クラス分類問題
Classification Problem

（教師つき学習
Supervised Learning）

決定木

2000年ごろからバイオデータ解析にデータマイニングを利用することが盛んになる

KDD Cup 2001

Because of the rapid growth of interest in mining biological databases, KDD Cup 2001 was focused on data from genomics and drug design. Sufficient (yet concise) information was provided so that detailed domain knowledge was not a requirement for entry. A total of 136 groups participated to produce a total of 200 submitted predictions over the 3 tasks: 114 for Thrombin, 41 for Function, and 45 for Localization.

KDD Cup 2001 Winners

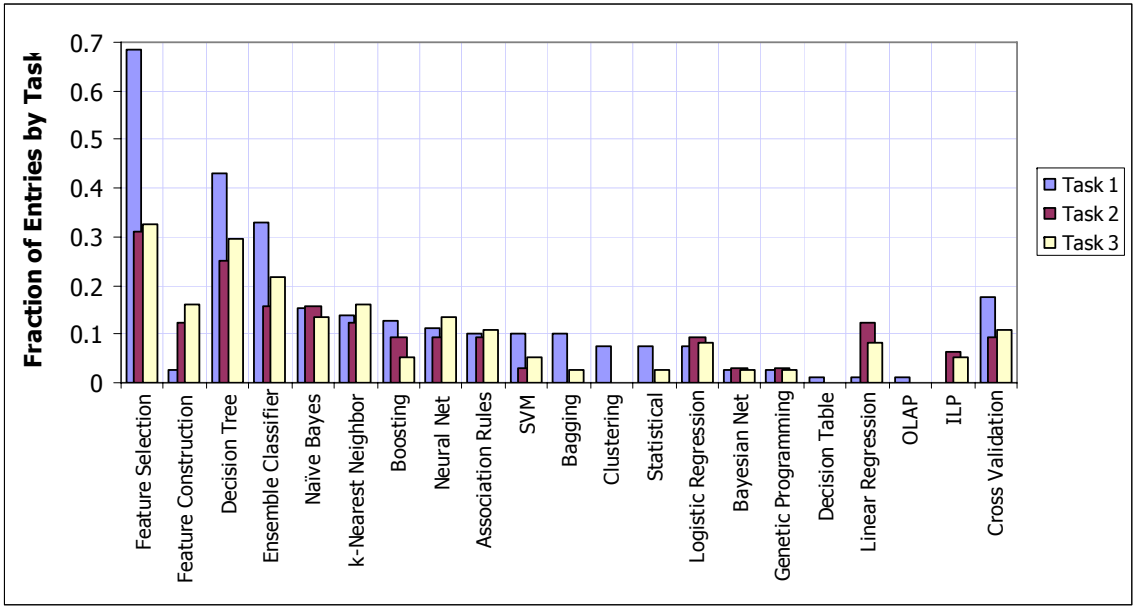
The KDD Cup summary presentation from KDD-2001 is available in powerpoint, postscript, or pdf. Winners' presentations are available below.

- Task 1, Thrombin: Jie Cheng (Canadian Imperial Bank of Commerce). Presentation: powerpoint, postscript, pdf.
- Task 2, Function: Mark-A. Krogel (University of Magdeburg). Presentation: powerpoint, postscript, pdf.
- Task 3, Localization: Hisashi Hayashi, Jun Sese, and Shinichi Morishita (University of Tokyo). Presentation: powerpoint, postscript, pdf.

KDD Cup 2001 Honorable Mention

- Task 1, Thrombin: T. Silander (University of Helsinki)
- Task 2, Function: C. Lambert (Golden Helix); J. Sese, H. Hayashi, and S. Morishita (University of Tokyo); D. Vogel and R. Srinivasan (A.I. Insight); S. Pocinki, R. Wilkinson, and P. Gaffney (Lubrizol Corp.)
- Task 3, Localization: M. Schonlau (RAND), W. DuMouchel, C. Volinsky and C. Cortes (AT&T); B. Frasca, Z. Zheng, R. Parekh, and R. Kohavi (Blue Martini)

- ACM　SIGKDD 主催のコンテスト KDD (Knowledge Discovery and Data-mining) Cup. 1997年より開催.
- 2001年は遺伝子機能予測, 蛋白質局在予測, ドラッグデザインが出題.
- 延べ 200 チームが参加.
- 知識抽出用訓練データ(目標属性値あり)が出題.
- つづいて予測用テストデータ(目標属性値がない！)が与えられ、目的属性の的中精度で順位.

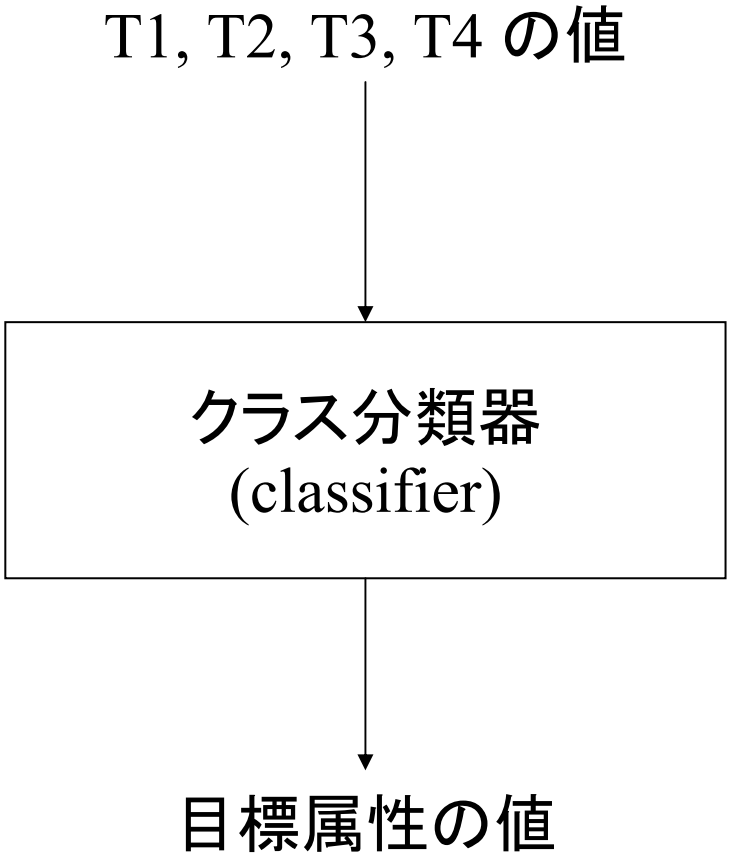


出展　Jie Cheng, Christos Hatzis, Hisashi Hayashi, Mark-A. Krogel, Shinichi Morishita , David Page, and Jun Sese: KDD Cup 2001 Report. *ACM SIGKDD Explorations, Volume* 3, Issue 2, Janurary 2002, 47-64.

クラス分類器 (Classifier) の様々な手法

- 決定木
- Boosting
- Support Vector Machine
- k-nearest neighbor
- Naïve Bayes
- ニューラルネットワーク
- Hidden Markov Model

T1	T2	T3	T4	目標属性
1	0	1	1	1
1	0	1	1	1
1	1	1	1	1
1	1	1	0	0
1	0	1	0	0
1	1	0	1	0
1	0	0	1	0
1	1	0	1	0
0	1	0	1	1
0	0	1	1	1
0	1	0	1	1
0	1	0	1	1
0	0	0	1	1
0	0	1	0	0
0	1	0	0	0
0	0	1	0	0

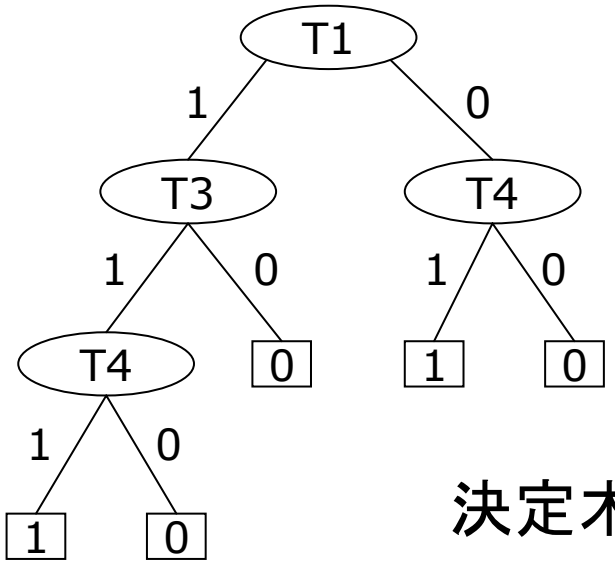


T*i*: 遺伝子発現量の増減, 遺伝子の局在, SNP, ハプロタイプ　等
目標属性: ある表現型が出たか否か, 疾患か否か 等

テスト

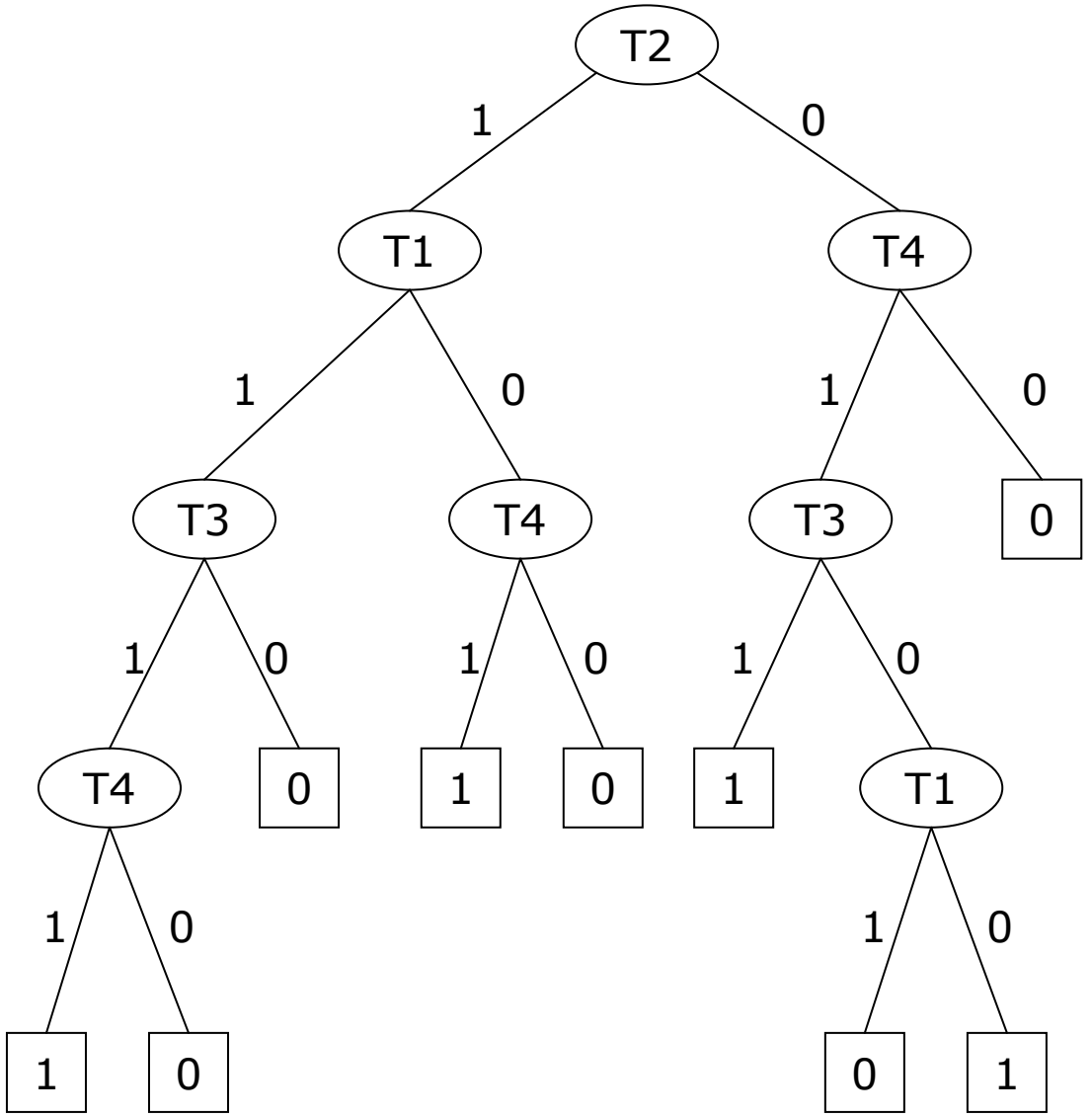
T1	T2	T3	T4	目標属性
1	0	1	1	1
1	0	1	1	1
1	1	1	1	1
1	1	1	0	0
1	0	1	0	0
1	1	0	1	0
1	0	0	1	0
1	1	0	1	0
0	1	0	1	1
0	0	1	1	1
0	1	0	1	1
0	1	0	1	1
0	0	0	1	1
0	0	1	0	0
0	1	0	0	0
0	0	1	0	0

```
If T1=1
Then
        If T3=1
        Then
                If T4=1
                Then    目標属性＝1
                Else    目標属性＝0
        Else
                目標属性＝0
Else
        If T4=1
        Then    目標属性＝1
        Else    目標属性＝0
```



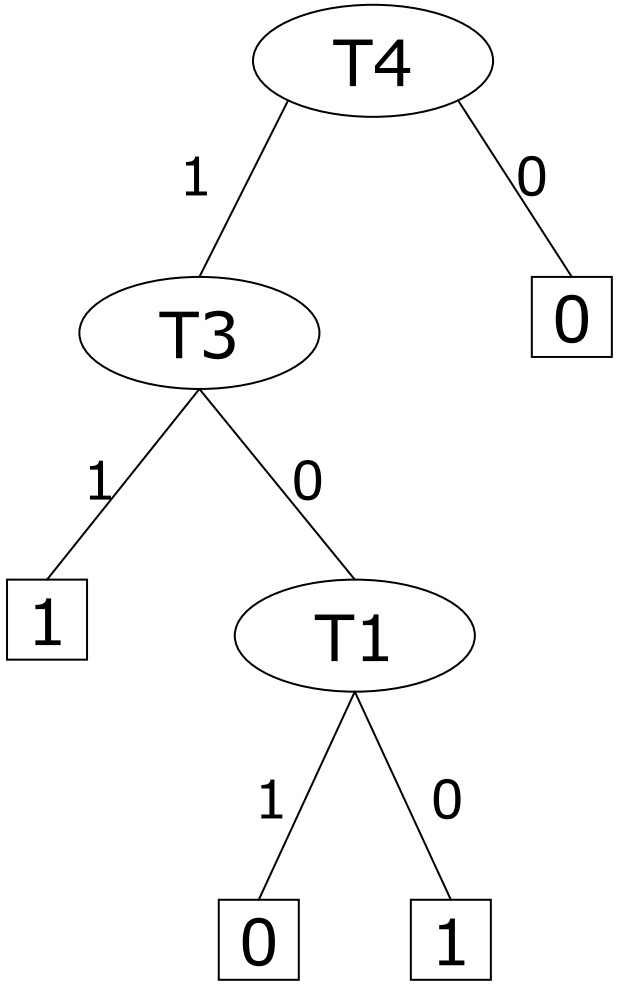
決定木(decision tree)

