クラス分類問題
Classification Problem

教師つき学習
Supervised Learning

クラス分類など知識発見ツールの精度を競うコンテスト

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)

知識発見ツール（製品、フリーソフト）の充実.

ACM SIGKDD 主催のコンテスト KDD (Knowledge Discovery and Data-mining) Cup. 1997年より開催.

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.

2001年は遺伝子解析データが出題.

KDD Cup 2001 の話題から

Task 1

訓練データ

- 1909 個の既知の化合物
- 各化合物の3次元構造の特徴に関する139,351個の属性
- 目標属性　thrombinに結合するか否か？　42個が結合する

テストデータ

- 636個の新しい化合物

Task 2 & 3

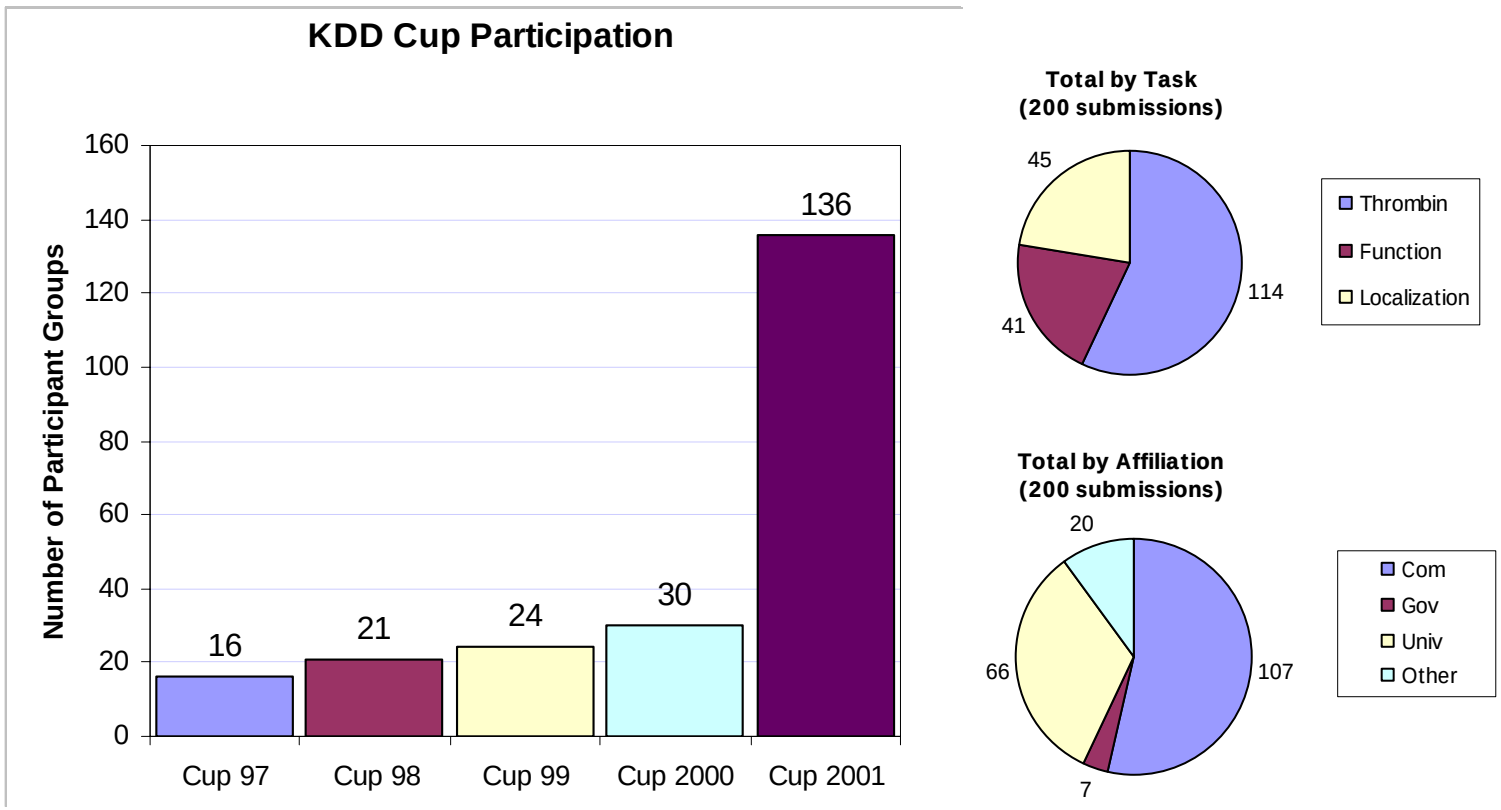
訓練データ

- 862個の遺伝子データ
- 6種類の属性(50%の値が欠損)、遺伝子間の相互作用の表
- 目標属性　遺伝子の機能(Task2) 14機能
  遺伝子の細胞内局在(Task3) 15部位

