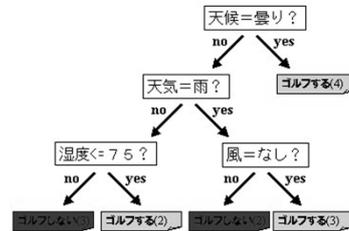


決定木 その7 まとめ

慶應義塾大学理工学部
櫻井彰人

これなら分りやすいか？

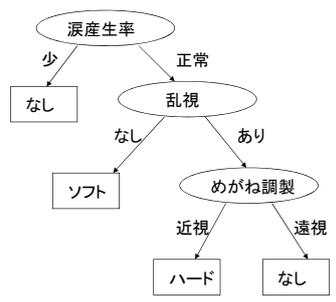


決定木と決定表

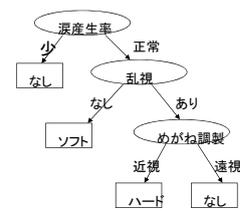


決定表

年齢	性別	乱視	涙産生率	確率
若い	近視	なし	少	ソフト
若い	近視	なし	正常	ソフト
若い	近視	あり	少	ソフト
若い	近視	あり	正常	ハード
若い	遠視	なし	少	ソフト
若い	遠視	なし	正常	ソフト
若い	遠視	あり	少	ソフト
若い	遠視	あり	正常	ハード
老眼以前	近視	なし	少	ソフト
老眼以前	近視	なし	正常	ソフト
老眼以前	近視	あり	少	ソフト
老眼以前	近視	あり	正常	ハード
老眼以前	遠視	なし	少	ソフト
老眼以前	遠視	なし	正常	ソフト
老眼以前	遠視	あり	少	ソフト
老眼以前	遠視	あり	正常	ハード
老眼	近視	なし	少	ソフト
老眼	近視	なし	正常	ソフト
老眼	近視	あり	少	ソフト
老眼	近視	あり	正常	ハード
老眼	遠視	なし	少	ソフト
老眼	遠視	なし	正常	ソフト
老眼	遠視	あり	少	ソフト
老眼	遠視	あり	正常	ハード



決定木



どんなものか

- 木
- 木の節(ノード)に「属性」
- 木の枝(エッジ)に「属性値」
- ただし、葉(これも節の一つ)には、ある特別な「属性値」

どう使うか

- 「属性=属性値」の組(決定表の一行)に対し、
- 根からスタートして、
- 自分の属性値に従って、枝をたどり、葉に至ると
- 葉には行すべき行動とか属するクラスの名称が書かれている

属性と属性値

- 属性:「表」で言えば、縦の欄
 - うるさく言えば、確率変数
 - 人間でいえば、身長、体重、年齢、生年月日、...
- 属性の名:「表」で言えば、縦の欄の名称
 - 確率変数の名称
 - 「身長」、「体重」、...
- 属性値:「表」に入っている値
 - 確率変数がとる値
 - 身長であれば、180cm というような連続数値や、「高い」「普通」「低い」と離散値(カテゴリ値)

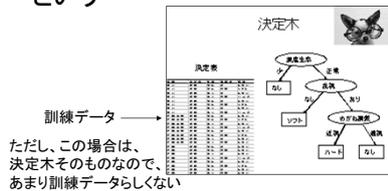
決定木の使い方



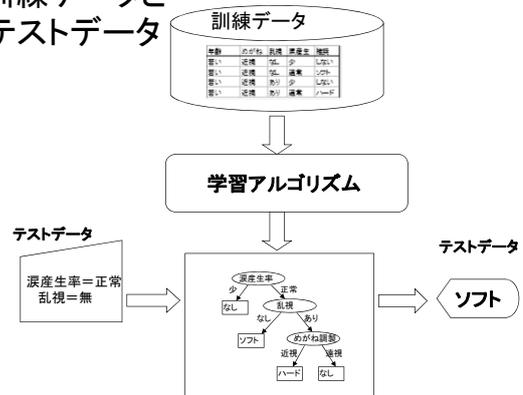
- 顧客に、「こういう時は、これがよい」という商品選択の方法を説明する
- 「顧客」でなくてもある。条件が複雑なときに決定木でかかっていると分りやすい
- 「危機対応マニュアル」