T1	T2	T3	T4	目標属性
1	1	1	1	1
1	1	1	0	0
1	1	0	1	0
1	1	0	1	0
0	1	0	1	1
0	1	0	1	1
0	1	0	1	1
0	1	0	0	0
1	0	1	1	1
1	0	1	1	1
0	0	1	1	1
0	0	0	1	1
1	0	0	1	0
0	0	1	0	0
1	0	1	0	0
0	0	1	0	0



テストの選択の順番を変更すると決定木の大きさも変化

T1	T2	T3	T4	目標属性
1	0	1	1	1
1	0	1	1	1
1	1	1	1	1
0	0	1	1	1
0	1	0	1	1
0	1	0	1	1
0	1	0	1	1
0	0	0	1	1
1	1	0	1	0
1	0	0	1	0
1	1	0	1	0
0	0	1	0	0
0	0	1	0	0
1	1	1	0	0
1	0	1	0	0
0	1	0	0	0



各葉ノードにおいて、到達するレコードがすべて正であるか、すべて負であるかのいづれかが成り立つ決定木を生成することを考える

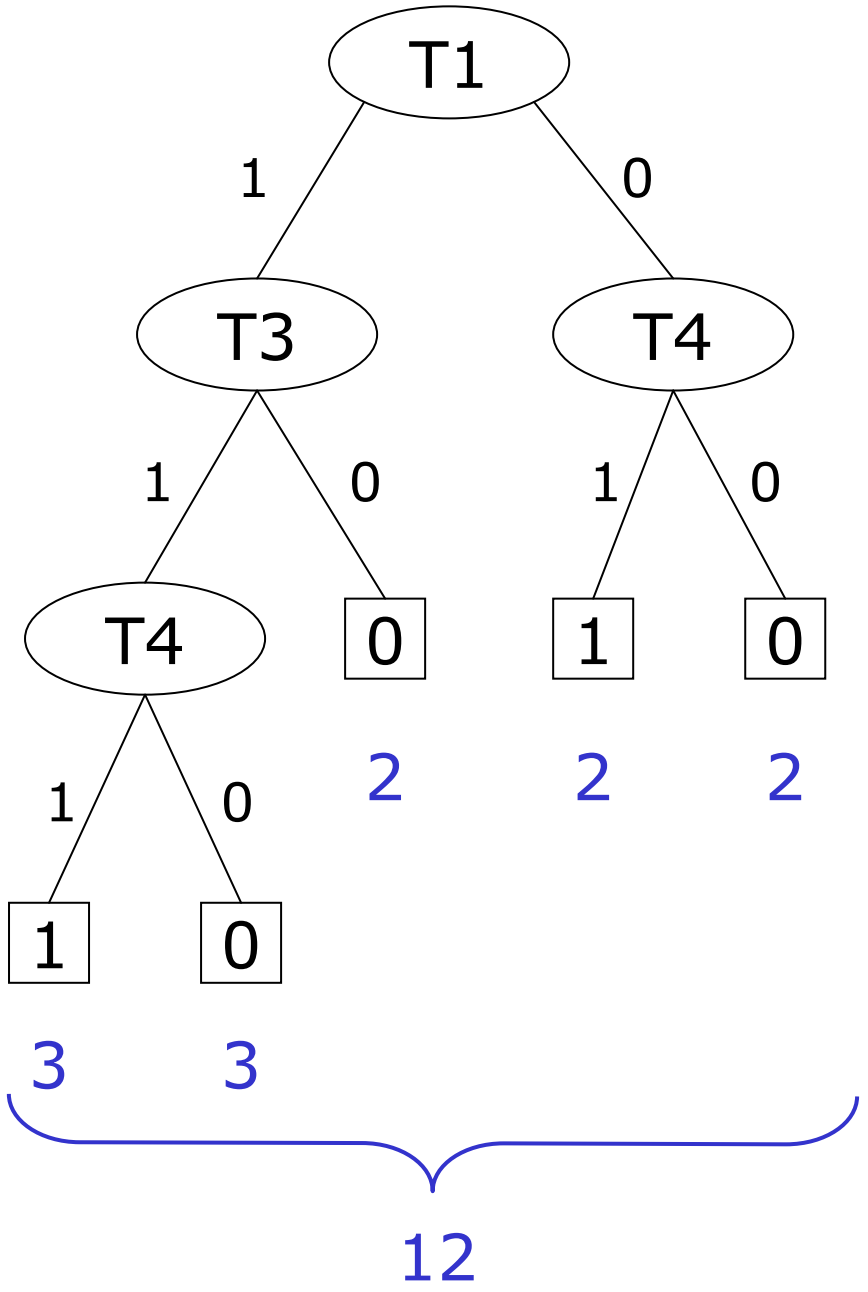
決定木の評価

- 根ノードから葉ノード x に至るテストの数 $\mathrm{cost}(x)$

- 決定木のコスト

$= \sum \{\mathrm{cost}(x) \mid x \text{ は葉ノード}\}$

- 決定木のコストは小さいほうが望ましい



決定木の最小化はNP困難

NP問題

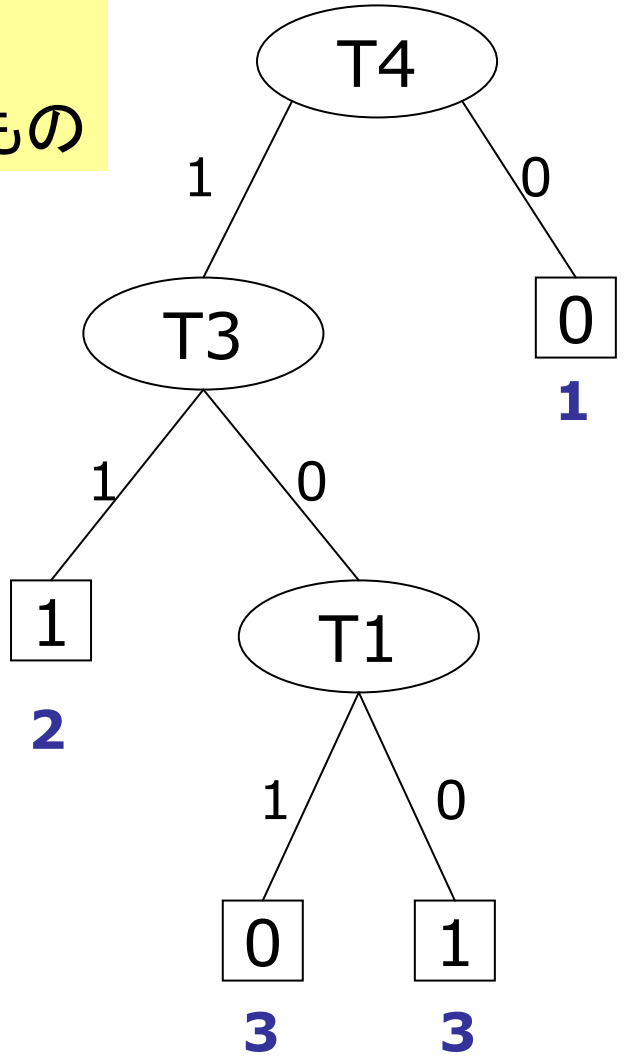
NP問題

- 正しいか否かを判定できる決定問題で、非決定性チューリングマシンが問題のサイズに関する多項式時間で計算可能な問題ある。
- 非決定性チューリングマシンとは、問題が正しいことを示す例に導いてくれる神託（Oracle, ヒント）が与えられると、多項式時間で計算してくれる抽象的計算機。ヒントに依存して非決定的に動作する。
- 実際には都合のよいヒントを与えることは難しい。すべての解の可能性を並列的に生成してみて（このステップが手に負えない計算時間となる）、正しい例をさがす。
- 「コストが閾値以下の決定木が存在するか否か」はNP問題。
  閾値以下のコストの決定木をヒントとして与えられれば、コストに比例する時間（多項式時間）で判定できるので。
- Bin Packing 問題もNP問題。
  容器への分け方をヒントとして与えてもらえば、容器のサイズに納まっているか否かは積木の数に比例する時間で計算できるので。

コストが10以下の決定木は存在するか？

T1	T2	T3	T4	目標属性
1	0	1	1	1
1	0	1	1	1
1	1	1	1	1
0	0	1	1	1
0	1	0	1	1
0	1	0	1	1
0	1	0	1	1
0	0	0	1	1
1	1	0	1	0
1	0	0	1	0
1	1	0	1	0
0	0	1	0	0
0	0	1	0	0
1	1	1	0	0
1	0	1	0	0
0	1	0	0	0

ヒントは
答えそのもの

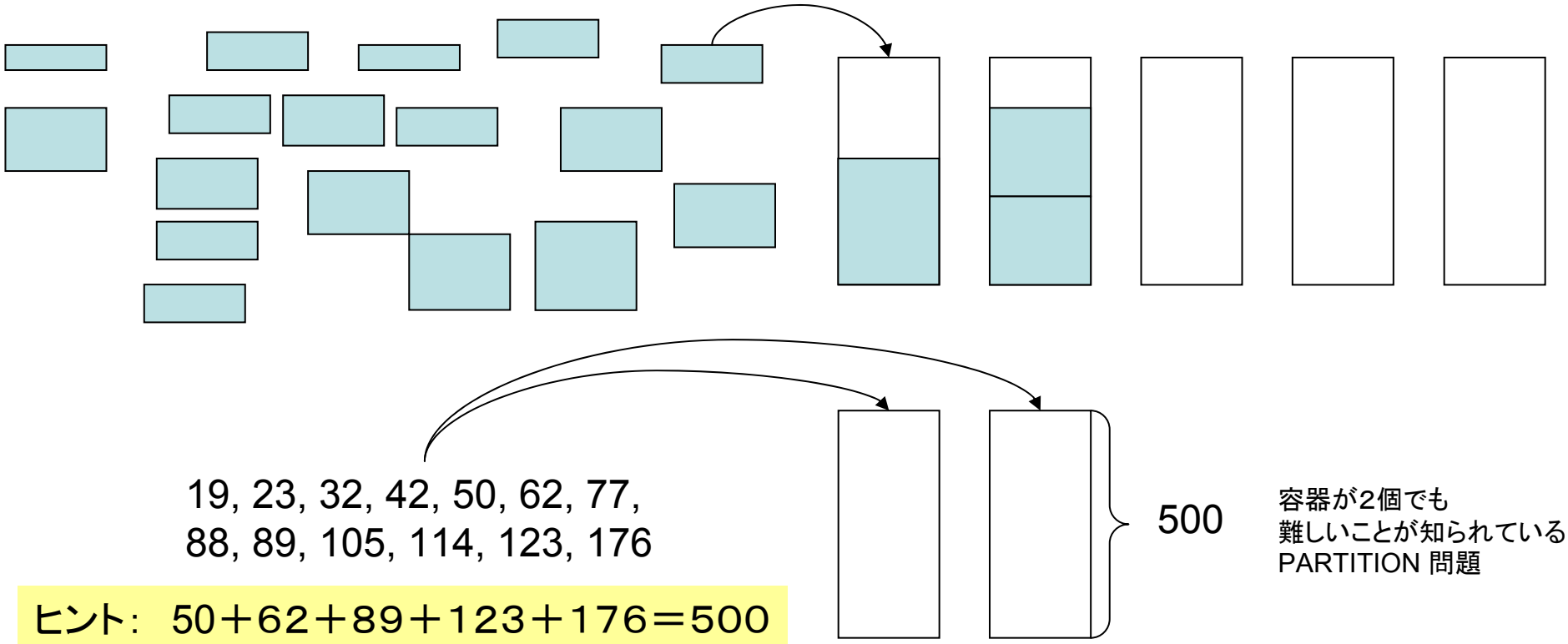


2+3+3+1=9 ＜ 10

Bin Packing 問題

- Bin Packing: 長さが正の整数 size(u) である積木 u の集合 U を k 個の部分集合 U_i (i=1,…,k)に分ける。

  各部分集合を長さが正の整数 B の容器に格納できるように分けられるか？　　$\Sigma\{\text{size}(u) \mid u \in U_i\} \leqq B$



NP完全問題

NP完全問題

- NP問題のひとつで、いかなるNP問題も多項式時間の手間をかければNP完全問題を解くことに帰着できる。
- すると、もしNP完全問題が多項式時間で解けるのであれば（しかし誰も肯定も否定もできていない計算機科学最大の問題）、どのようなNP問題も多項式時間で計算可能。
- 最初のNP完全問題は Cook により 1972年に提示された。
  すべてのNP問題が多項式時間の手間で命題論理式が充足可能か否かを判定する問題(SATと呼ぶ)へと帰着できることを示した。
  - 非決定性チューリング機械の動作をSATにより表現できるように工夫。講義では証明する時間がない。興味がある人は以下の本の17～44ページを読んでください。
    Michael R. Garey and David S. Johnson: *Computers and Intractability - A Guide to the Theory of NP-Completeness* (1979)
- SAT を多項式時間で他のNP問題 X を解くことに帰着できれば、その XもNP完全問題になる（いなかるNP問題も多項式時間で X の解決に帰着できるので）。
  このようにして SAT と同格のNP完全問題が多数認定される。
- 例　Bin Packing,「コストが閾値以下の決定木が存在するか否か」
- 多項式時間で解けそうにない問題が多いことを知る。