テストデータ

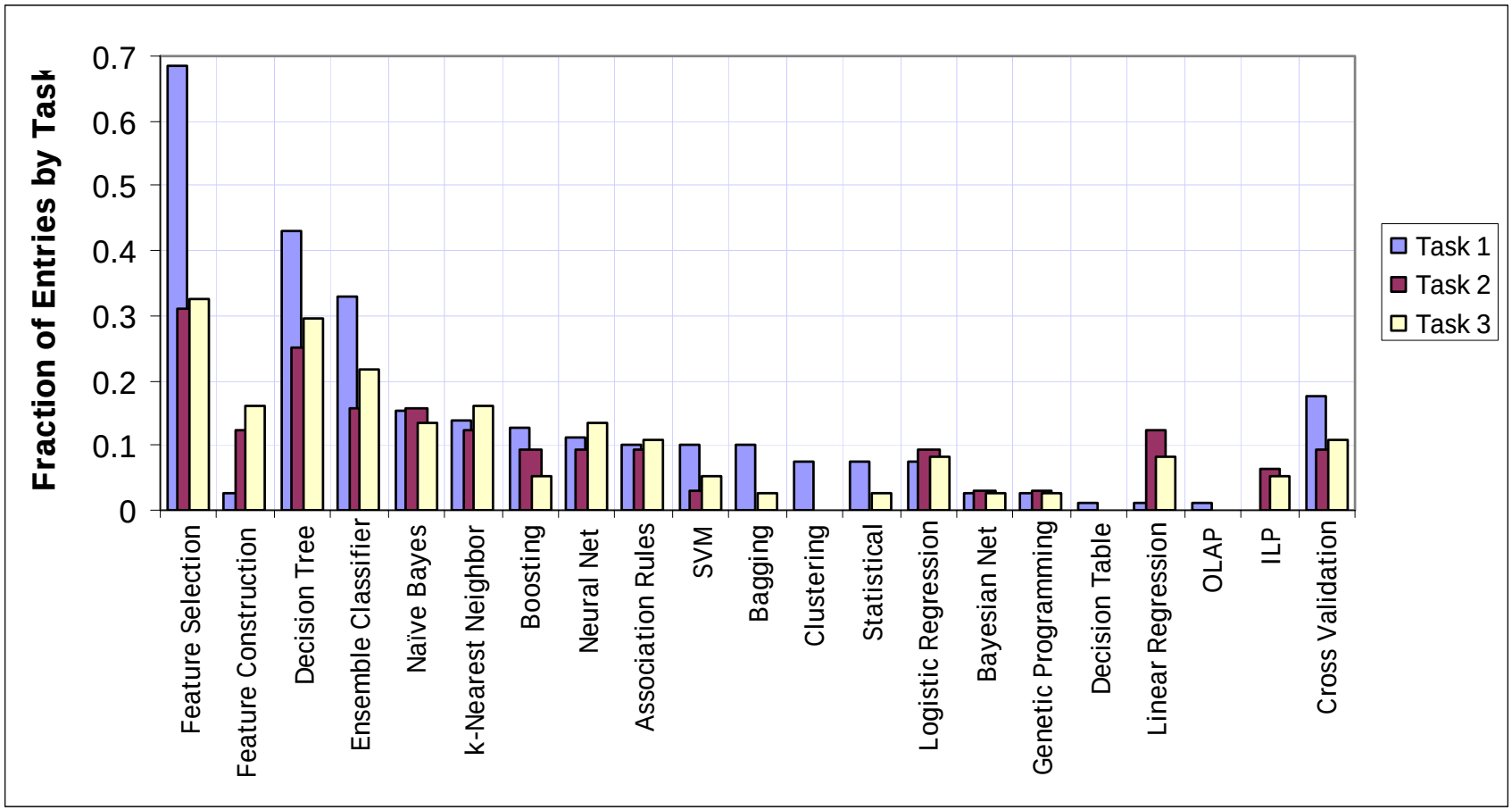
- 381個の遺伝子

Statistics: Participation



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.

Statistics: Algorithms

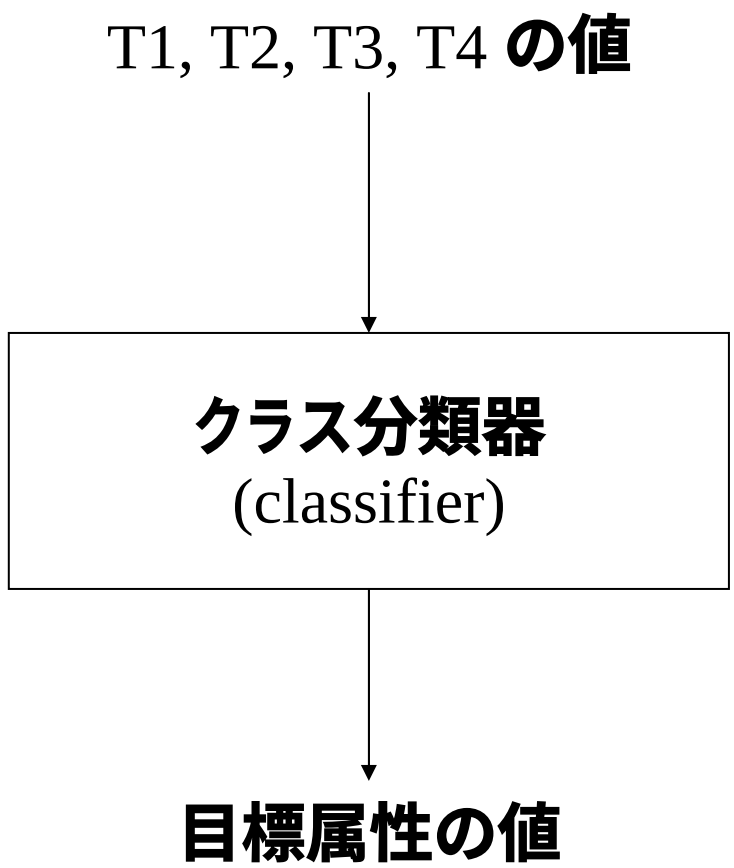


出展 Jie Cheng, Christos Hatzis, Hisashi Hayashi, Mark-A. Krogel, Shinichi Morishita , David Page, and Jun Sese: KDD Cup 2001 Report. *ACM SIGKDD Explorations, Volume* 3, Issue 2, Janurary 2002, 47-64.

クラス分類器 (Classifier) の様々な手法

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

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



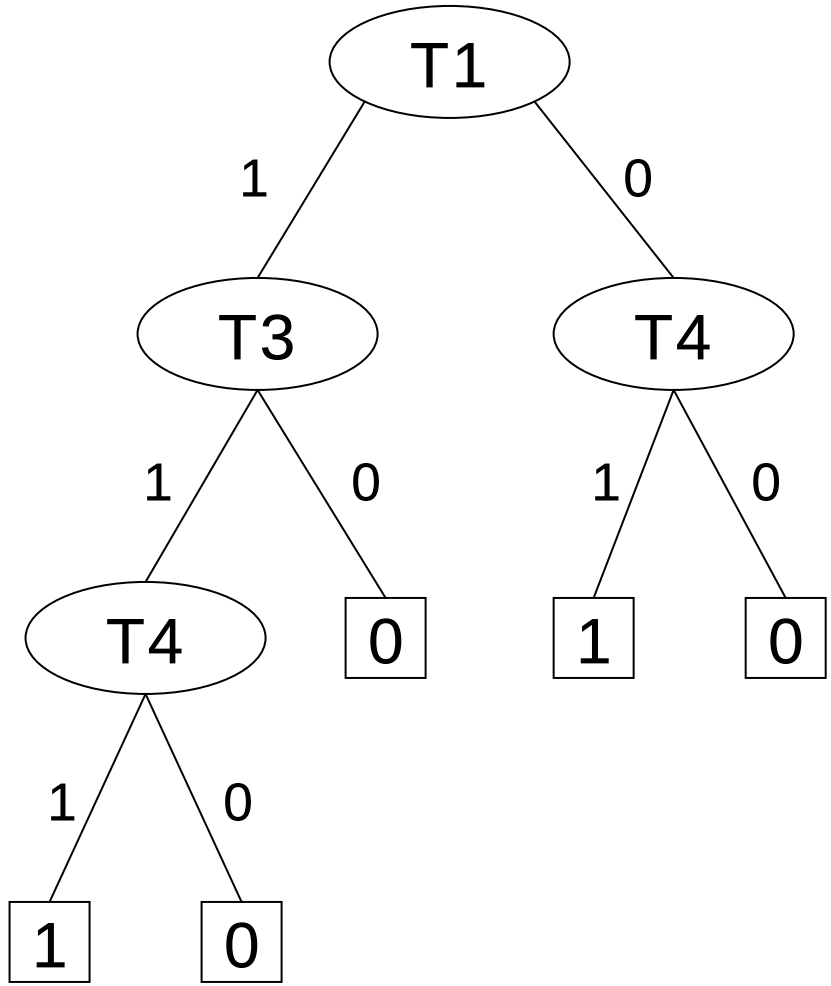
テスト

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)

```
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
```