決定木の作り方

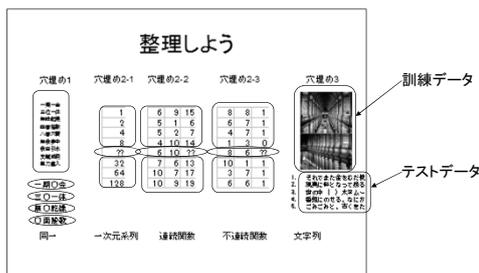
- 決定木を作ること考えよう
- 材料が必要
- 材料を、機械学習の分野では、「訓練データ」という



訓練データとテストデータ



訓練データとテストデータ



訓練データとテストデータ

- 訓練データとテストデータは、原則的には、別物
 - 丸暗記のテストをするときには、当然、同じもの。
 - 普通は、学習結果は、訓練データを一般化したもの。テストをするときには、いろいろなテストデータを用いてテストを行う。それゆえ、訓練データとテストデータは別物
 - 人間だって、自動車運転教習所で習った道路(訓練データ)と実際に走る道路(テストデータ)とは別物

決定木を作る時の課題

- 節(ノード)に置く属性を決めれば、自動的にできる(節を分割し、枝を伸ばすことができる)。

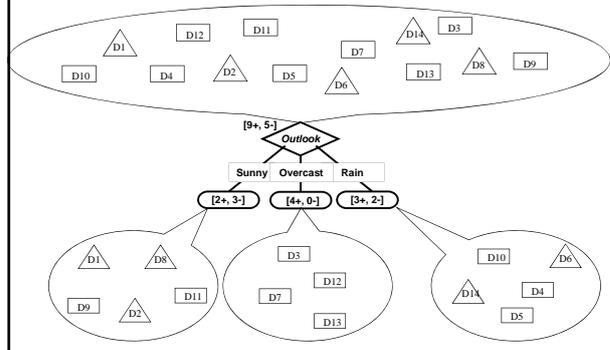
次のスライドの Hunt のアルゴリズムを参照

- 節(ノード)に置く属性の決め方が課題だ。 - どうしよう?

- 実は、もう2つあります。
- 一つは、「いつやめるか？」
- もう一つは、属性値がたくさんあるとき、属性値は、実はグループ分けした方がよい。それをどうするか？

講義は、
(1) Huntの方法
(2) 属性の決め方
と属性値のグループ化
(3) やめ方

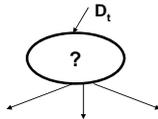
決定木の学習の1ステップ



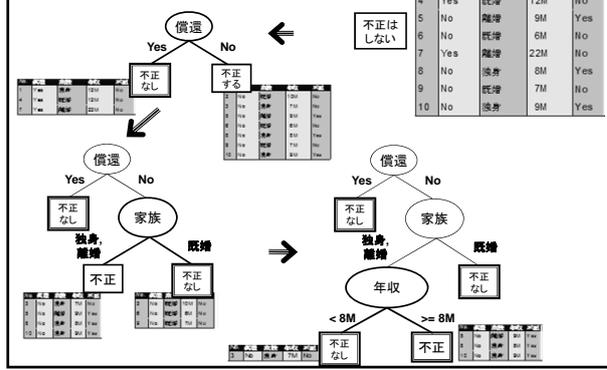
Hunt のアルゴリズムの構造

- D_t をノード t にたどりついた訓練データの集合とする
- 手続きの概要:
 - もし D_t に含まれるデータがすべて同じクラス y_t に属するなら, t は葉ノードであってそのラベルは y_t .
 - もし D_t が空集合なら, t は葉ノードであって, そのラベルは予め決めておくデフォルトラベル y_d となる
 - もし D_t に含まれるデータは複数個のクラスに属するとき, t には属性値のチェックを入れ, 訓練データ D_t をより小さな集合(部分集合)に分割する. この手続きを再帰的に, 当該部分集合に適用する.