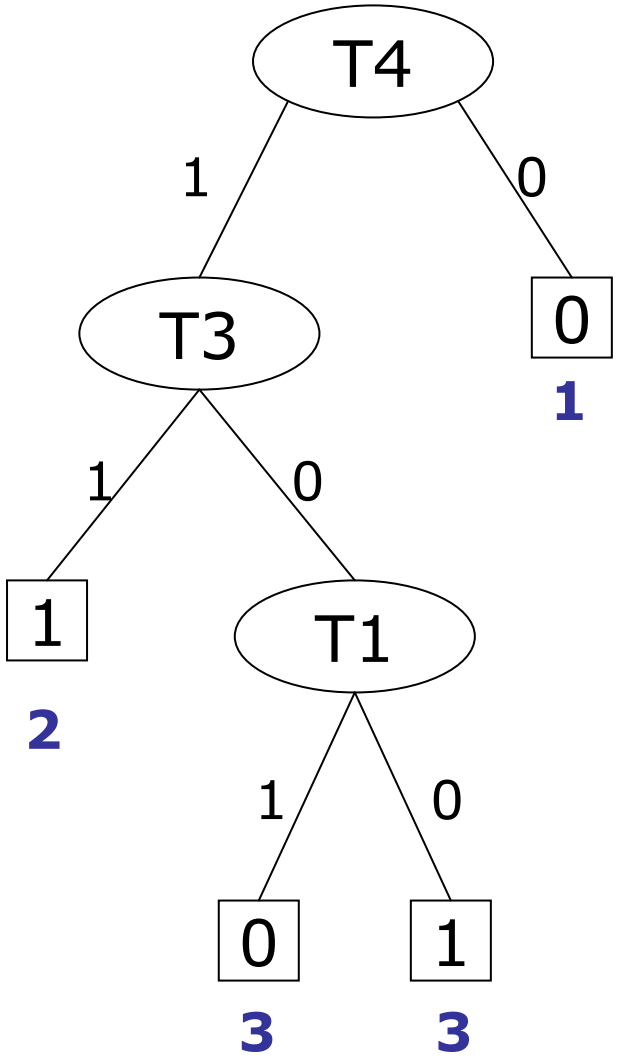
NP困難問題

NP困難問題

- NP問題のように正しいか否かを判定できる決定問題とは限らない。いかなるNP問題も多項式時間の手間をかければNP困難問題を解くことに帰着できるという性質を持つ。
- たとえば「コストが最小の決定木を求めよ」という最適化問題は、真偽を判定する決定問題でないが、NP困難問題である。
- なぜなら最小のコストが求まれば、NP完全問題「コストが閾値以下の決定木が存在するか否か」は直ぐに判定でき、任意のNP問題が多項式時間の手間をかけて、この最適化問題を解くことに帰着できるからである。
- NP完全問題とNP困難問題は混同されるので注意してほしい。特に、NP完全問題を最適化問題の形にしたときNP困難問題になる。

決定問題　　コストが10以下の決定木は存在するか？　NP完全
最適化問題　コスト最小の決定木は？　　NP困難

T1	T2	T3	T4	目標属性
1	0	1	1	1
1	0	1	1	1
1	1	1	1	1
0	0	1	1	1
0	1	0	1	1
0	1	0	1	1
0	1	0	1	1
0	0	0	1	1
1	1	0	1	0
1	0	0	1	0
1	1	0	1	0
0	0	1	0	0
0	0	1	0	0
1	1	1	0	0
1	0	1	0	0
0	1	0	0	0



「コストが閾値以下の決定木が存在するか否か」がNP完全問題であることを証明する。

L. Hyafil and R. L. Rivest: “Constructing Optimal Binary Decision Trees is NP-complete,” *Information Processing Letters*, Vol. 5, No. 1, May 1976

証明のロードマップ

SAT は、多項式時間の手間をかければ EXACT COVER by 3-SETS というNP完全問題を解くことに帰着できる。

この EXACT COVER by 3-SETS を多項式時間の変形により、「コストが閾値以下の決定木が存在するか否か」を解くことに帰着できることを示す。

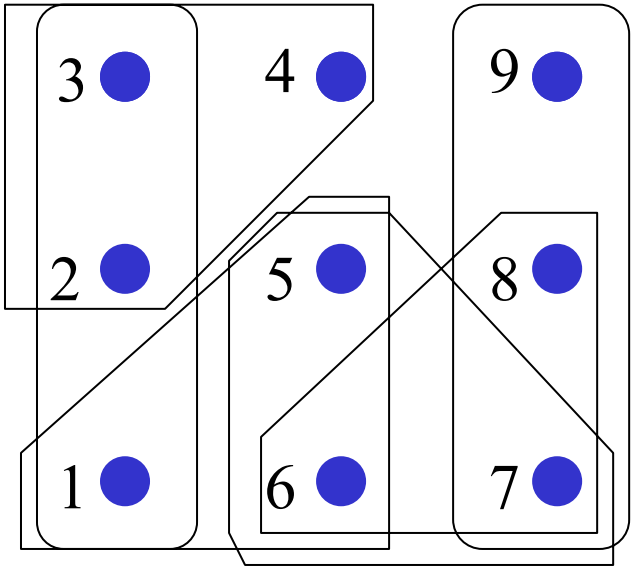
EXACT COVER BY 3-SETS 問題 （NP完全問題）

3の倍数の個数だけ元を含む有限集合 X と、
3個の元を含む X の部分集合の集まり $S = \{T1, T2, \ldots\}$ が与えられたとき、 次の条件を満たす部分集合 $S1 \subseteq S$ は存在するか？

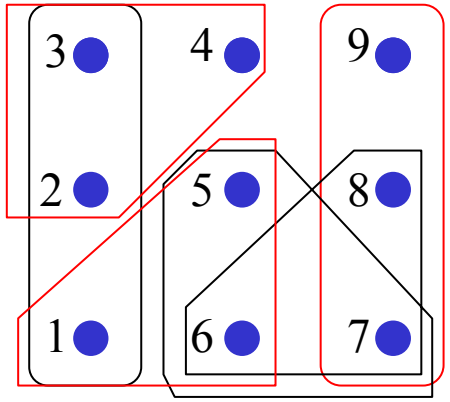
- $\cup \{T \mid T \in S1\} = X$
  X の任意の元は、$S1$ のどれか一つに含まれる
- $S1$ の元は互いに交わらない
  任意の $i \neq j$ について $Ti \cap Tj = \phi$ ）

例

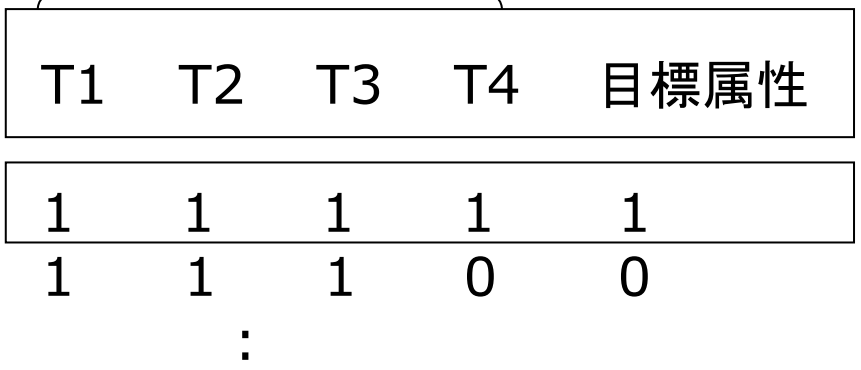
$X = \{1,2,3,4,5,6,7,8,9\}$
$S = \{\{1,2,3\}, \{2,3,4\}, \{1,5,6\}, \{5,6,7\}, \{6,7,8\}, \{7,8,9\}\}$



ヒント



テスト属性



属性(attribute) フィーチャー

レコード　データ　タップル

レコード t の属性 A の値を $t[A]$ と表記

目標値　　目標属性の値から一つ選択.　　例えば 1

目標属性が A、目標値が 1 のとき、

- $t[A]= 1$ となるレコード t を　正(positive)
- $t[A]\neq 1$ となるレコード t を　負(negative)