クラス分類問題の様々な応用例

- 製品の歩留まりの向上
  - 目標属性　機械の故障の有無
  - テスト　機械の状態
- 薬品効果の分類
  - 目標属性　薬の効果　発病
  - テスト　心拍数　血糖値　血圧　遺伝子変異
- 顧客の分類
  - 目標属性　顧客の忠誠度　自動車事故の有無
  - テスト　年齢　収入　購入履歴
- 詐欺行為の検出
  - 目標属性　詐欺行為の有無
  - テスト　口座引落回数などの予兆

テスト属性

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

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**

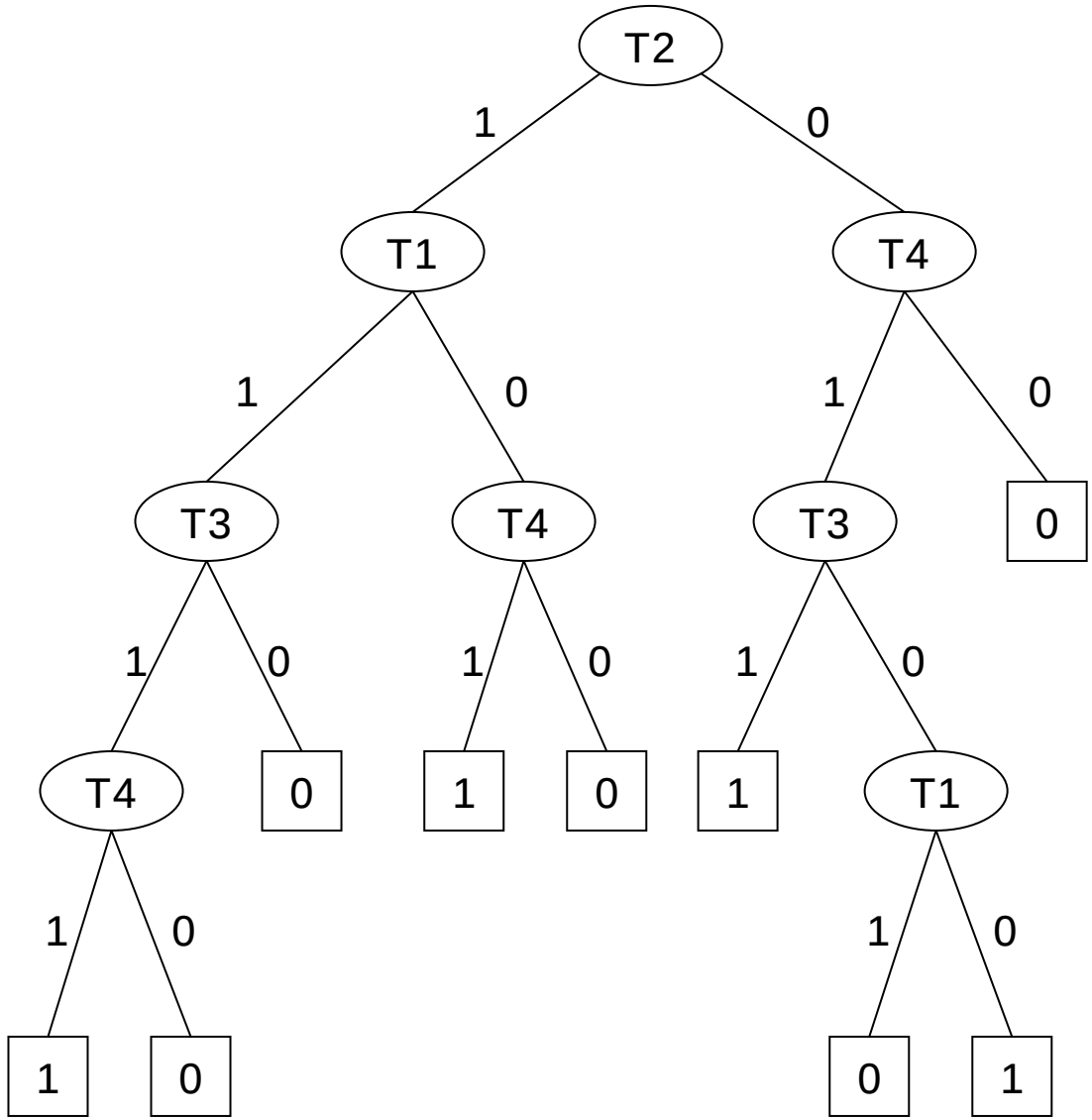
目標値　**目標属性の値から一つ選択.**　**例えば** 1

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)

決定木の最小化はＮＰ困難

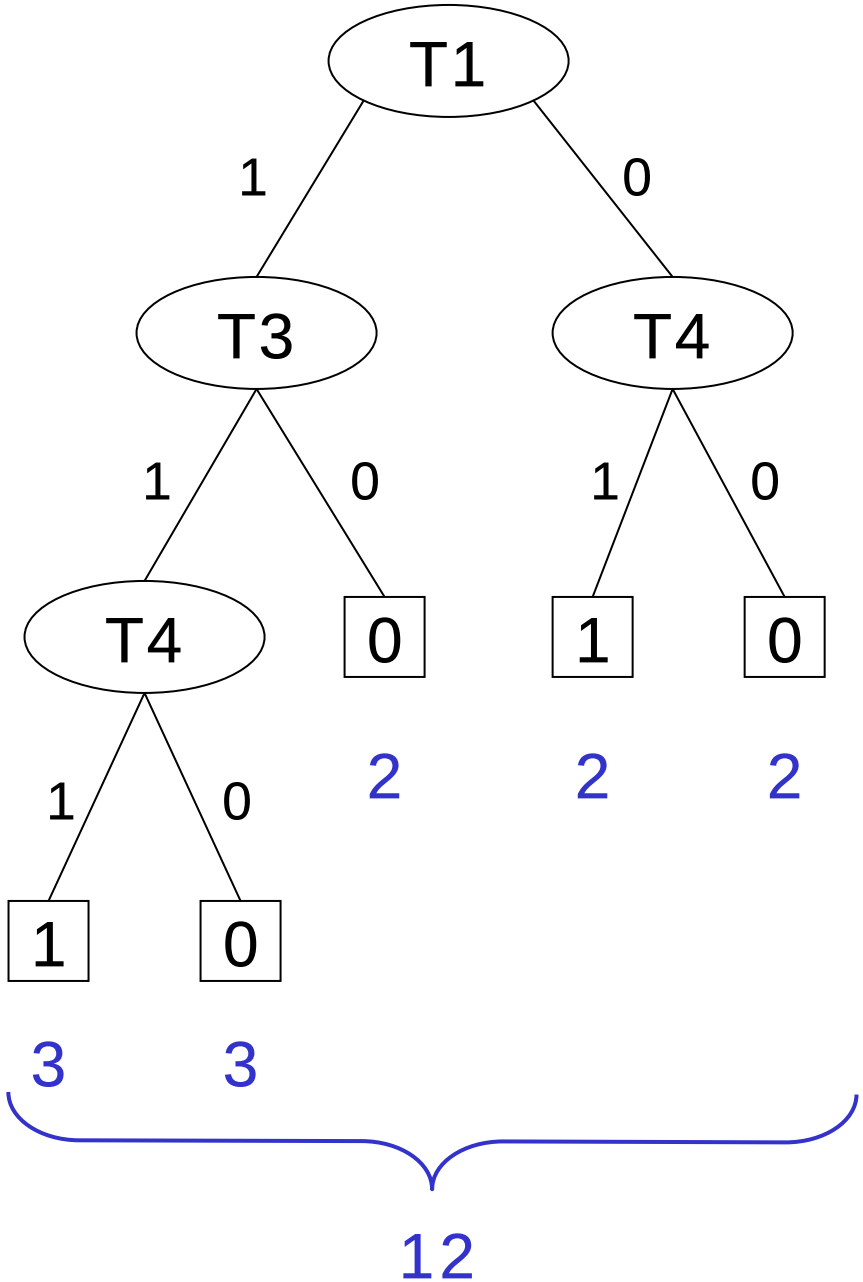
T1	T2	T3	T4	目標属性
1	1	1	1	1
1	1	1	0	0
1	1	0	1	0
1	1	0	1	0
0	1	0	1	1
0	1	0	1	1
0	1	0	1	1
0	1	0	0	0
1	0	1	1	1
1	0	1	1	1
0	0	1	1	1
0	0	0	1	1
1	0	0	1	0
0	0	1	0	0
1	0	1	0	0
0	0	1	0	0