No.	職業	家族	年収	不正
1	Yes	既婚	12M	No
2	No	既婚	10M	No
3	No	独身	7M	No
4	Yes	既婚	12M	No
5	No	離婚	9M	Yes
6	No	既婚	6M	No
7	Yes	離婚	22M	No
8	No	独身	8M	Yes
9	No	既婚	7M	No
10	No	独身	9M	Yes



Hunt のアルゴリズム

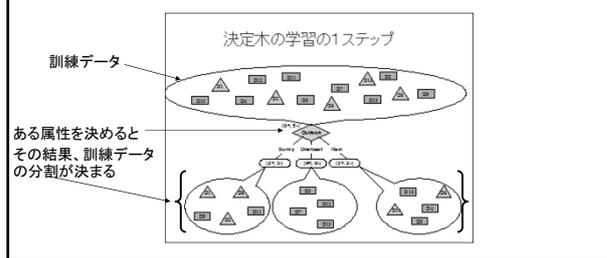


節に置く属性の決定

- グリーディな方略をとる.
 - いったん決めたら, 心変わりしない
 - 迷路を進むときに, 後戻りしない. 一度掘ったら離さない.
 - 最適ではないが, 後戻りしない分, 速い.
 - つまり, ある節に置く属性(と使う属性値グループ化)を決めたら, 撤回しない.
 - 一度分割してある枝を作ったら, それを取りやめることはない
 - その結果, 最適決定木となることは保証出来ない(一般には最適ではない)
- 課題
 - できるだけよい
 - 属性の選び方は?
 - 属性値のグループ化の仕方は?

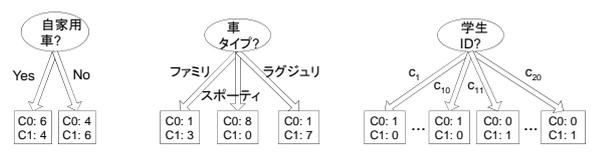
考え方

- 結局、「訓練データの分割方法として、どれがいいか」を考えればよいと気がつく



やはり、分割の問題(分割の良さを比較する問題)だ！ 最良な分割はどうやって見つける？

分割前: C0 (クラス0) に 10 データ,
C1 (クラス1) に 10 データ



どの条件が最適か？

“最良”の属性の選択

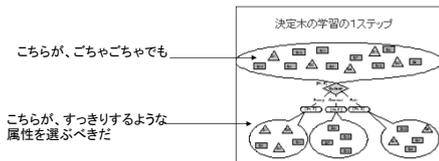
- 最良の分類木とは何だろうか？
 - 「正しい木である」こと
 - それはそうだが, 我々は, 出来上がった木が「正しい」かどうか分らない
 - 正解を知っている「先生」がいないのだ.
 - そこで, こう考えた。「これから来るデータを正しく分類できる」ならどうだろうか？
 - 「分類木」が間違っていないでも, 答え(新しいデータの分類結果)が正しければよい.
 - しかし, 「これから来るデータ」は, 入手できないよね.
 - そこで, こう考えた. 与えられたデータを, 訓練データとテストデータに分けよう.
 - 訓練データで分類木を作り, テストデータでテストすればよい.
 - これはすごくいい考えだ. けれども, これでは, 訓練データを使って一個分類木を作って, テストデータで調べるだけであって(点数が分る. 評価するとう), それがいいものかどうか分らない.
 - そこで, こう考えた. 最もよい分類木を作るためには, 色々な訓練データを作ってみればよい. ランダムに訓練データを作って, テストデータでテストし, テスト結果が最良のものを選べばよい. これはよい考えだ!

“最良”の属性の選択

- 最良の分類木とは何だろうか？
 - いや、そうでもない。それなら、最初から、全データを正しく分類する分類木を作れば、よいでしょう。どんなテストデータに対しても正しく分類する。
 - これは困った。だめか、やっぱり、テストデータを使って最も良いものを選んでほしいんだ。
 - そこで、こう考えた。最もよいものを選ぶのはやめよう。身の回りで一番よいもので我慢しよう。
 - (1) その方法の一つとして、訓練データは固定して、そのもとの、いくつか決定木を作り、テストデータで評価して、一番よいものを使おう。 ← 例: pruning法
 - (2) もう一つは、昔の哲学者の言葉に従うことだ。オッカムが言うには「データを説明する同じような仮説が複数あるときには、一番短い仮説をどれ」
 - 実は方法(1)の場合も、「短い」決定木が選ばれることが知られている。

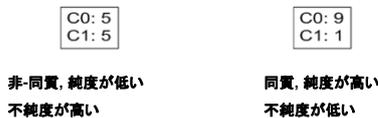
“最良”の属性の選択

- 最良の分類木とは何だろうか？
 - 間違いが同じ程度なら、一番小さい分類木がよいと、いえる
 - 間違いが同程度でないときどうするかは、少々難しい問題ゆえ、ここでは省略。
 - そうであれば、「最良の属性」とは、その属性を選んだら分類木が小さくなりそうな属性であろう。
 - となると、「最良の属性」とは、その属性を選んだら、(訓練データが分割されるわけだが)分割後の訓練データが、各分割内で、そろっているような属性だ。