EXACT COVER BY 3-SETS

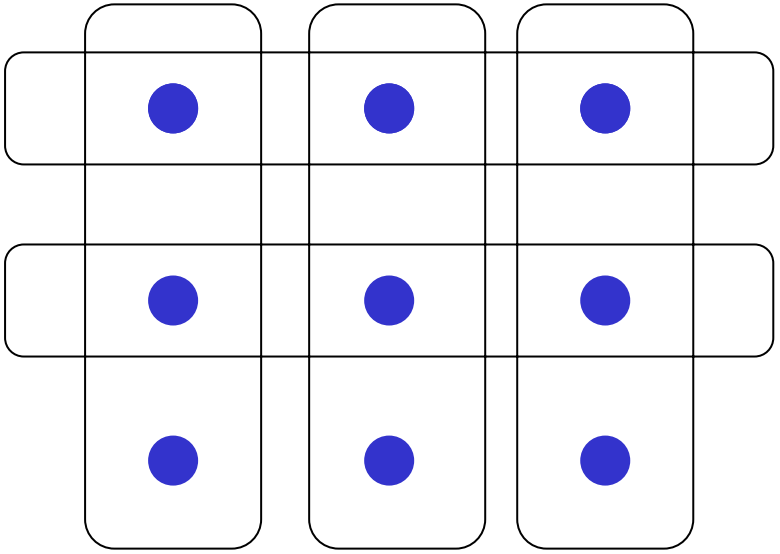
$X,\ S = \{T1, T2, \ldots\}$

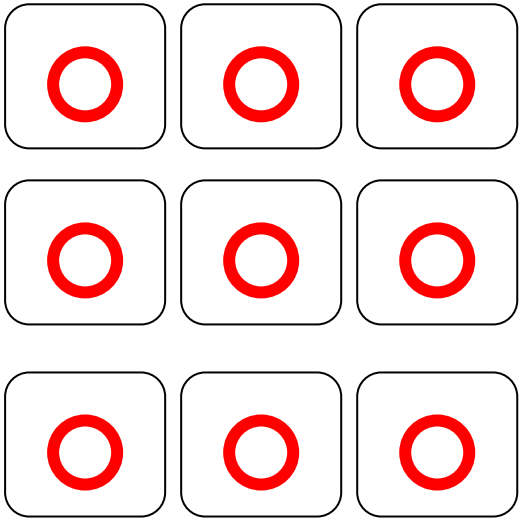
正レコード　Xの元

テスト属性

負レコード　Yの元

決定木を作成するために
負のレコードを追加する





EXACT COVER BY 3-SETS

X, $S = \{T1, T2, \ldots\}$

正レコード: X の元

負レコード: $|X|$ 個の元を生成

$Y=\{y1, y2, \ldots\}$ とおく

$t[A]= 1 \quad t \in X$ 正レコード

$\quad = 0 \quad t \in Y$ 負レコード

$t \in X \cup Y$ に対して

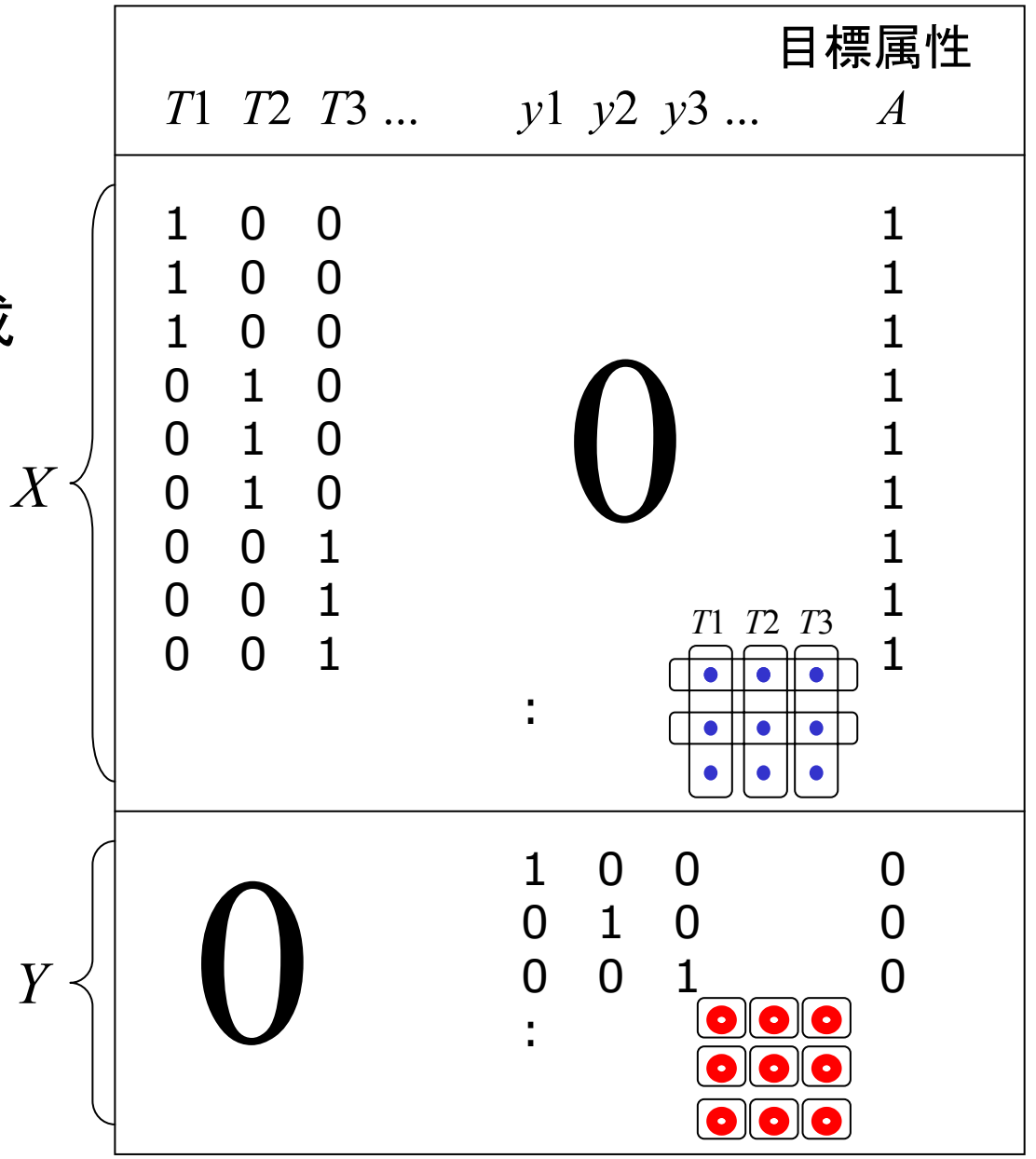
$S \cup Y$ の元はテスト属性

$t[Ti] \quad = 1 \quad t \in Ti$

$\quad = 0 \quad$ otherwise

$t[y] \quad = 1 \quad t = y \ (y \in Y)$

$\quad = 0 \quad$ otherwise



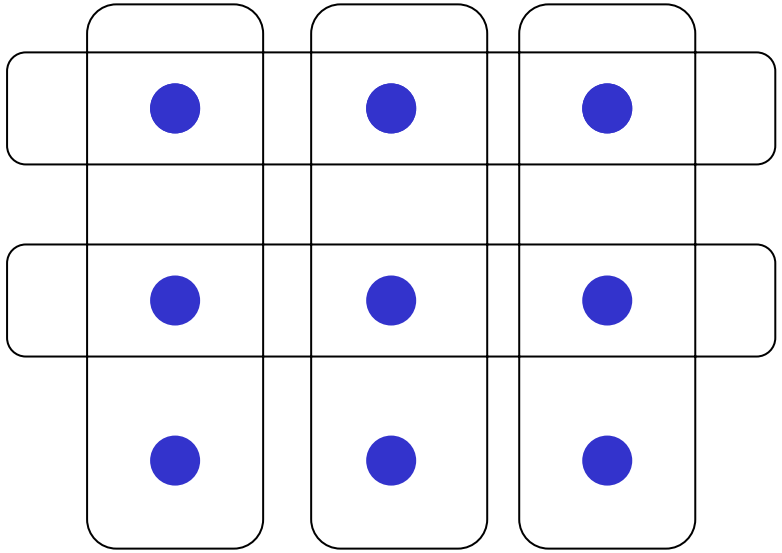
多項式時間の手間で変換可能

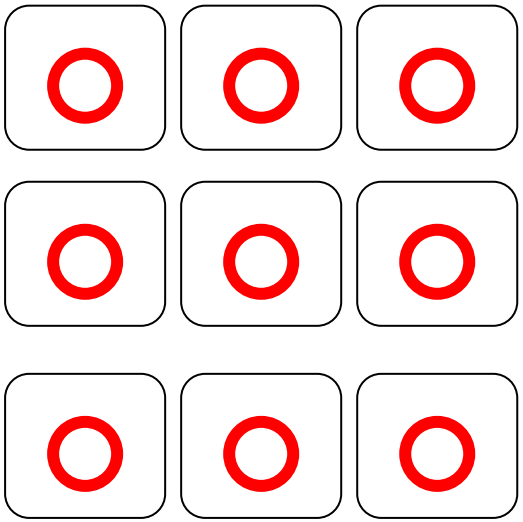
例

正レコード　$t[A]=1$　Xの元

負レコード　$t[A]=0$　Yの元

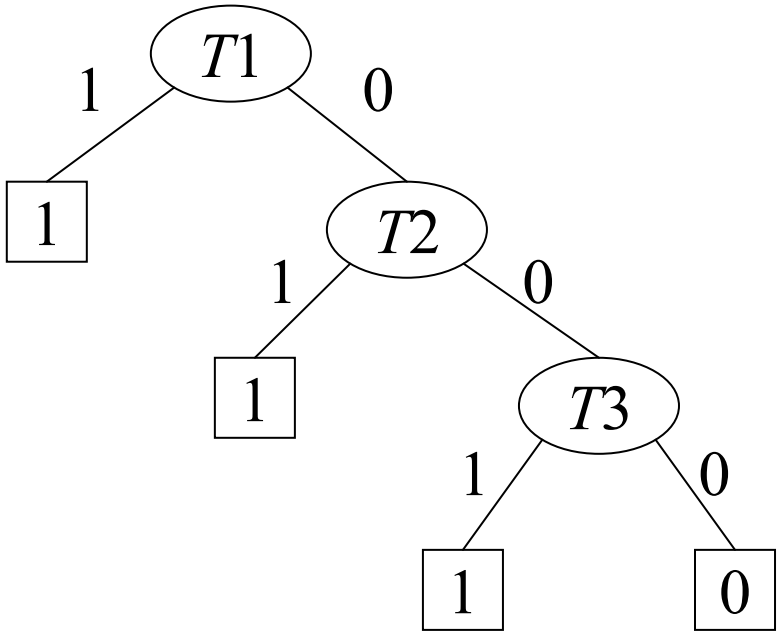
テスト属性

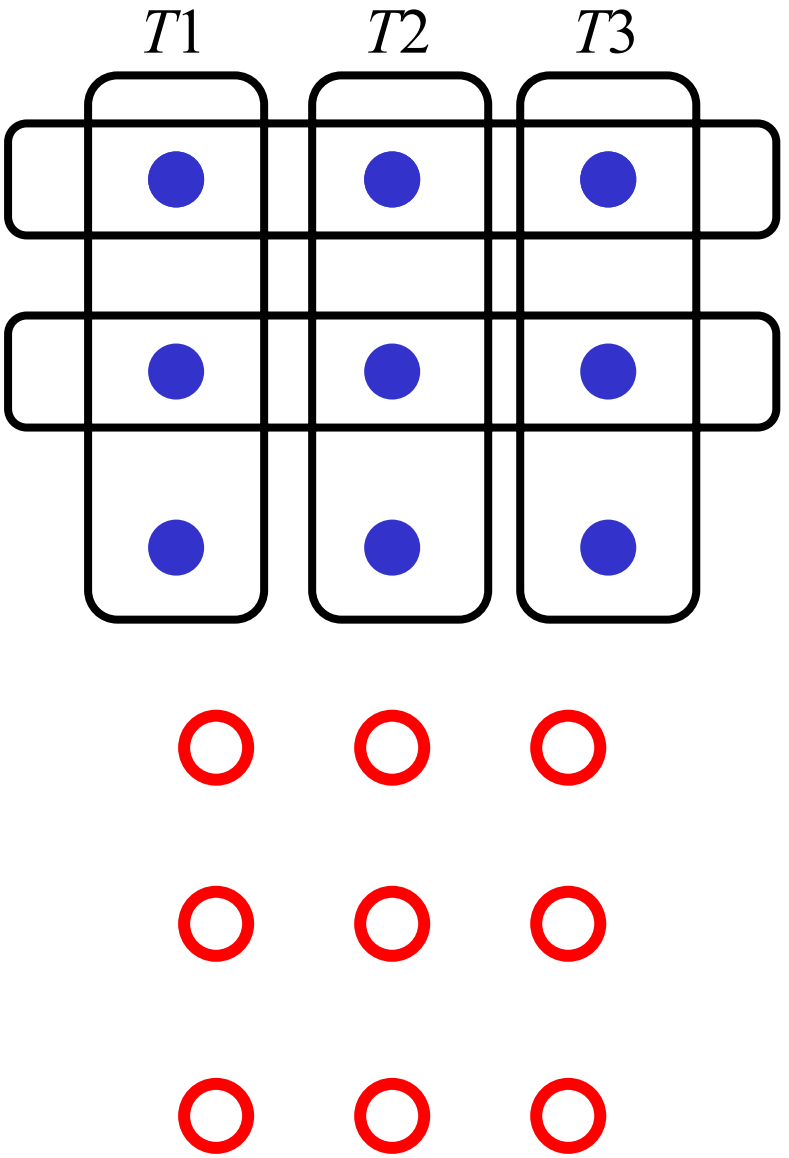




正レコードと負レコードを
決定木で分離する戦略1

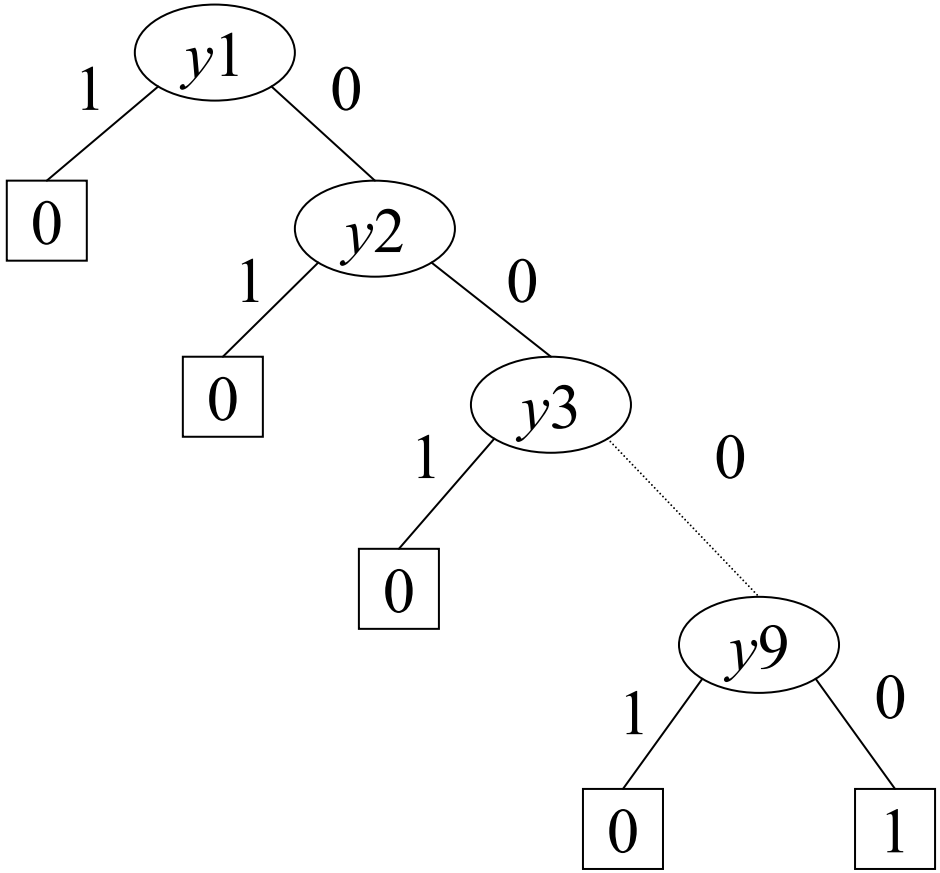
$S = \{T1, T2, \ldots\}$ のテスト属性
を使って正レコードを包含する

