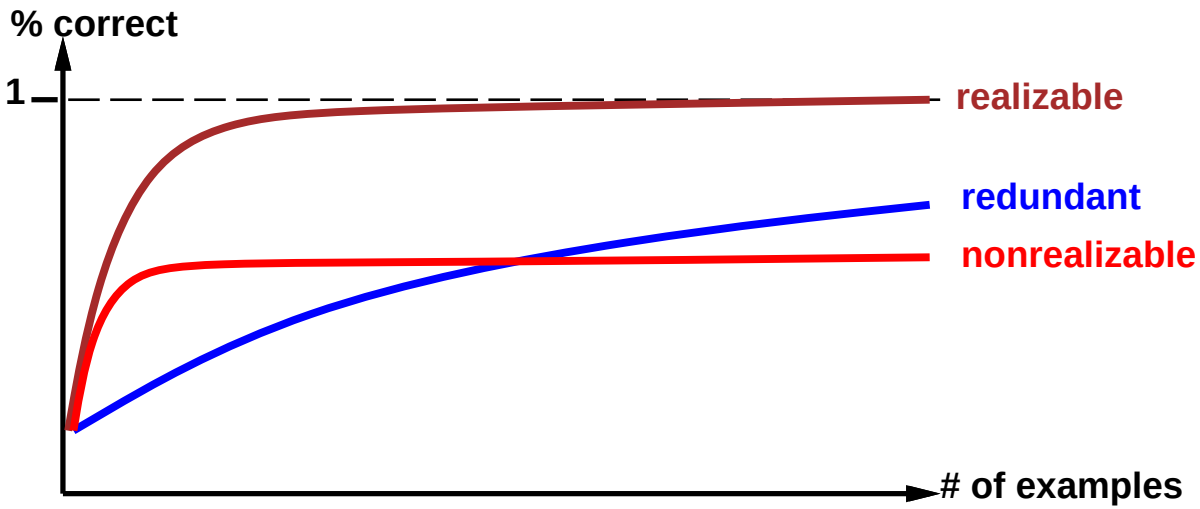


Miara efektywności cd.

Krzywa uczenia zależy od

- realizowalności (wyrażalność funkcji docelowej) vs. niereali-zowalności (spowodowanej np. brakiem atrybutu lub nad-miernym ograniczeniem klasy hipotez)

- redundancji w wyrażalności (np. nierelewantne atrybuty)


% correct
1
realizable
redundant
nonrealizable
# of examples