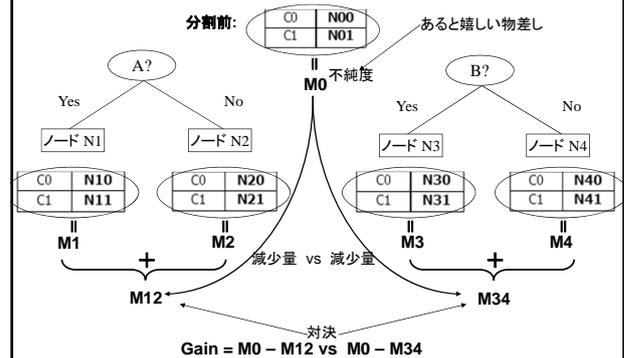


最良な属性、最良な分割は？ つまり

- 「決定！」に近くなる方がよい、すなわち:
 - 新ノード内のクラス分布が 同質となる 分割がベター
 - どこかのクラスが圧倒的な多数となる(これが同質)ということは、それだ！ といっても間違いが少ないから
- そのためには、(ノードの)同質さの物差しが必要:



最良な属性、最良な分割は？



不純度のものさし

- エントロピー
- ジニ・インデックス Gini Index
- 誤分類率

エントロピー

- 平均情報量とも呼ばれる。式で書くと

$$H(p_1, \dots, p_m) = -p_1 \log_2 p_1 - \dots - p_m \log_2 p_m$$

$$= p_1(-\log_2 p_1) + \dots + p_m(-\log_2 p_m)$$
- (比較のために)サイコロの出目の平均

$$+ p_1 * 1 + p_2 * 2 + \dots + p_6 * 6$$
- つまり、平均情報量が情報量の平均だとすると $-\log_2 p_i$ が「情報量」ということになる

負の符号「-」がついているのは、 $p < 1$ 故 $\log p < 0$ となるが、負の数はいろいろと不便なため、符号反転しているから。

情報量

ある事象の情報量は、その事象が起こったということ(他の皆が知らないときに)知るこの価値

「事象」として「コインの表が出ること」(確率1/2)としよう。
「表が出たこと」を知る価値を a としよう。
「コイン1が表」「コイン2が表」という2つの情報を知る価値は $a + a = 2a$ だろう(一つずつ聞く場合を考えればよい)。
「コイン1が表」「コイン2が表」の二つの事象が起こる確率は $\frac{1}{2} * \frac{1}{2} = 1/4$ 。
「事象」として「サイコロの1が出ること」(確率1/6)としよう。
「1が出たこと」を知る価値を b としよう。
「サイコロ1が1」「サイコロ2が1」という2つの情報を知る価値は $b + b = 2b$ だろう(一つずつ聞く場合を考えればよい)。
「コイン1が表」「コイン2が表」の二つの事象が起こる確率は $1/6 * 1/6 = 1/36$ 。
つまり、事象が起こる確率が2乗になると、価値は2倍になる



情報量を表す関数

事象が起こる確率が2乗になると、価値は2倍になる

事象が起こる確率 p が p^2 になると、価値 v は $2v$ になる

事象が起こる確率 p が p^2 になると、価値 v(p) は $v(p^2) = 2v(p)$ になる

上記のような関数は log しかないことが示せる(底は決まらない。何でもよい)

そこで、底を2とし価値が正になるように符合反転すると(底を1/2にしたのと同じ)、生起確率 p の事象が生じたことを知るという情報の価値は、 $-\log p$ とすればよいことが分る。

$$\text{情報量}(p) = -\log_2 p$$

不公平かもしれないコイン

- 表が出る確率 p, 裏が出る確率が 1-p であるコインのコイン投げを考える。
- このコインを1回投げたときに出た「表・裏」を知る情報の価値はどのくらいであろうか？
- 「表が出る」という情報の価値は、 $-\log p$, 「裏が出る」という情報の価値は、 $-\log(1-p)$ である。
- 表が出る確率は p, 裏が出る確率は 1-p であるので、この確率に基づく(情報価値の)平均値を考えよう

$$H(p, 1-p) = p(-\log_2 p) + (1-p)(-\log_2(1-p)) \\ = -p \log_2 p - (1-p) \log_2(1-p)$$