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
  $= \sum \{\mathrm{cost}(x) \mid x$ は葉ノード$\}$
- 決定木のコストは小さいほうが望ましい



- **コストが与えられた閾値以下の決定木は存在するか否か？**
  ＮＰ完全

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

- **従って、コスト最小の決定木を求める最適化問題はＮＰ困難**

ＮＰであること
- **決定木を生成する非決定性チューリング機械Ｔを構成する**
- **Ｔが出力する決定木のコストは多項式時間で計算可能**

ＮＰ完全であること
- EXACT COVER BY 3-SETS **問題を決定木のコストを計算する問題に多項式時間で変換できる**

EXACT COVER BY 3-SETS 問題 （NP完全）

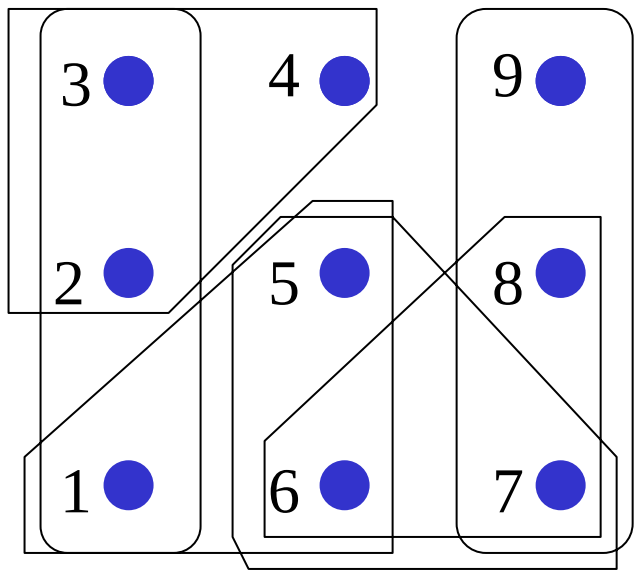
3の倍数の個数だけ元を含む有限集合 X と、
3個の元を含む X の部分集合の集まり $S = \{T1, T2, \ldots\}$ が
与えられたとき、 次の条件を満たす $S1 \subseteq S$ は存在するか？

- $\bigcup \{T \mid T \in S1\} = X$
  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

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

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



EXACT COVER BY 3-SETS **問題のＮＰ完全性については**

M. R. Garey and D. S. Johnson.
Computers and Intractability. A Guide to NP-Completeness
W. H. Freeman, 1979

SAT ➞ 3SAT ➞ 3D Matching ➞ EXACT COVER BY 3-SETS

EXACT COVER BY 3-SETS

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

X の元を正レコード
$|X|$ 個の負レコードを新たに生成し $Y=\{y1, y2, \ldots\}$ とおく

$t[A] = 1 \quad t \in X$

$= 0 \quad t \in Y$

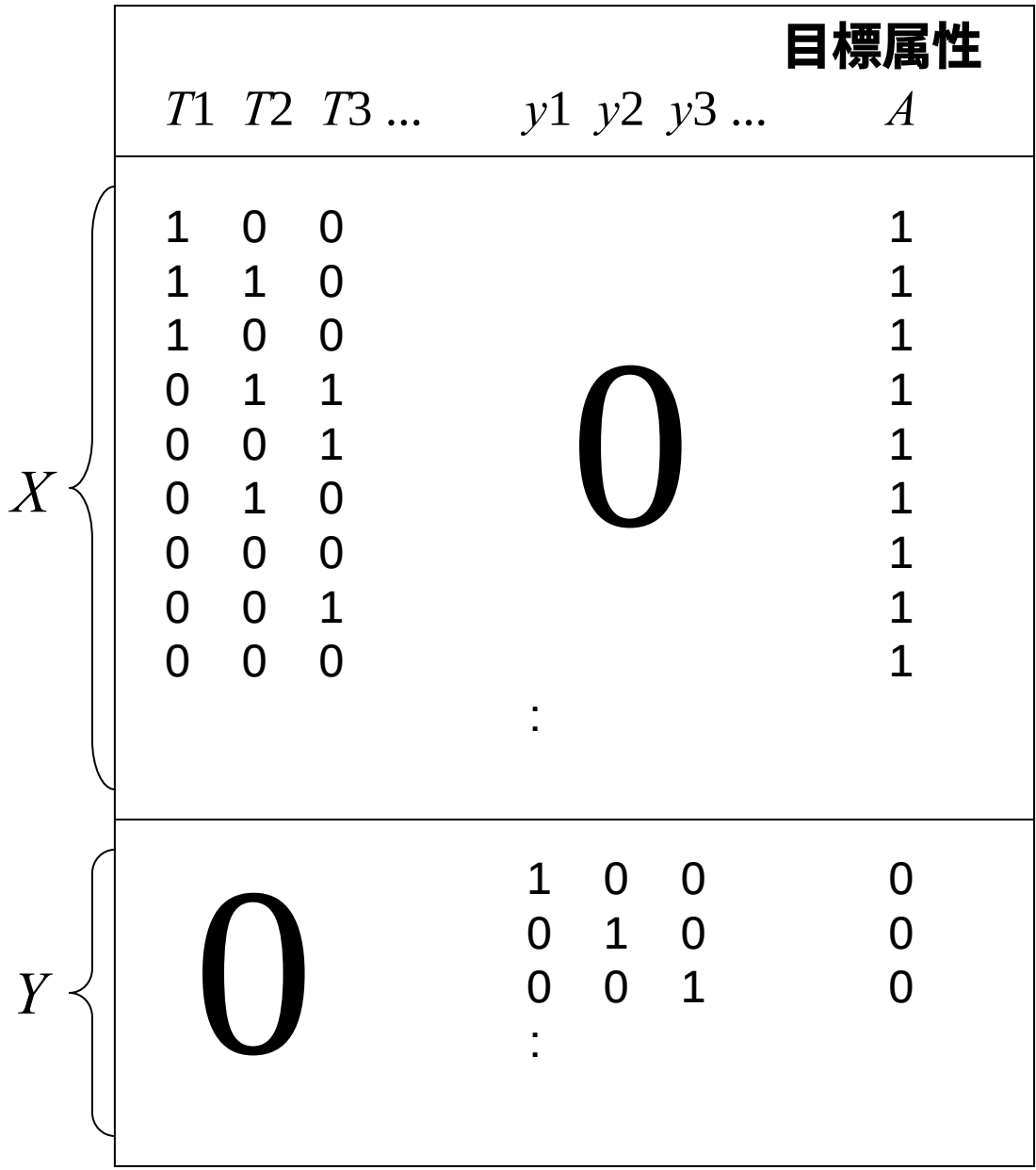
$t \in X \cup Y$ に対して
$S \cup Y$ の元はテスト属性