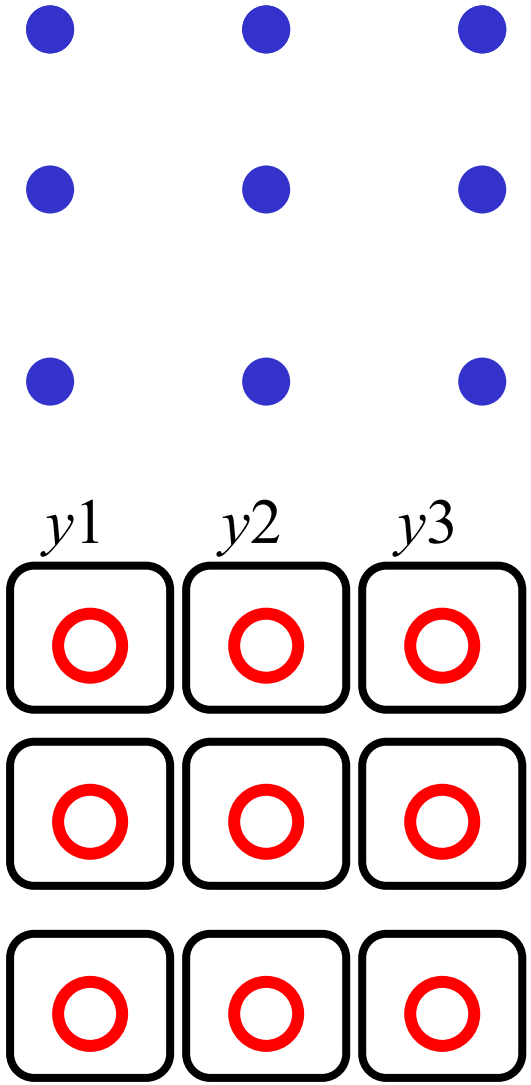




正レコードと負レコードを
決定木で分離する戦略2

$Y = \{y1, y2, \ldots\}$ のテスト属性を
すべて使って負レコードを包含する



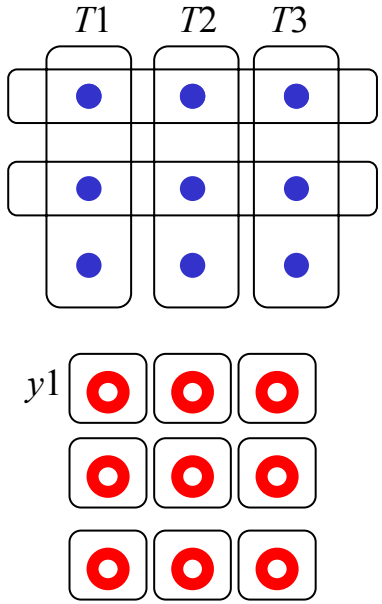


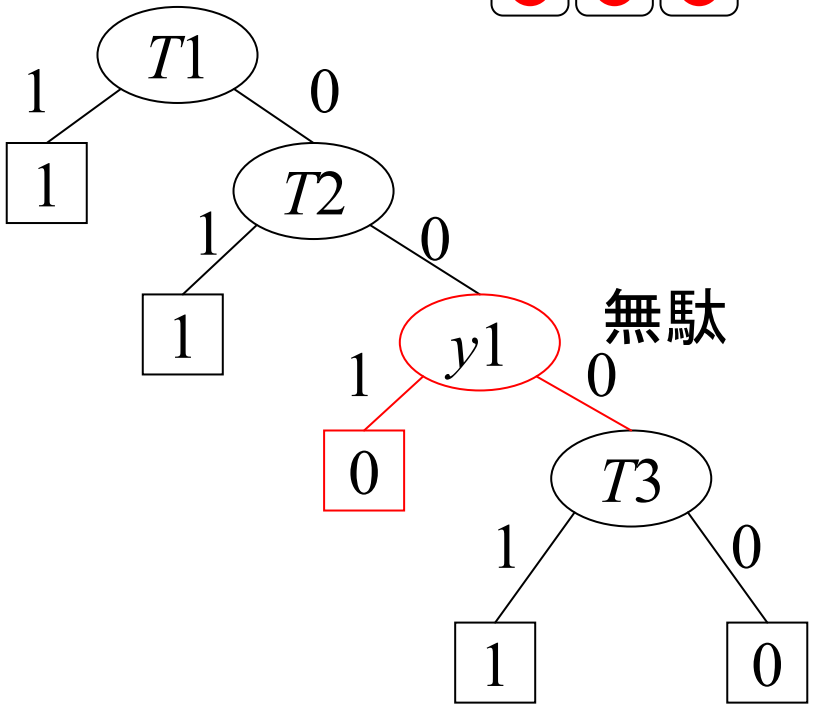
正レコードと負レコードを
分離する決定木の構成法

戦略１
正レコードをすべて包含する
S の部分集合をテスト属性として使う
（Y のテスト属性を使うのは無駄）

戦略２
負レコードを包含する Y の
テスト属性をすべて使う
（S のテスト属性を使うのは無駄）

したがって、戦略１もしくは戦略２が
コスト最小の木を生成する

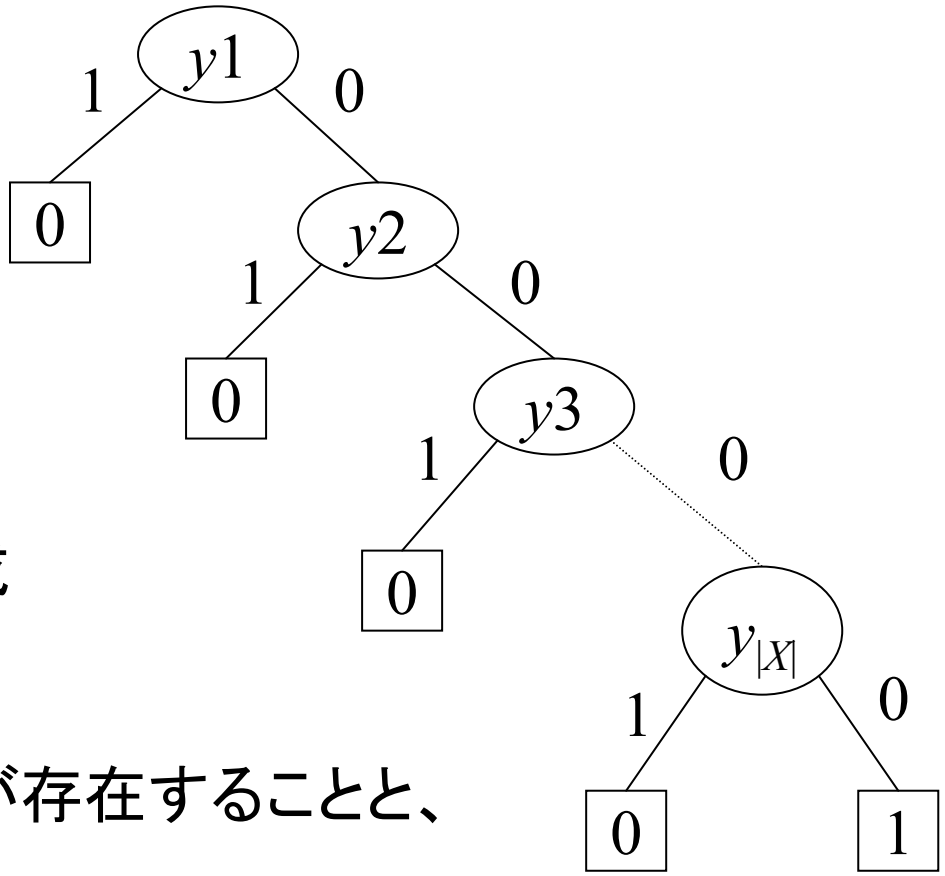




戦略２のコスト

$1+2+\ldots+|X|+|X|$

は戦略１のコストより大きい



戦略１がコスト最小の木を生成

EXACT COVER BY 3-SETSが存在することと、
戦略１よりコストが

$1+2+\ldots+|X|/3+|X|/3$　(= w とおく)

となる決定木が存在すること
(コストが w 以下の決定木が存在すること) は同値.