不公平かもしれないサイコロ

- 目が出る確率 p_i であるサイコロを考える。
- このサイコロを1回投げたときに出た目を知る情報の価値はどのくらいであろうか？
- 「目が出る」という情報の価値は、 $-\log p_i$ である。
- この確率に基づく(情報価値の)平均値を考えよう

$$H(p_1, p_2, \dots, p_6) \\ = p_1(-\log_2 p_1) + p_2(-\log_2 p_2) + \dots + p_6(-\log_2 p_6) \\ = -p_1 \log_2 p_1 - p_2 \log_2 p_2 - \dots - p_6 \log_2 p_6$$

不純度を測る物差しとしての情報量

- 不純度
 - 2種類の個体が混在している場合を考える。割合を p と 1-p とする。p=0 または p=1 のときは最も純粋であるので、このとき0、p=1/2 のとき最も純度が低いので、このとき1になるような関数があるとよい。明らかにエントロピーがその性質を満たす。
 - 一般に n 種類の個体が混在している場合はどうか。割合を p_1, \dots, p_n とする。p_i のいずれかが1で他が0というとき最も純度が高い。逆に p_i のすべてが等しいとき(1/nの時)最も純度が低い。明らかにエントロピーはこの性質をもつ。そこで、

$$\text{non-Information}(D) = H(D) = \sum_{c \in \text{class}(D)} \left[-\frac{|D_c|}{|D|} \log \frac{|D_c|}{|D|} \right]$$

- 補足: エントロピー値は、「集合の要素一個あたり」の情報量となっている

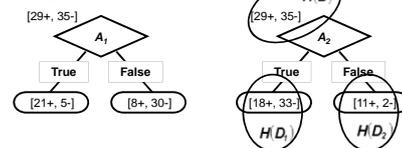
情報量増分

- 定義
 - 属性 A に関する D の情報量増分は、A を用いた分割によるエントロピー減少分の期待値:

$$\text{InformationGain}(D, A) = H(D) - \sum_{v \in \text{value}(A)} \left[\frac{|D_v|}{|D|} H(D_v) \right] = \frac{1}{|D|} \left(|D| H(D) - \sum_{v \in \text{value}(A)} |D_v| H(D_v) \right)$$

- 但し D_v は $\{x \in D \mid x(A) = v\}$, すなわち、D 中の事例で属性 A の値が v であるものの集合
- 補足: A による分割によって生じる部分集合 D_v の大きさに従ってエントロピーの大きさを調整
 - エントロピー値は、「集合の要素一個あたり」の情報量となっているため

- どちらの属性を使うのがいい?



GINI に基づく分割基準

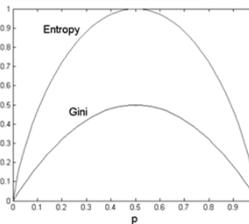
- これまで説明してきた分割基準はエントロピーであった:
(注: $p(j|t)$ はノード t におけるクラス j データの相対頻度)
- 別法に GINI インデックスを用いるものがある:

$$Entropy(t) = - \sum_j p(j|t) \log p(j|t)$$

$$GINI(t) = 1 - \sum_j [p(j|t)]^2$$

- 両者とも:
 - 最大値 ($\log n_c$ または $1 - 1/n_c$) が得られるのは、当該データがどのクラスにも等分に分配されているときである。「等分である」ということは何の面白さもない。しかし、この状態で、ずっと「実はこれ！」と教わり続けることは結構価値のあることである。
 - 最小値 (0.0) に近い値が得られるのは、ほとんどすべてのデータが同一のクラスに属するとき、少数派が発生する場合は、非常に面白い。けれども、たいていは多数派が発生するので、まったく面白くない。

2個のクラスに分ける場合:



あらためて: 決定木の構築

- 通常の手順: 上から下に(根から葉へ)、再帰的かつ分割統治 (divide-and-conquer)
 - まずは: 一つの属性を選び根とする。属性値ごとに枝を作る
 - 次は: 訓練データを部分集合に分割 (枝一本につき一個)
 - 最後に: 同じ手順を、個々の枝について行う。その場合、個々の枝に割り当てられた訓練データのみを用いる (全体は用いない)
- ノードに(それへの枝に)割り当てられた訓練データがすべて同じクラスになったら、終了