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

$= 0$ otherwise

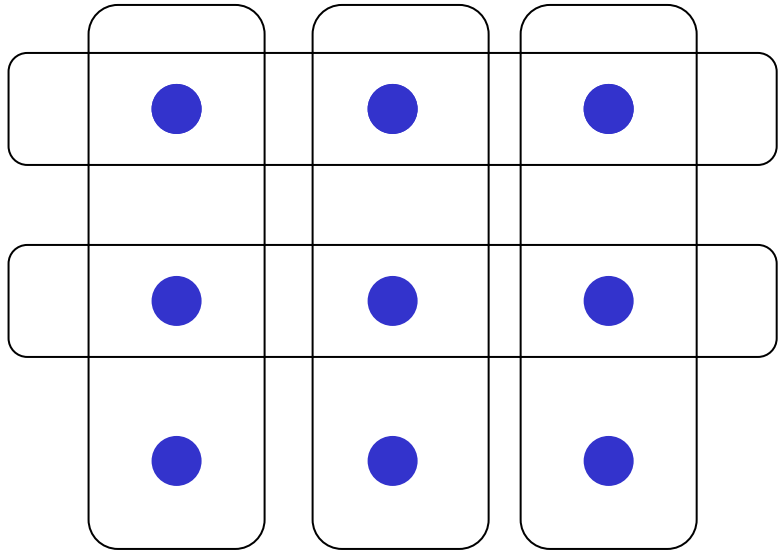
$t[y] = 1 \quad t = y \ (y \in Y)$

$= 0$ otherwise



多項式時間と領域で変換可能

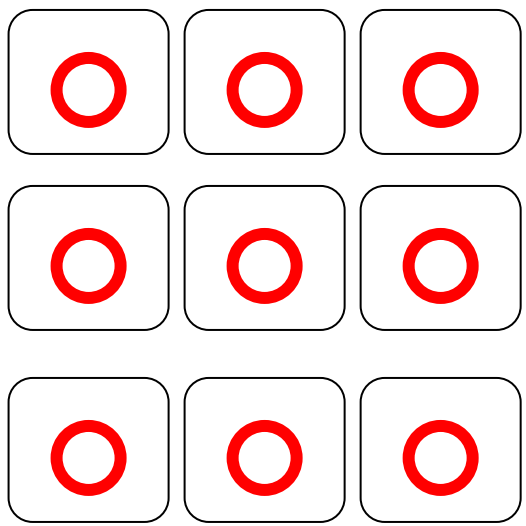
例



正レコード　$t[A]=1$　Xの元

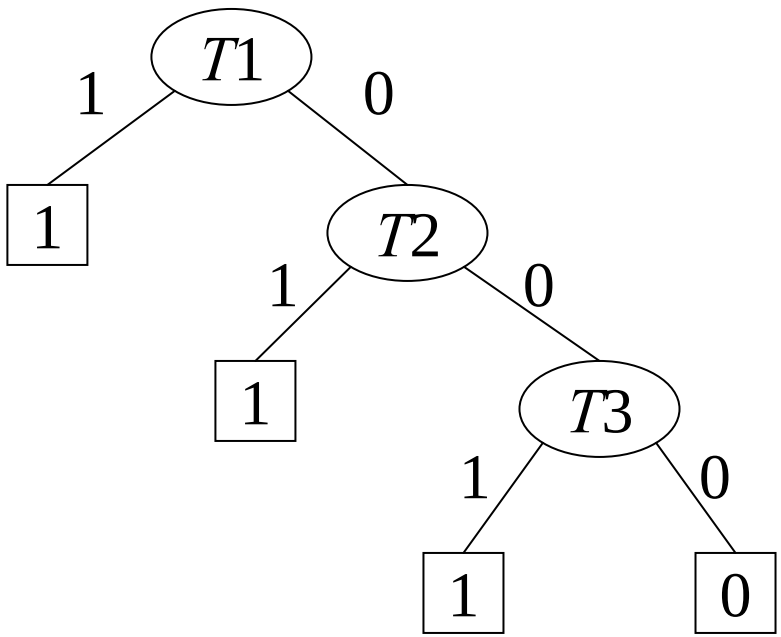
負レコード　$t[A]=0$　Yの元

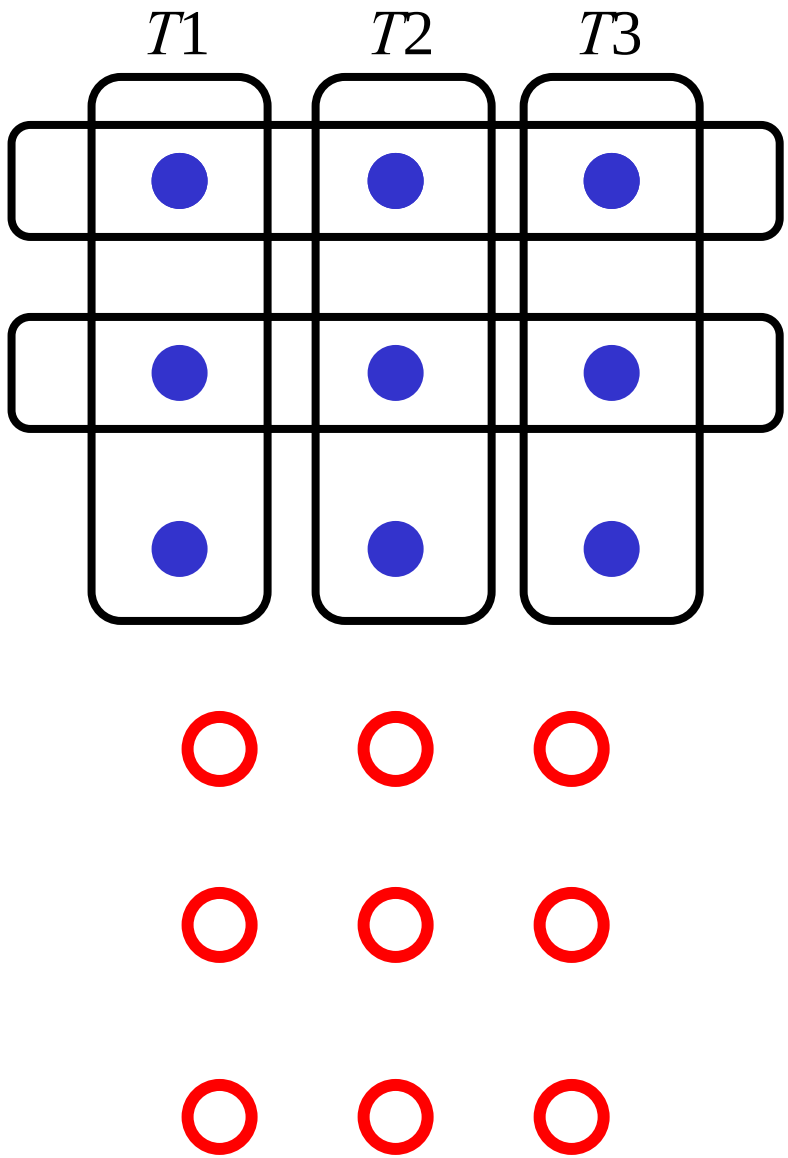
テスト属性



正レコードと負レコードを決定木で分離する戦略1

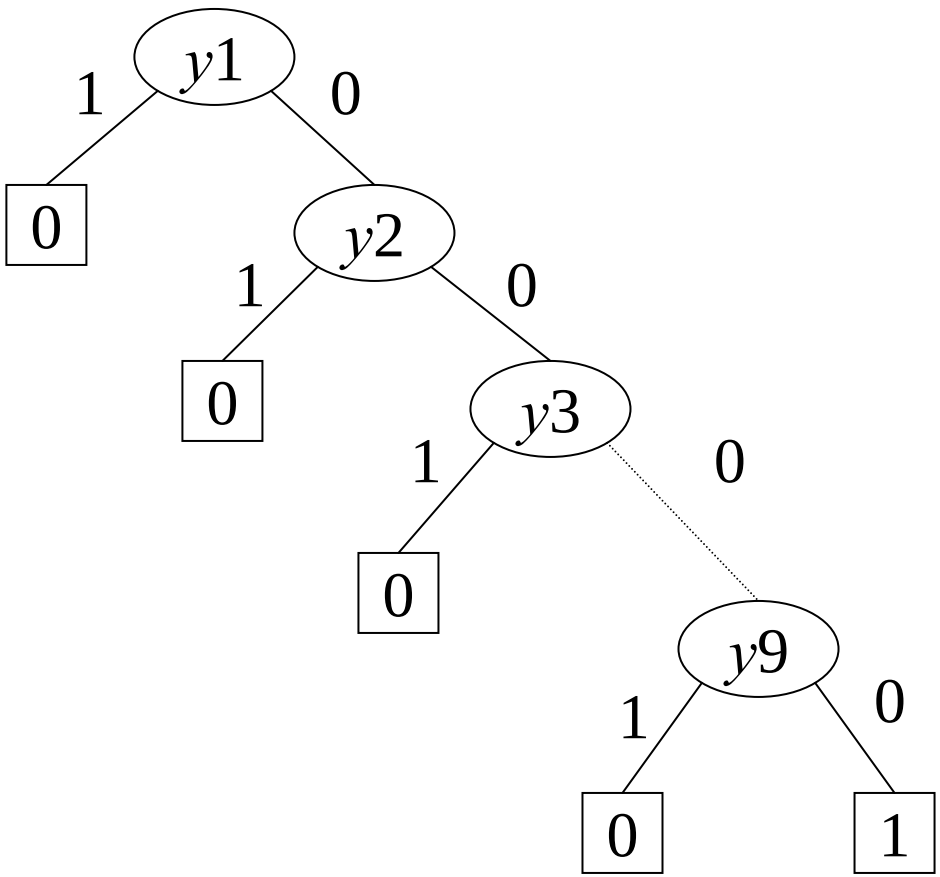
$S = \{T1, T2, \ldots\}$ **のテスト属性を使って正レコードを包含する**