したがって、EXACT COVER by 3-SETS を多項式時間の変形により、「コストが閾値以下の決定木が存在するか否か」を解くことに帰着できる （証明終）

近似的解法

Greedy Algorithm

エントロピー

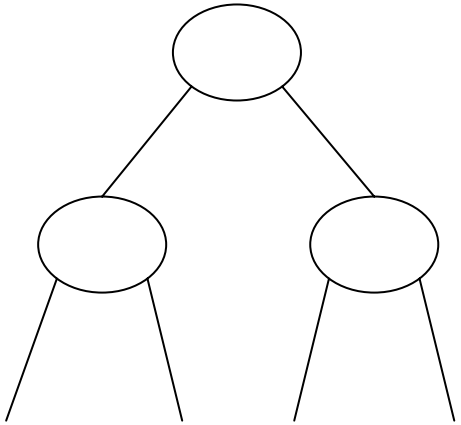
近似的解法　　貪欲(greedy)アルゴリズム

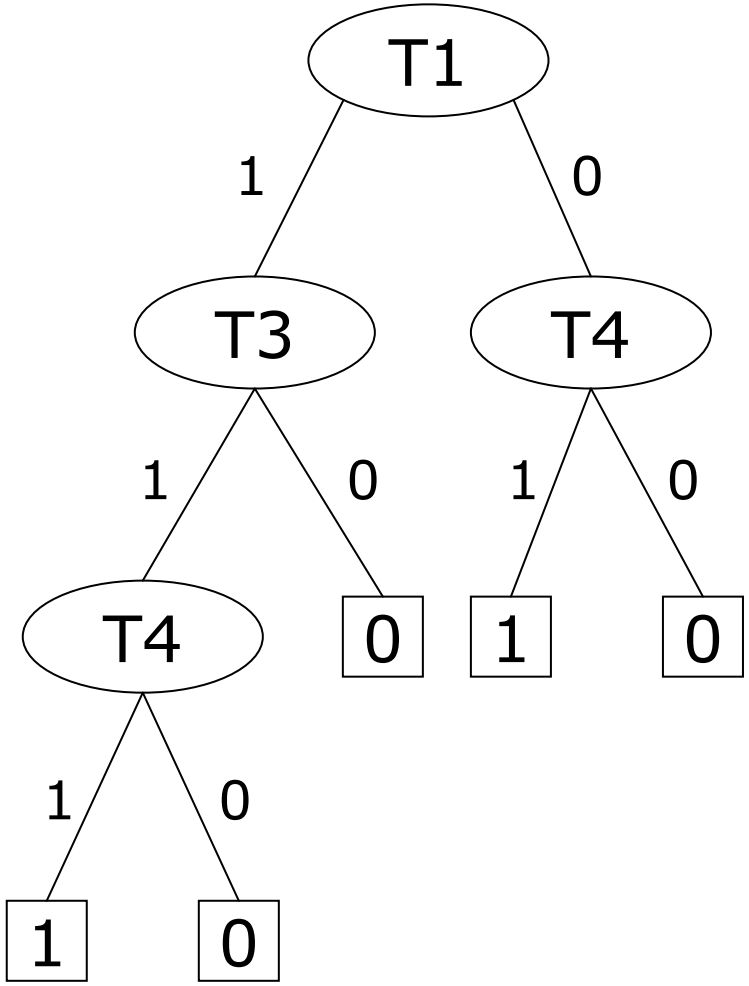
- 「最も良さそうな」テストを選択し、
  レコード集合を分割

- 分割したレコード集合に
  同じ戦略を繰り返し適用する

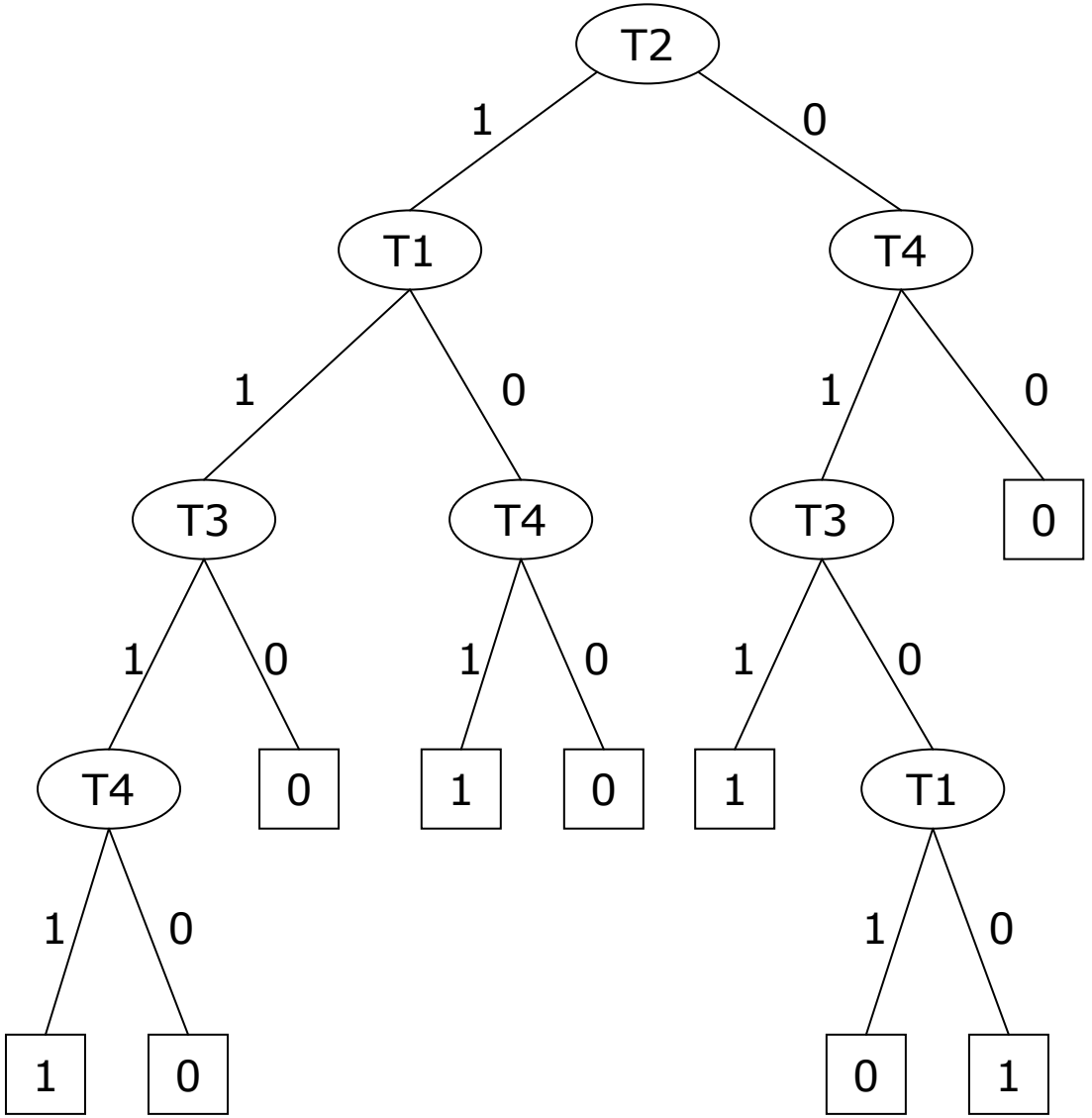
- 一度選択したテストは変更しない

- 「最も良さそうな」 の基準を工夫





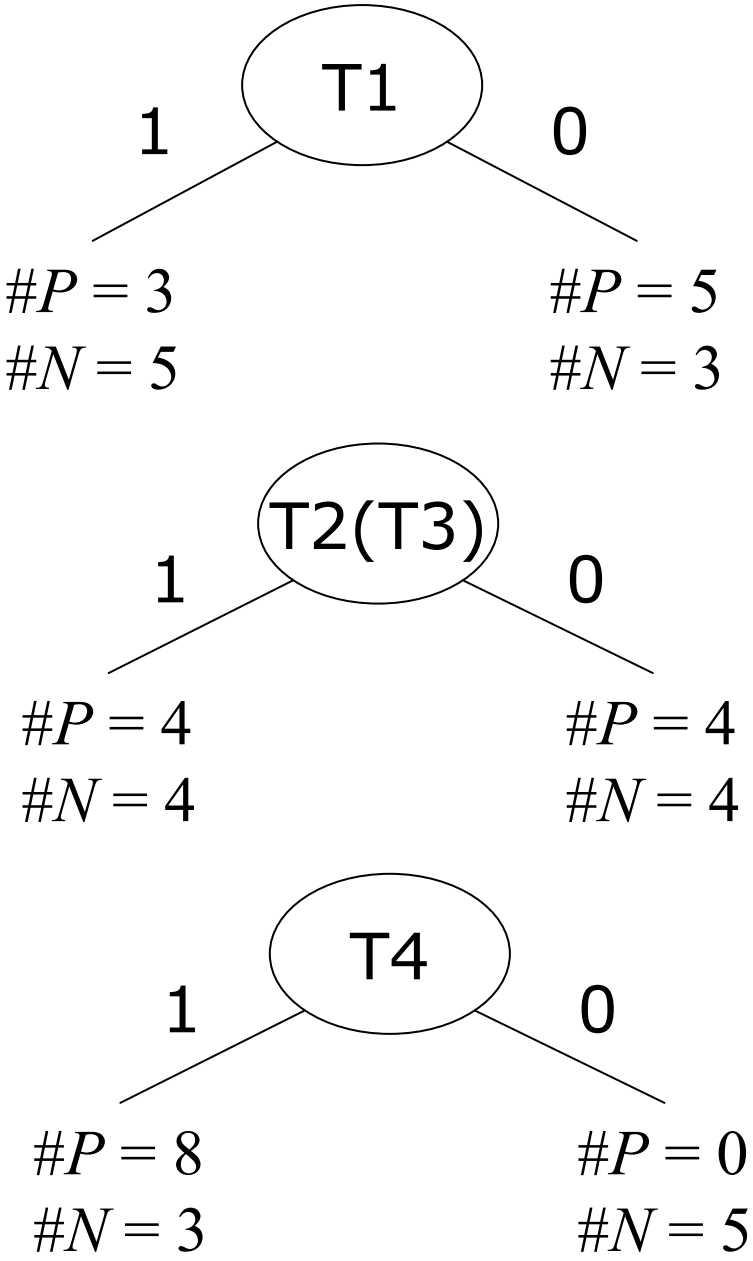
3+3+2+2+2=12

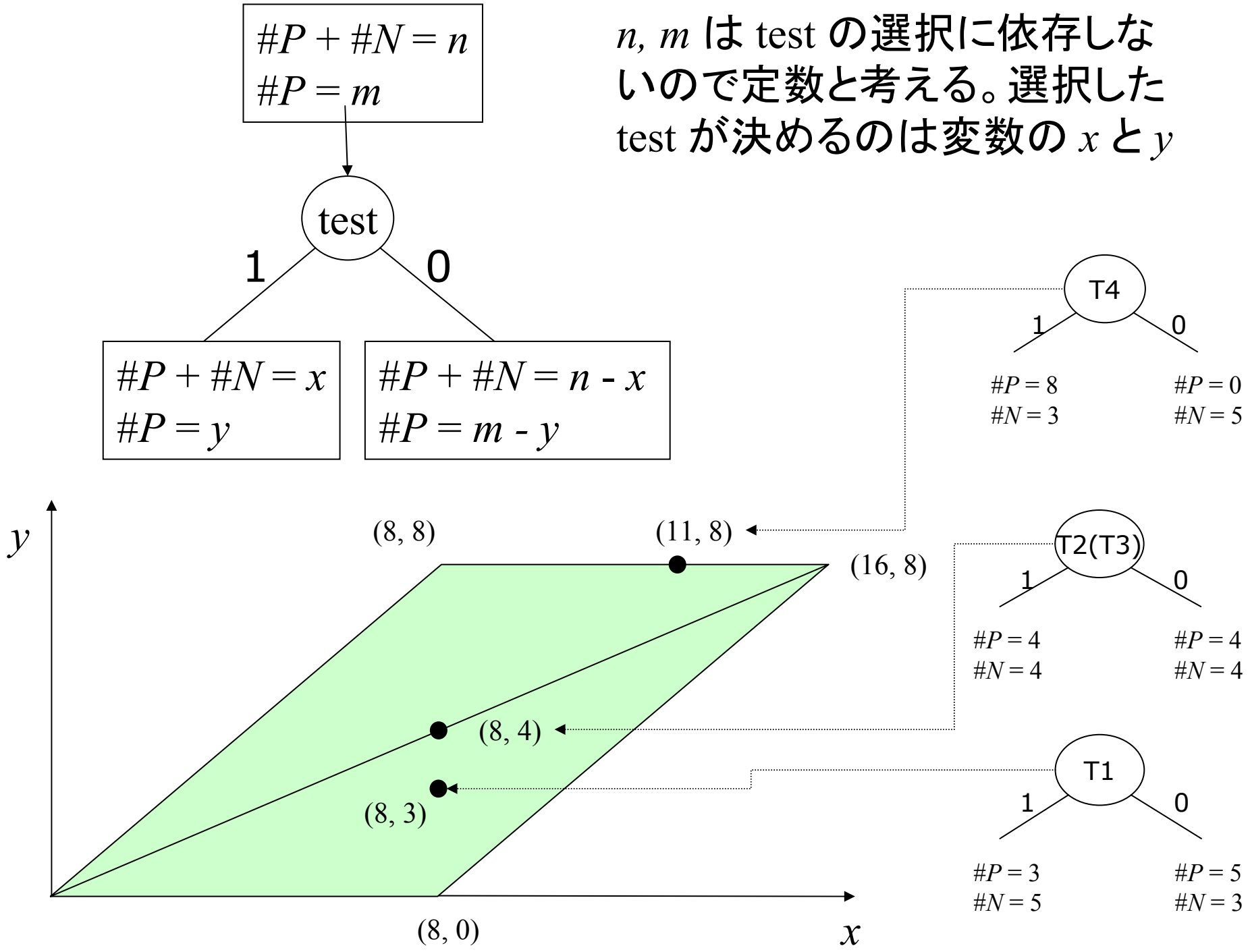


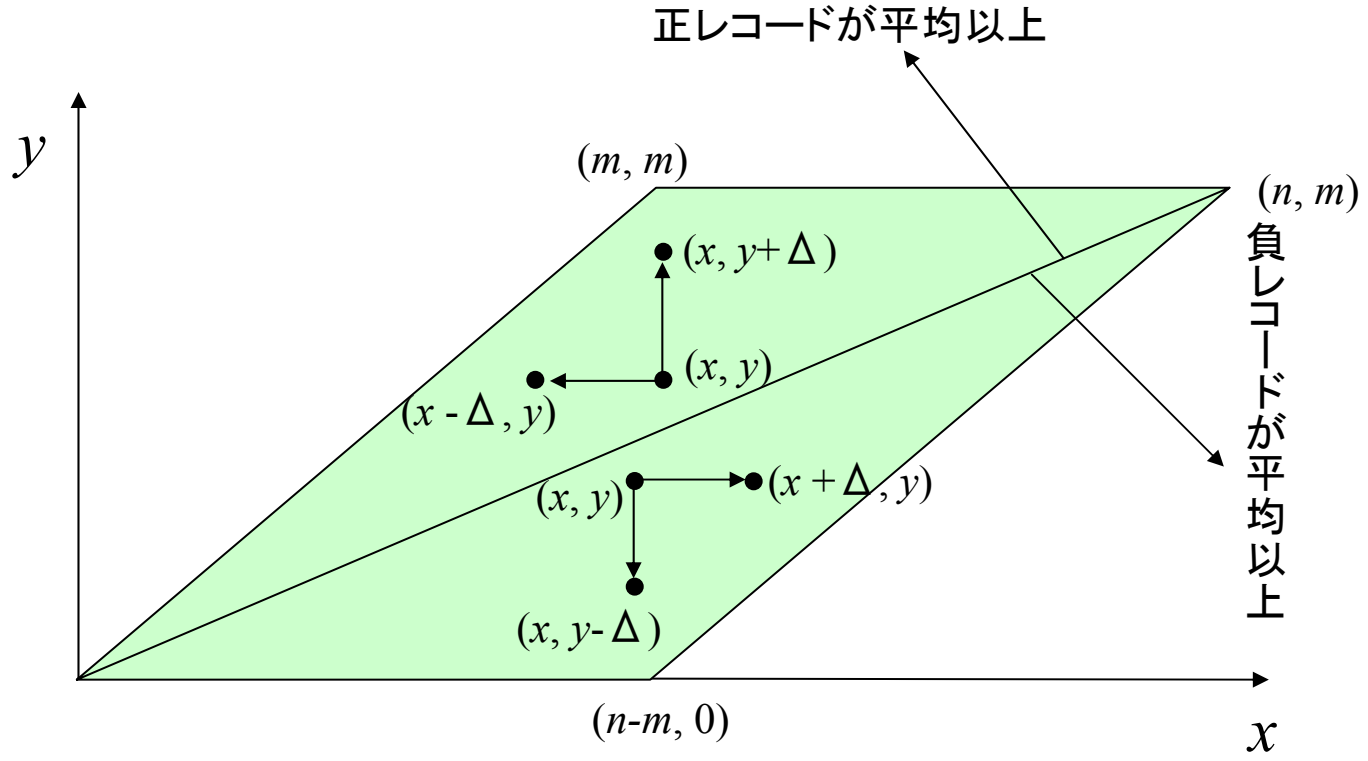
4+4+3+3+3+3+4+4+2=30

$\#P$: 正レコードの数　$\#N$: 負レコードの数

T1	T2	T3	T4	目標属性
1	0	1	1	1
1	0	1	1	1
1	1	1	1	1
1	1	1	0	0
1	0	1	0	0
1	1	0	1	0
1	0	0	1	0
1	1	0	1	0
0	1	0	1	1
0	0	1	1	1
0	1	0	1	1
0	1	0	1	1
0	0	0	1	1
0	0	1	0	0
0	1	0	0	0
0	0	1	0	0




#P + #N = n
#P = m
test
1
0
#P + #N = x
#P = y
#P + #N = n - x
#P = m - y
n, m は test の選択に依存しないので定数と考える。選択した test が決めるのは変数の x と y
T4
1
0
#P = 8
#N = 3
#P = 0
#N = 5
T2(T3)
1
0
#P = 4
#N = 4
#P = 4
#N = 4
T1
1
0
#P = 3
#N = 5
#P = 5
#N = 3
y
(8, 8)
(11, 8)
(16, 8)
(8, 4)
(8, 3)
(8, 0)
x



評価基準 $\phi(x, y)$ が満たすべき条件

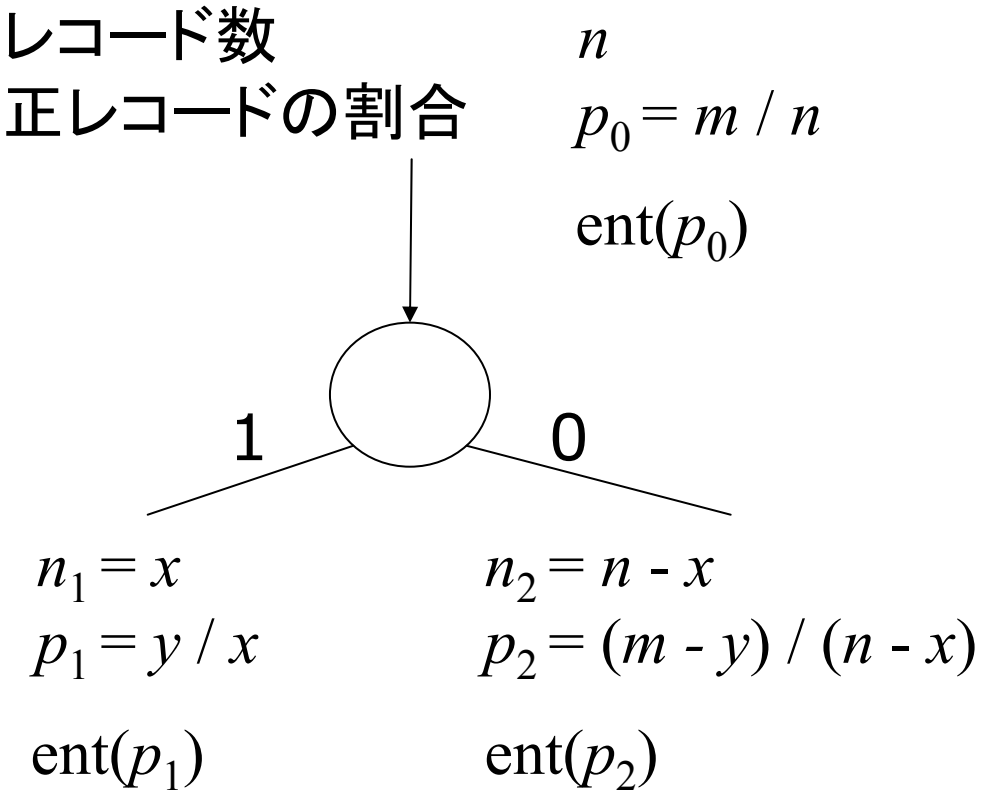
$\phi(x, y)$ は $m/n = y/x$ のとき最小

$\phi(x, y) \leqq \phi(x, y+\Delta)$, $\phi(x, y) \leqq \phi(x-\Delta, y)$ if $m/n < y/x$

$\phi(x, y) \leqq \phi(x, y-\Delta)$, $\phi(x, y) \leqq \phi(x+\Delta, y)$ if $m/n > y/x$

$\Delta > 0$

正レコードの割合	$p = \#P / (\#P + \#N)$
負レコードの割合	$1 - p$
エントロピー	$\mathrm{ent}(p) = - p \log_2 p - (1-p) \log_2 (1-p)$

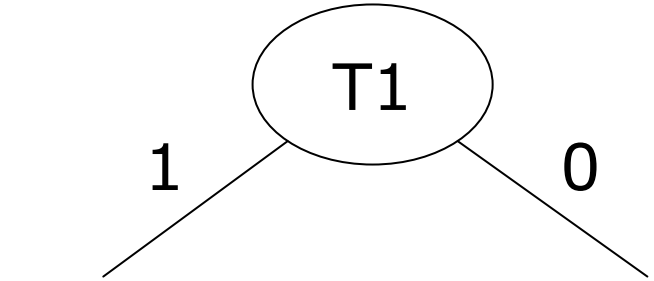


Entropy Gain / Information Gain

$$\mathrm{Ent}(x, y) = \mathrm{ent}(p_0) - (n_1/n)\,\mathrm{ent}(p_1) - (n_2/n)\,\mathrm{ent}(p_2)$$

$\#P = 8$
$\#N = 8$

ent(8/16)
$= 2\ (-\ (1/2)\ \log_2(1/2))$
$= 1$



$\#P = 3$
$\#N = 5$

$\#P = 5$
$\#N = 3$

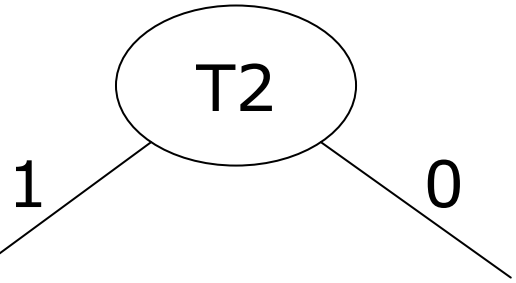
ent(3/8)
$= -\ (3/8)\log_2(3/8) - (5/8)\log_2(5/8)$
$= 0.95444$

ent(5/8) = ent(3/8)

Entropy Gain $=$ ent(8/16) - (8/16)ent(3/8) - (8/16)ent(5/8)
$= 1 - 0.95444$
$= 0.04556$

$\#P = 8$
$\#N = 8$

ent(8/16) = 1



$\#P = 4$
$\#N = 4$

ent(4/8) = 1

$\#P = 4$
$\#N = 4$

ent(4/8) = 1

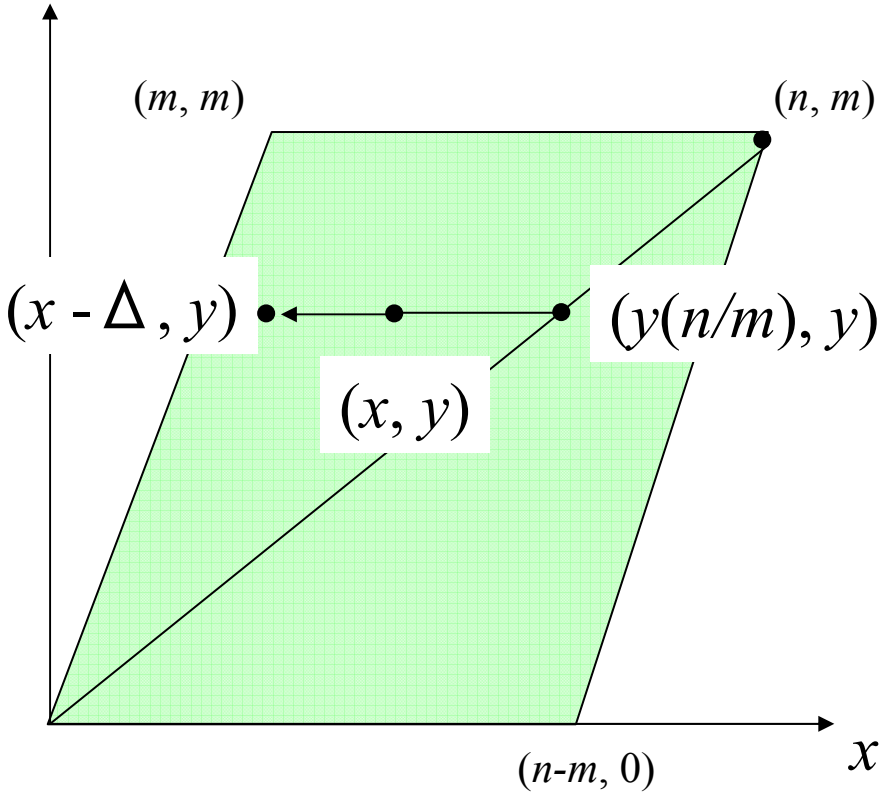
Entropy Gain = ent(8/16) - (8/16)ent(4/8) - (8/16)ent(4/8)
= 0