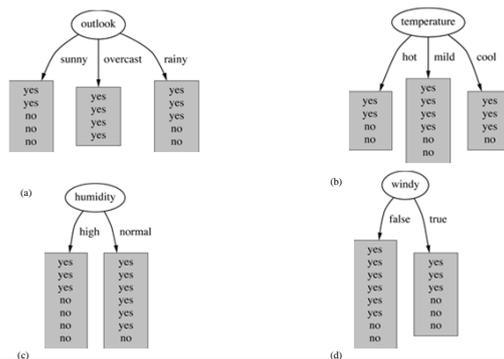
これは「選すぎ」「作りすぎ」「やりすぎ」です。
どうしてでしょうか?
どうしたらよいでしょうか?

テニスをするや否や

Outlook	Temp.	Humidity	Windy	Play
Sunny	Hot	High	False	No
Sunny	Hot	High	True	No
Overcast	Hot	High	False	Yes
Rainy	Mild	High	False	Yes
Rainy	Cool	Normal	False	Yes
Rainy	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Sunny	Mild	High	False	No
Sunny	Cool	Normal	False	Yes
Rainy	Mild	Normal	False	Yes
Sunny	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes
Rainy	Mild	High	True	No

Tom Mitchell "Machine Learning" の例題。よく使われる

どの属性がいいのか?



属性選択の基準

- どの属性が最適化?
- できあがる決定木が最小のものがよい
 - ヒューリスティック: 「純度」最高の属性を選ぶ
 - 「最小」のものを選ぶことに関し、深遠な議論がある
- 良く使われる「不純度」の基準: (ノードの)エントロピー
 - エントロピーが低いほど、ノードの「純度」は高い。
- 方略: 子供のノードのエントロピーが最小となる属性を選べ。

計算例: 属性 "Outlook"

"Outlook" = "Sunny":
 $info(2,3) = entropy(2/5, 3/5) = - (2/5) \log(2/5) - (3/5) \log(3/5) = 0.971$

"Outlook" = "Overcast":
 $info(4,0) = entropy(1,0) = - 1 \log(1) - 0 \log(0) = 0$

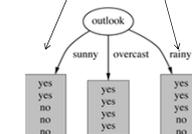
"Outlook" = "Rainy":
 $info(3,2) = entropy(3/5, 2/5) = - (3/5) \log(3/5) - (2/5) \log(2/5) = 0.971$

この属性を用いたときの情報量は
 $info(3,2, [4,0], [3,2]) = \frac{5}{14} \times 0.971 + (4/14) \times 0 + (5/14) \times 0.971 = 0.693 \text{ bits}$

non - Information $n(D_v) = H(D_v) = - \sum_{j \in \text{values}(D)} \frac{|D_{v,j}|}{|D_v|} \log \frac{|D_{v,j}|}{|D_v|}$

Information Gain $G(D, A) = H(D) - \sum_{j \in \text{values}(A)} \frac{|D_j|}{|D|} H(D_j)$

$= \frac{1}{|D|} \left(|D| H(D) - \sum_{j \in \text{values}(A)} |D_j| H(D_j) \right)$



情報量増分 Information gain

- ただし、通常は、ノードのエントロピーを直接用いることはない。情報量増分を用いる。

情報量増分: 分割前の情報量 - 分割後の情報量
 $gain("Outlook") = info([9,5]) - [info([2,3],[4,0],[3,2])] = 0.940 - 0.693 = 0.247 \text{ bits}$

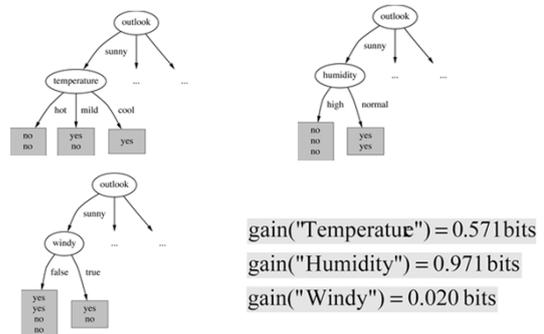
同様に計算すると
 $gain("Outlook") = 0.247$
 $gain("Temperature") = 0.029$
 $gain("Humidity") = 0.152$
 $gain("Windy") = 0.048$

$$Information\ Gain(D, A) = \frac{|D|}{|D|} \sum_{v \in \text{value}(A)} \frac{|D_v|}{|D|} H(D_v)$$