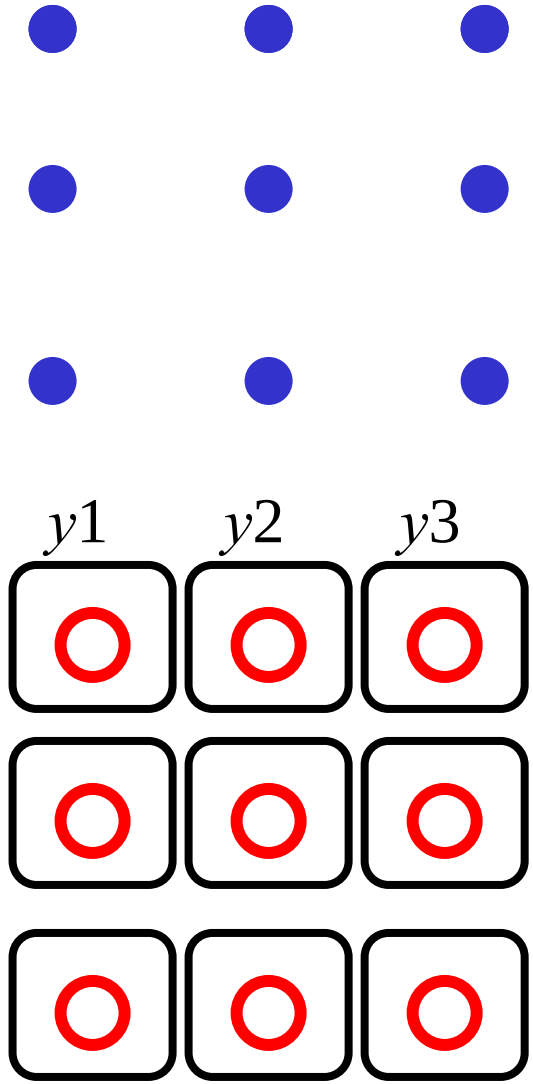




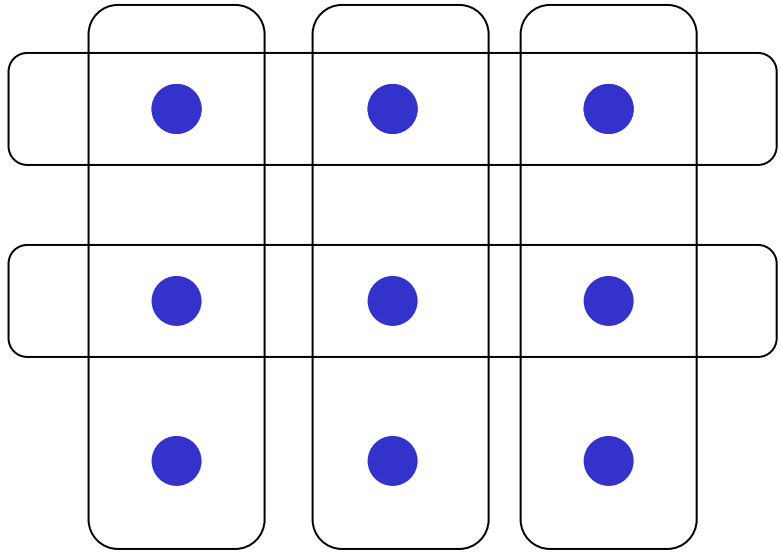
正レコードと負レコードを
決定木で分離する戦略2

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



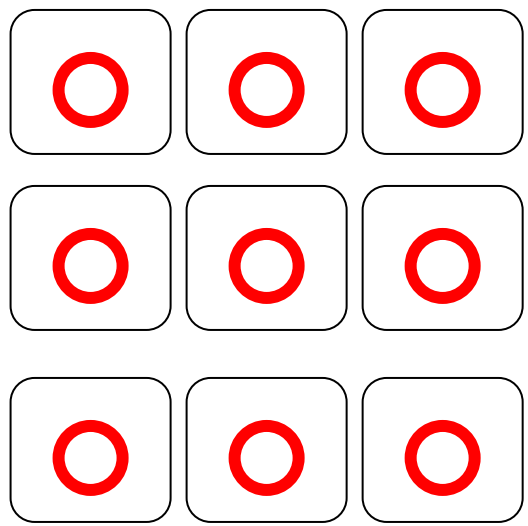


正レコードと負レコードを
分離する決定木の構成法

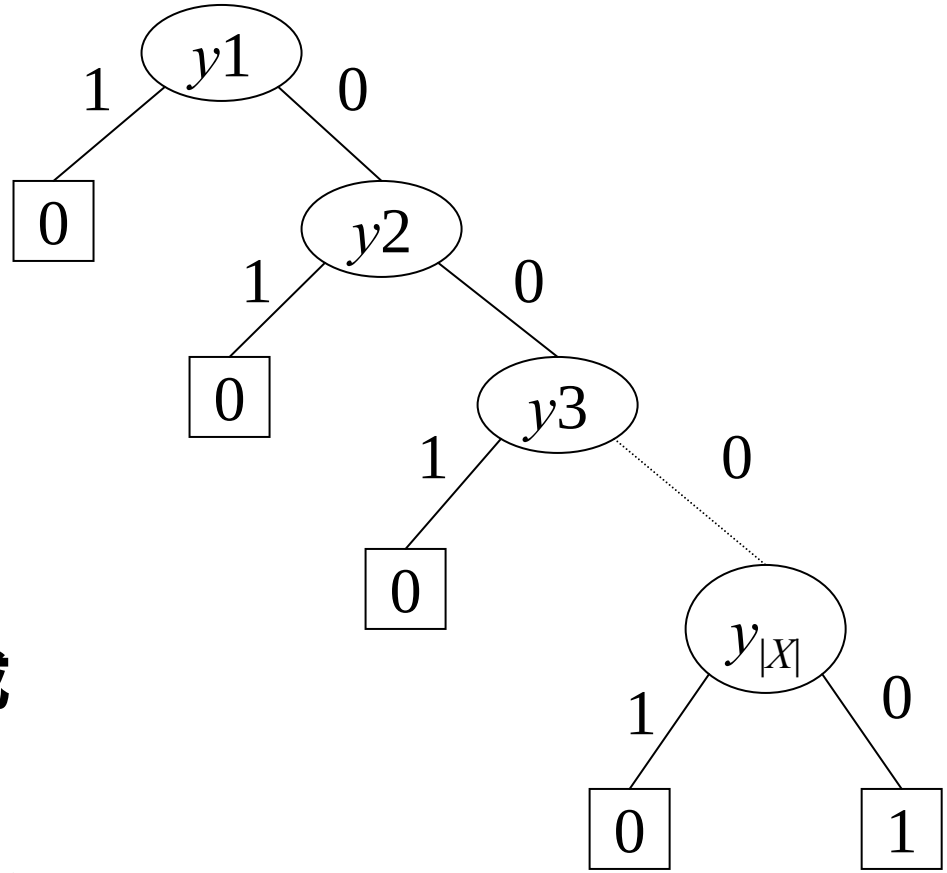


戦略1
正レコードをすべて包含する
S の部分集合をテスト属性として使う
(Y のテスト属性を使うのは無駄)