- Ent(x, y) は $m/n = y/x$ のとき最小
- Ent(x, y) は凸関数　任意の点 **v1**, **v2** と $0 \leqq \lambda \leqq 1$について

$$\mathrm{Ent}(\lambda \mathbf{v1}+(1-\lambda)\mathbf{v2}) \leqq \lambda\,\mathrm{Ent}(\mathbf{v1}) + (1-\lambda)\,\mathrm{Ent}(\mathbf{v2})$$

すると

$$\mathrm{Ent}(\lambda \mathbf{v1}+(1-\lambda)\mathbf{v2}) \leqq \max\{\mathrm{Ent}(\mathbf{v1}), \mathrm{Ent}(\mathbf{v2})\}$$



$\mathrm{Ent}(x,y)$

$\leqq \max\{\mathrm{Ent}(x-\Delta, y), \underline{\mathrm{Ent}(y(n/m), y)}\}$　最小値

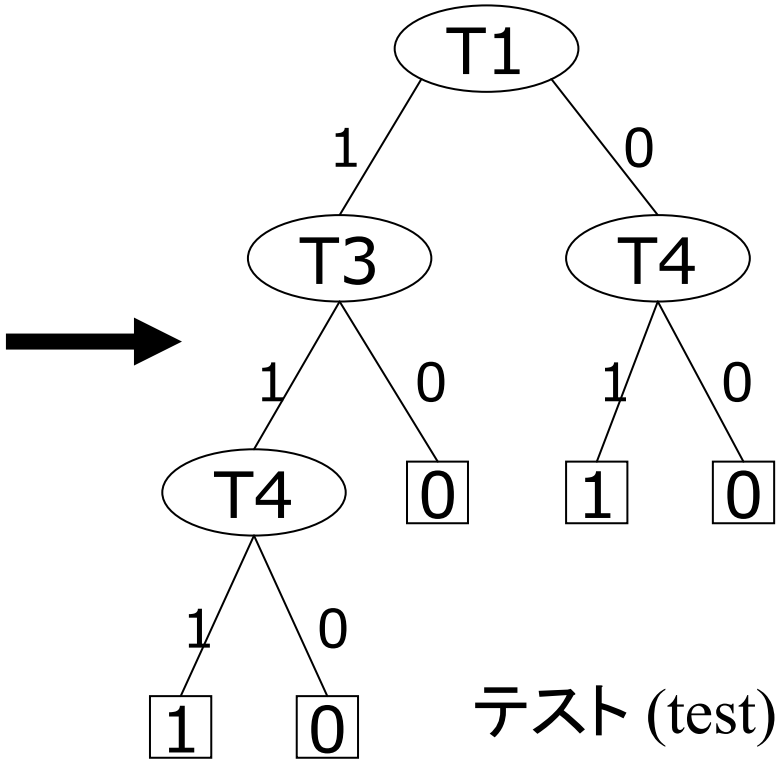
$\leqq \mathrm{Ent}(x-\Delta, y)$

巨大な決定木と Overfitting 問題

訓練(training)レコード

T1	T2	T3	T4	目標属性
1	0	1	1	1
1	0	1	1	1
1	1	1	1	1
1	1	1	0	0
1	0	1	0	0
1	1	0	1	0
1	0	0	1	0
1	1	0	1	0
0	1	0	1	1
0	0	1	1	1
0	1	0	1	1
0	1	0	1	1
0	0	0	1	1
0	0	1	0	0
0	1	0	0	0
0	0	1	0	0

未知のテストレコードに対する正答率



テスト (test) レコード

T1	T2	T3	T4	目標属性	予測
1	1	1	1	1	1
1	0	0	1	0	0
1	1	0	1	1	0
0	1	1	1	1	1
0	0	0	0	0	0

正答率
= 4/5

Overfitting 問題

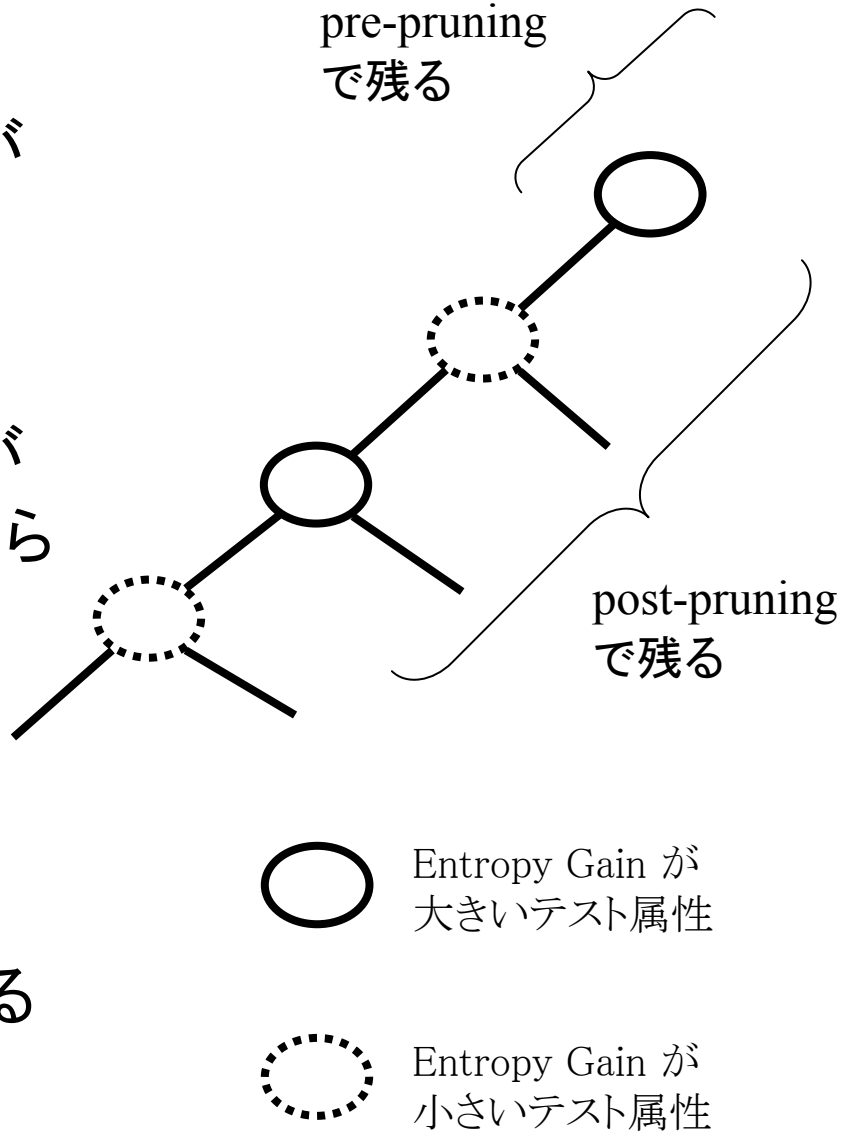
- 決定木を十分に大きくすれば、
  訓練レコードに関しては正答率を改善できる
- しかし巨大な木のテストレコードに対する正答率は
  必ずしも高くない
- 訓練レコードに対して過剰に適合 (overfitting)した状態

Overfitting の回避方法1

- pre-pruning 法
  決定木を構成中に Entropy Gain が
  小さいテスト属性は生成しない

- post pruning 法
  決定木を構成後に Entropy Gain が
  小さいテスト属性を木の葉ノードから
  ボトムアップに除く

- 実験的には Overfitting を
  ある程度回避できると言われている



このような例題を作ってみてください

Overfitting の回避方法2
小さい決定木を生成するアプローチ
Entropy Gain を最大化するテストを効率的に計算できるか？

- 論理積でレコード集合を分割
  $(T1 = 1) \wedge (T2 = 1) \wedge \ldots \wedge (Tn = 1)$
  NP困難　branch-and-bound 法

- テスト属性 T の領域に3つ以上の異なる離散値がある場合
  例　血液型 $\in$ {A, O}　（$\subseteq$　{A, B, AB, O}）
  k 個の離散値のとき O(k)、つまり k に線形比例する計算時間

- テスト属性が数値の場合
  例　21 ≦年齢 ≦ 44
  多項式時間

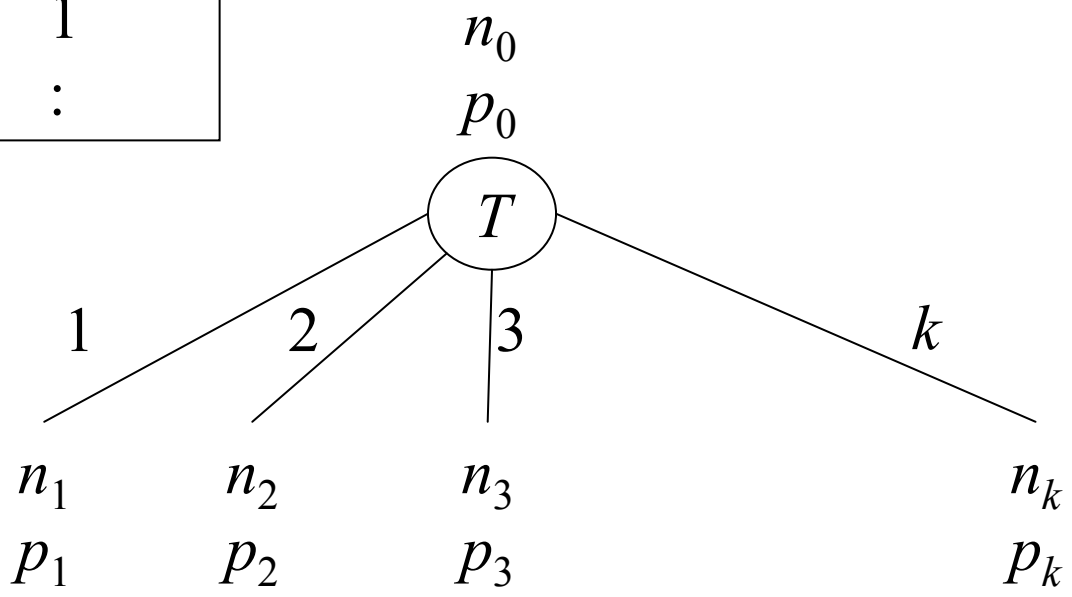
テスト属性 T の領域に3つ以上の異なる離散値がある場合

- T の領域を $\{1, 2, \ldots, k\}$

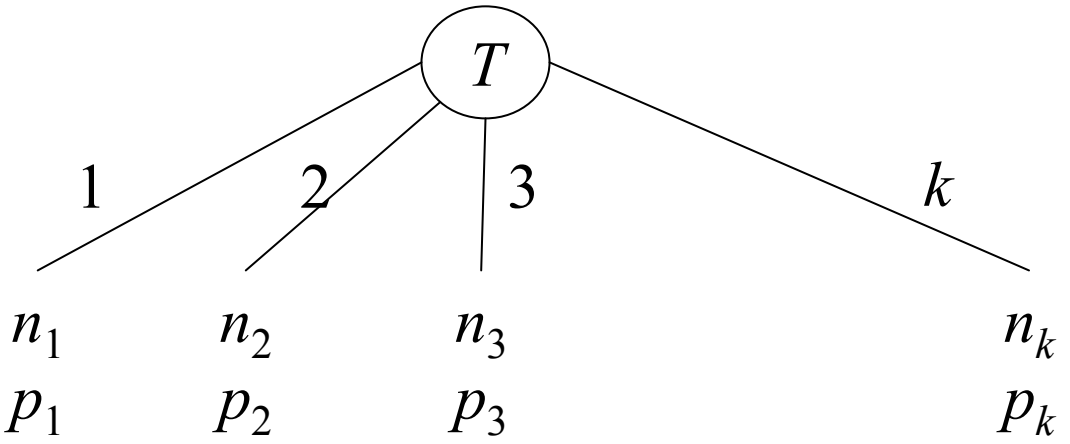
T ...	目標属性A
2	1
3	0
1	1
:	:

<u>各属性値ごとに分割する方式</u>

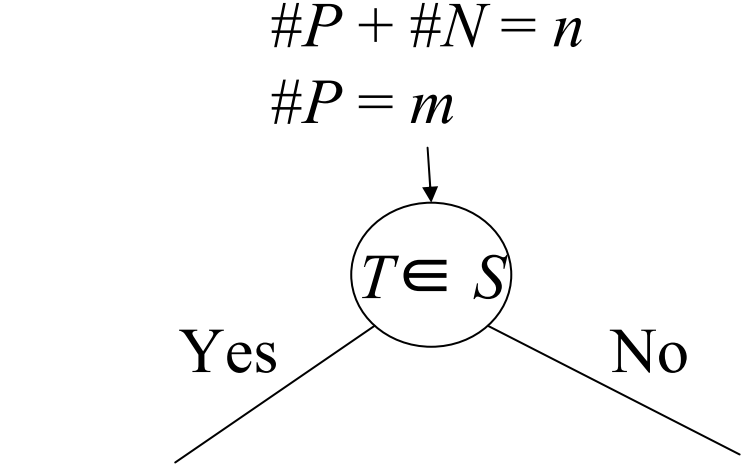
例えば k=1000 のときは
巨大な木になる



$$\mathrm{ent}(p_0) - \sum \left(n_i / n_0\right) \mathrm{ent}(p_i)$$



一般性を失うことなく
$p_1 \geqq p_2 \geqq \ldots \geqq p_k$
と仮定



データ数
$\Sigma \{n_i \mid i \in S\} = x$
$\Sigma \{n_i \mid i \notin S\}$

正のデータ数
$\Sigma \{p_i n_i \mid i \in S\} = y$
$\{p_i n_i \mid i \notin S\}$

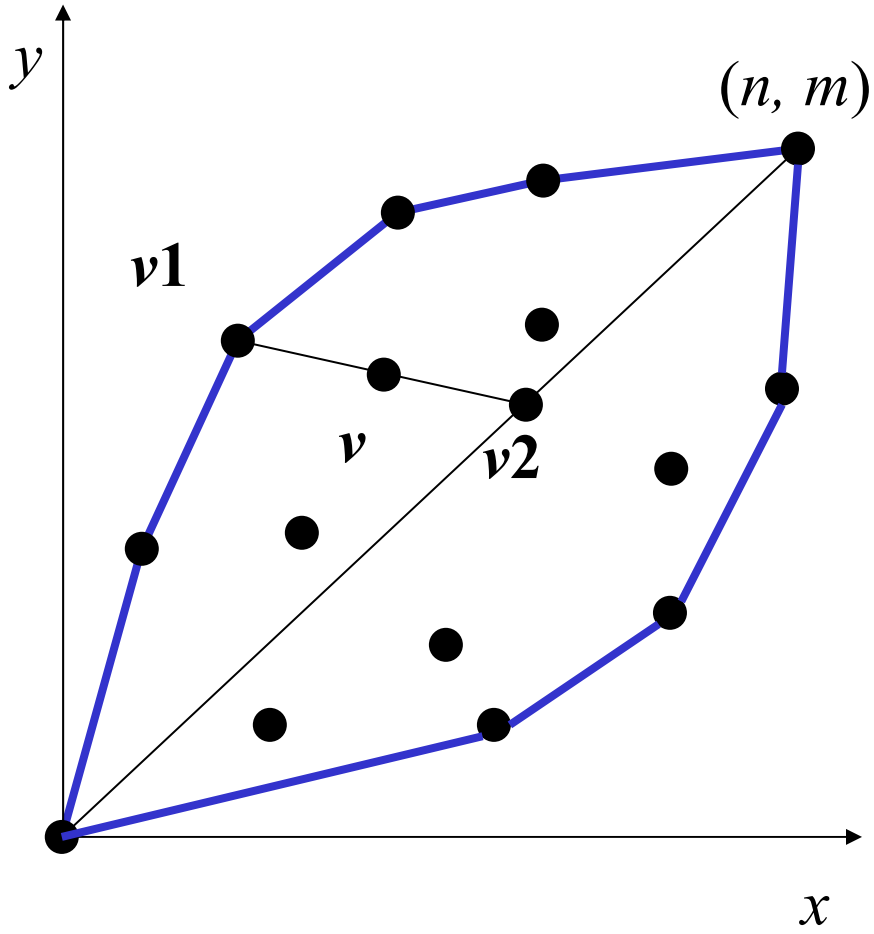
<u>属性値の部分集合で分割する方式</u>

$\{1, 2, \ldots, k\}$ の部分集合 S

定理
ある j が存在して
$S = \{1, \ldots, j\}$ または $\{j, \ldots, k\}$
が Entropy Gain を最大化

各テスト $T \in S$ に２次元座標の点(x, y)（スタンプ点と呼ぶ）を対応させる

凸包 (Convex Hull)：　すべてのスタンプ点を含む最小の凸多角形
（凸多角形 P: P の任意の２点を結ぶ直線が P 内を通過）

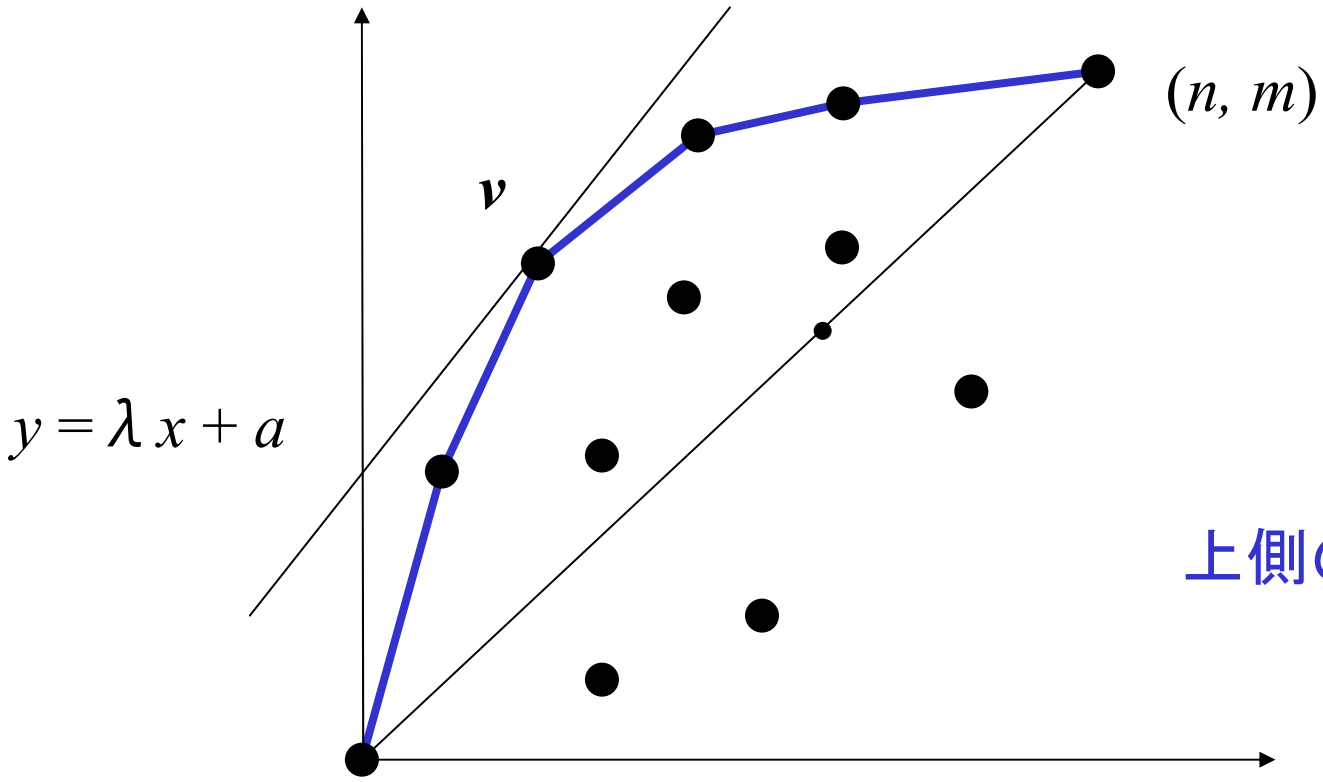


凸包内の任意のスタンプ点 v に対して、v を内分点として持つ凸包上のスタンプ点 $v1$ と対角線上の点 $v2$ が存在する（演習問題）。

$\mathrm{Ent}(v) \leqq \max\{\mathrm{Ent}(v1), \mathrm{Ent}(v2)\}$

$\mathrm{Ent}(v2)$ は最小値なので
$\mathrm{Ent}(v) \leqq \mathrm{Ent}(v1)$

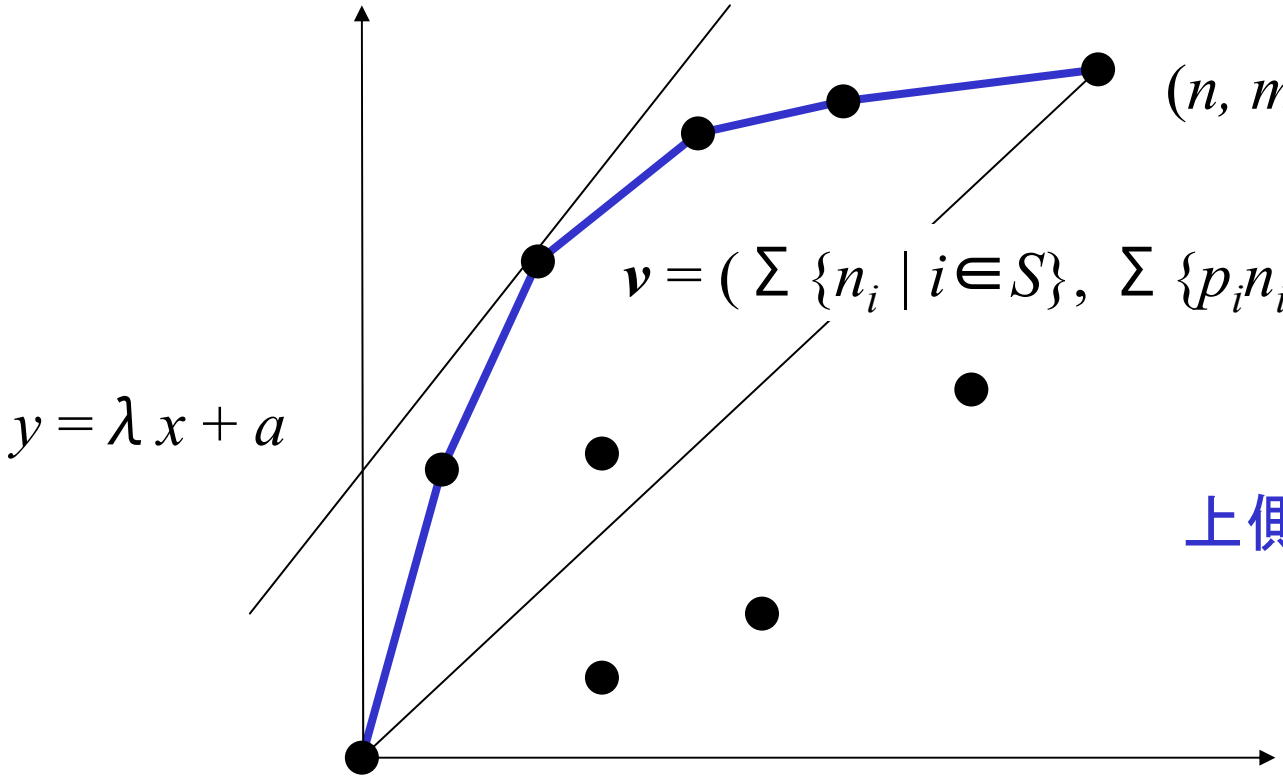
Entropy Gain を最大化するスタンプ点が凸包上に存在



上側の凸包の場合

上側の凸包上のスタンプ点 v に対して、ある傾き λ の直線が存在し、
全点中で v は y 切片 a を最大化

下側の凸包上のスタンプ点 v に対しては、ある傾き λ の直線が存在し、
全点中で v は y 切片 a を最小化



上側の凸包の場合

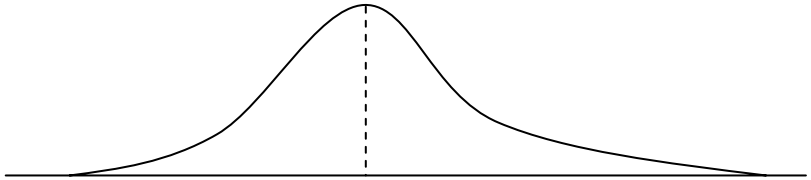
$$
\begin{aligned}
a &= y - \lambda x \\
&= \Sigma\{p_i n_i \mid i \in S\} - \lambda \Sigma\{n_i \mid i \in S\} \\
&= \Sigma\{(p_i - \lambda) n_i \mid i \in S\}
\end{aligned}
$$

$p_1 \geqq p_2 \geqq \ldots \geqq p_k$ であるので
最大化するのは $p_j - \lambda \geqq 0$ と $p_{j+1} - \lambda < 0$ を満たす $S = \{1, \ldots, j\}$
ただし $p_1 - \lambda < 0$ のときは $S = \phi$, $p_k - \lambda > 0$ のときは $S = \{1, \ldots, k\}$.

目標属性が数値の場合の拡張

遺伝子の状態

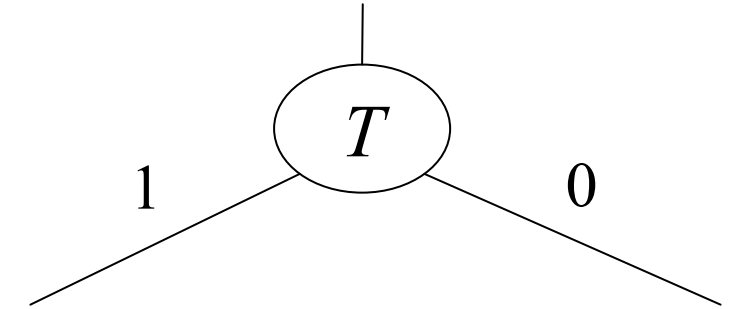
T_1	T_2 …	目標属性 A（血圧）
1	0	125
0	1	145
1	1	113
:	: :	



レコード数 $= n$

目標属性値の総和 $\Sigma\, t[A] = m$

平均値 $= m / n = p_0$



目標属性の例としては
癌細胞を1/10 にするのに
必要な放射線量や
抗がん剤の量など

レコード数 $= x$

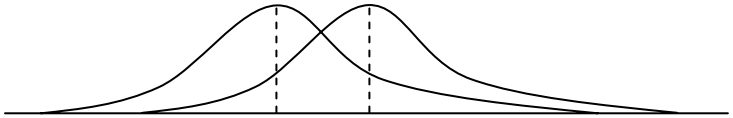
$\Sigma\, \{t[A] \mid t[T]=1\} = y$

平均値 $= y / x = p_1$

レコード数 $= n - x$

平均値 $= (m-y) / (n-x)$

$= p_2$



クラス間分散　$\mathrm{Var}(x, y) = x\,(p_1 - p_0)^2 + (n-x)\,(p_2 - p_0)^2$

クラス間分散を最大化する T_1 をみつけたい

- Var(x, y) は $m/n = y/x$ のとき最小
- Var(x, y) は凸関数　任意の点 **v1**, **v2** と $0 \leqq \lambda \leqq 1$について

$$\mathrm{Var}(\lambda \mathbf{v1} + (1-\lambda)\mathbf{v2}) \leqq \lambda \mathrm{Var}(\mathbf{v1}) + (1-\lambda)\mathrm{Var}(\mathbf{v2})$$

- Entropy Gain を最大化する方法がクラス間分散を最大化するのに使える
- 決定木のように、レコード分割を繰り返した木構造を回帰木 (Regression Tree) と呼ぶ