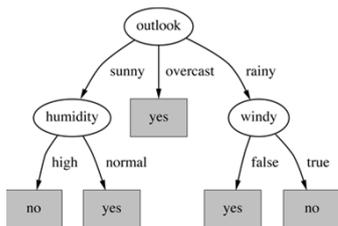
$$= \frac{1}{|D|} \left(|D| H(D) - \sum_{v \in \text{value}(A)} |D_v| H(D_v) \right)$$

- 情報量増分が多いほど、純度が高い。従って、“Outlook”を選ぶことにする。

分割を続ける



最終的に得られる決定木



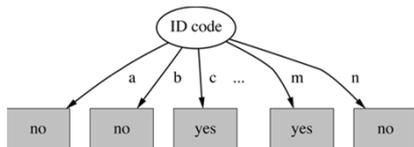
- 注: すべての葉が“純”である必要はない; というのも、同じデータなのにクラスが違うことがあるから(ノイズのせい)
 ⇒ データをそれ以上分割しない方がよくなったら、やめ

枝数が非常に多くなる属性があると、、、

- IDコードをつけてみよう

ID code	Outlook	Temp.	Humidity	Windy	Play
A	Sunny	Hot	High	False	No
B	Sunny	Hot	High	True	No
C	Overcast	Hot	High	False	Yes
D	Rainy	Mild	High	False	Yes
E	Rainy	Cool	Normal	False	Yes
F	Rainy	Cool	Normal	True	No
G	Overcast	Cool	Normal	True	Yes
H	Sunny	Mild	High	False	No
I	Sunny	Cool	Normal	False	Yes
J	Rainy	Mild	Normal	False	Yes
K	Sunny	Mild	Normal	True	Yes
L	Overcast	Mild	High	True	Yes
M	Overcast	Hot	Normal	False	Yes
N	Rainy	Mild	High	True	No

IDコードを根にもってくると、“切株”



この分割のエントロピー
 $info("IDcode") = info([0,1]) + info([0,1]) + \dots + info([0,1]) = 0 \text{ bits}$
 ⇒ 情報量増分は最大となる(すなわち、0.940 bits)

従って、属性値が多いと、訓練データの部分集合は“純”になりやすい

増分比

- 増分比 Gain ratio: 情報量増分のもつバイアスを減少させる
- 増分比は、枝の本数とそれに割り当てられる訓練データの大きさの両方を勘定に入れる
 - 情報量増分の修正は、訓練データの集合をどのような(大きさと要素数の)部分集合に分割するかという分割の情報量を用いて、行われる

増分比の計算例

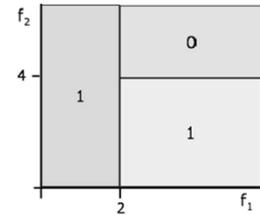
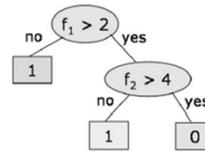
計算例: IDコードの分割情報量 (split information)
 $\text{info}([1,1,\dots,1]) = 14 \times (-(1/14) \log(1/14)) = 3.807 \text{ bits}$

増分比の定義
 $\text{gain_ratio}(\text{"Attribute"}) = \text{gain}(\text{"Attribute"}) / \text{split_info}(\text{"Attribute"})$

計算例:
 $\text{gain_ratio}(\text{"IDcode"}) = 0.940 \text{ bits} / 3.807 \text{ bits} = 0.246$

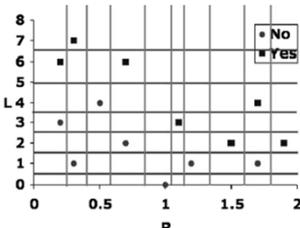
数値属性

- 勿論、これでもいい $x_j >$ ある定数
- 短冊への分割は同じ



分割によるエントロピーを計算

境界	下方にある No の個数	下方にある Yes の個数	上方にある No の個数	上方にある Yes の個数	エントロピー
6.5	7	6	0	1	0.93
5.0	7	4	0	3	0.74
3.5	6	3	1	4	0.85
2.5	5	2	2	5	0.86
1.5	4	0	3	7	0.63
0.5	1	0	6	7	0.93