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.**

近似的解法

Greedy Algorithm

エントロピー

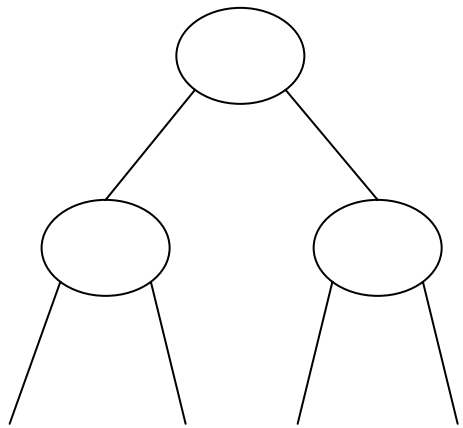
近似的解法　　貪欲(greedy)アルゴリズム

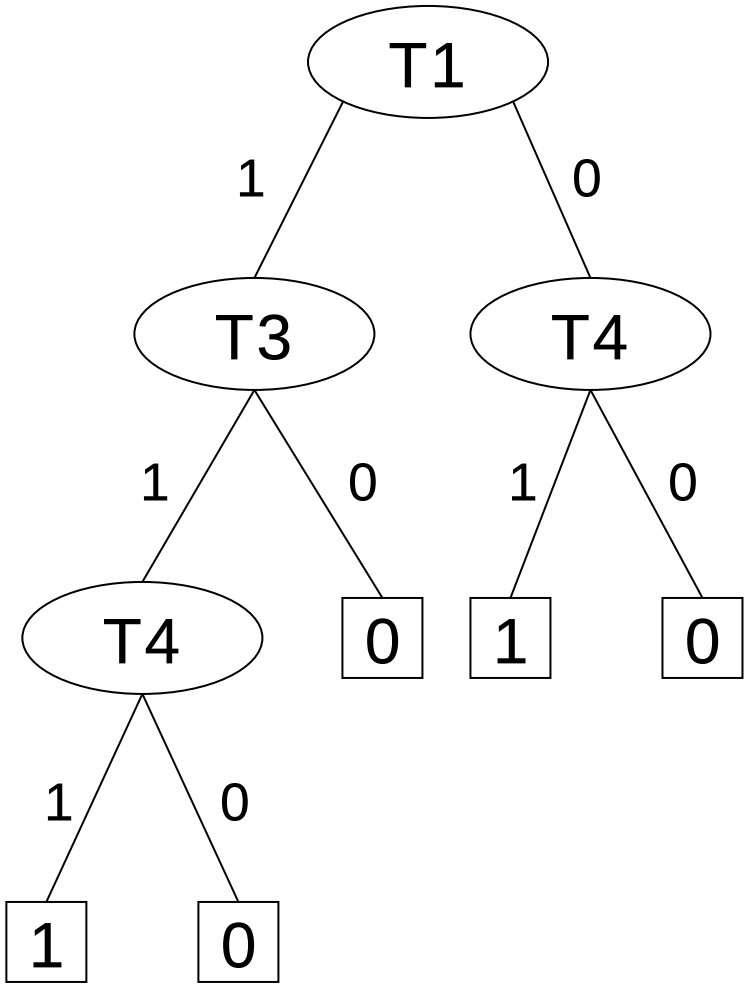
- 「最も良さそうな」テストを選択し、
  レコード集合を分割

- 分割したレコード集合に
  同じ戦略を繰り返し適用する