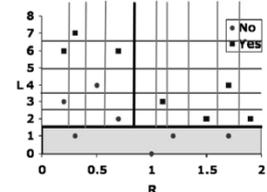


エントロピー	1.00	1.00	0.98	0.98	0.94	0.98	0.92	0.98	0.92
境界	0.25	0.40	0.60	0.85	1.05	1.15	1.35	1.60	1.80

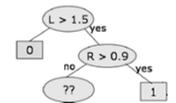
承前

- 今度の最適な分割は $R > 0.9$ である。しかも、すべて Yes であるので、葉を作ることができる。

境界	下方にある No の個数	下方にある Yes の個数	上方にある No の個数	上方にある Yes の個数	エントロピー
6.5	3	6	0	1	0.93
5.0	3	4	0	3	0.74
3.5	2	3	1	4	0.85
2.5	1	2	2	5	0.86

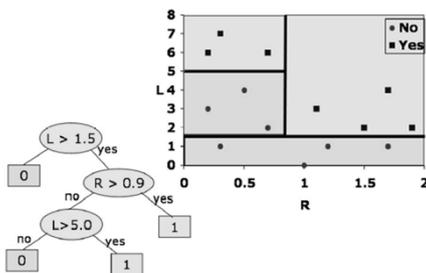


エントロピー	0.85	0.88	0.79	0.60	0.69	0.76	0.83
境界	0.25	0.40	0.60	0.90	1.30	1.60	1.80



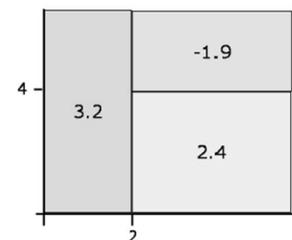
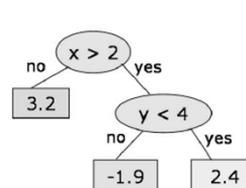
承前

- これを続ければ次のものが得られる:



回帰木 Regression Trees

- 決定木と同じ、但し葉において、実数値定数を出力する。

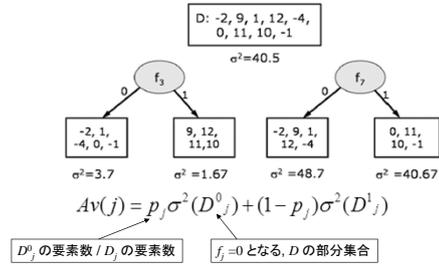


葉における値

- 今いる葉ノードには複数個のデータがあると仮定しよう。なおかつ、何らかの理由により、このノードはこれ以上分割しないものとする。
- 離散値の場合(これまでの場合)、葉における値(出力値)は、その葉における多数派の値としていた。
- 数値属性の場合、妥当な値は平均値であろう。
- 従って、もし葉ノードにおける出力値として平均値を用いるならば、(これからノードを分割して子供が葉ノードになろうというときには)枝分かれして作られる新たな葉ノードにおいて、データのもつ値が、当該葉ノード内の値の平均値よりあんまり離れていない方がよからう。
- 統計学には、数値の集合がどのくらい分散しているかを表す尺度がある
 - (言い換えれば、個々の数値が平均値からどれだけ離れているか);
 - ご存じの分散である。

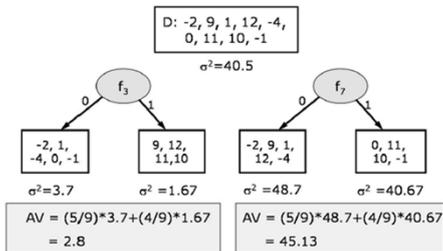
ノード分割

- 離散値の場合と同様に、分散の平均値を計算するとき、各新ノードに渡されるデータ数で重み付けた、重み付き平均値を用いることにする(そうでなければならぬ)。



ノード分割

- 簡単に計算できることだが、属性3を用いた分割の場合の分散平均値は、属性7を用いた場合のそれより、非常に小さい。従って、属性3を選ぶことになる。



まとめ

- 決定木は、木。ノードに「属性」、葉に「決定」
 - 機械学習・データマイニングで、データの規則性を説明する、一つの道具
 - 属性は、離散属性が主。数値属性も扱える。
- 決定木は、説明するのに便利。精度はいまいち
- 作るときは、グリーディに、「属性選択と枝分かれ」
- 属性選択は、結果ができるだけ「純」(「決定」が同じ)になるように。
 - 連続値属性も似た考えで。