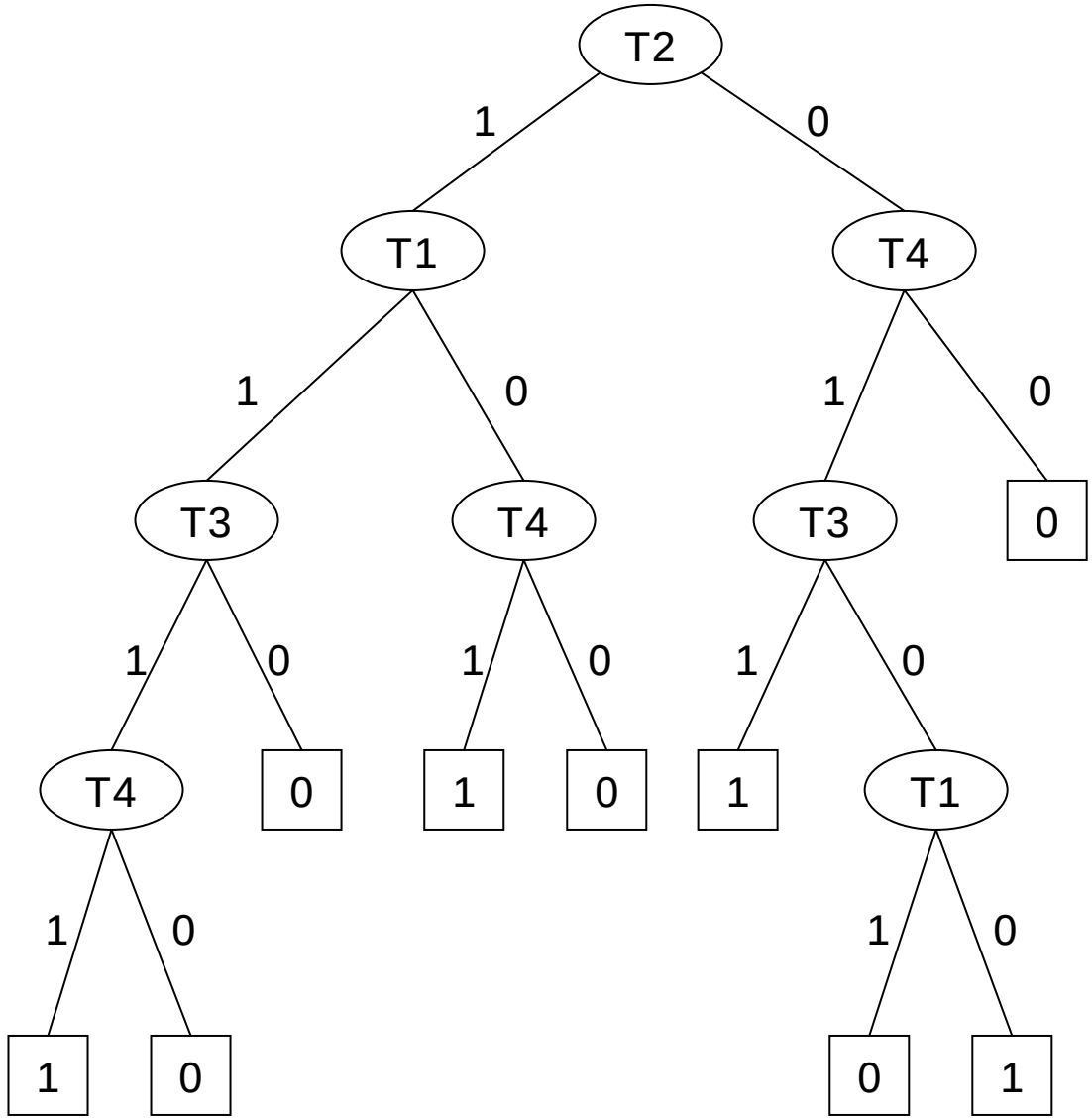
- 一度選択したテストは変更しない

- 「最も良さそうな」 の基準を工夫





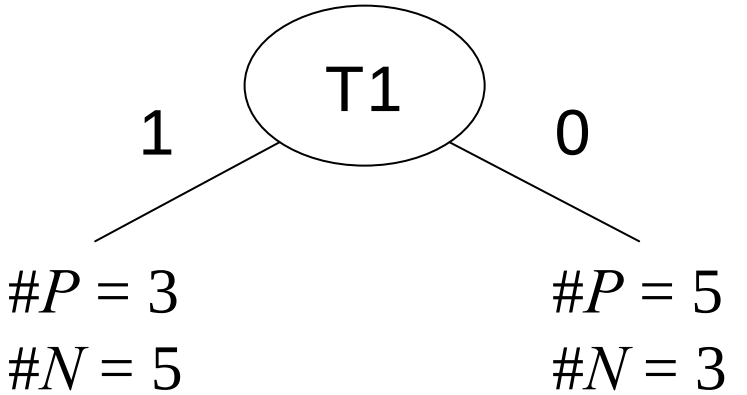
3 + 3 + 2 + 2 + 2 = 12

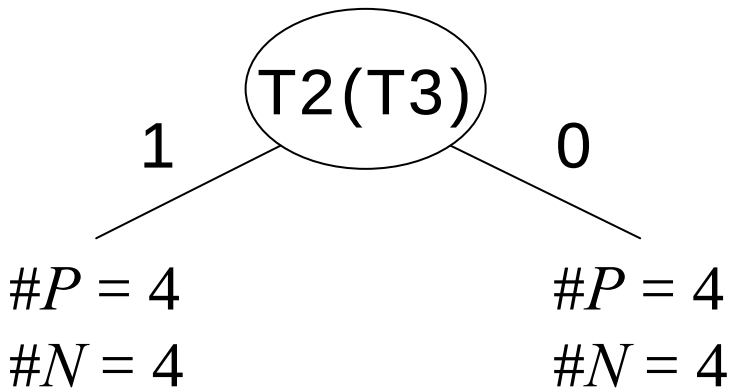


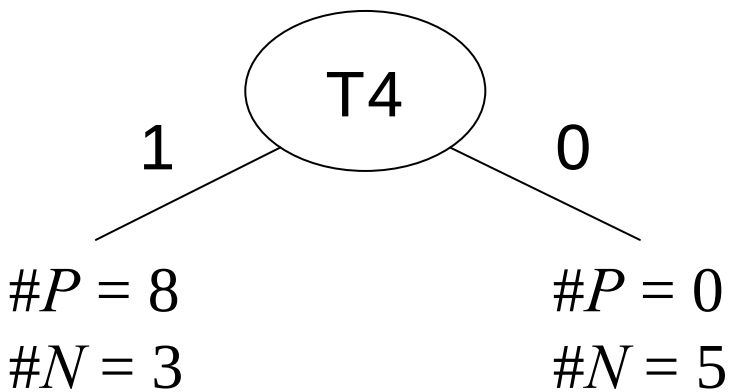
4 + 4 + 3 + 3 + 3 + 3 + 4 + 4 + 2 = 30

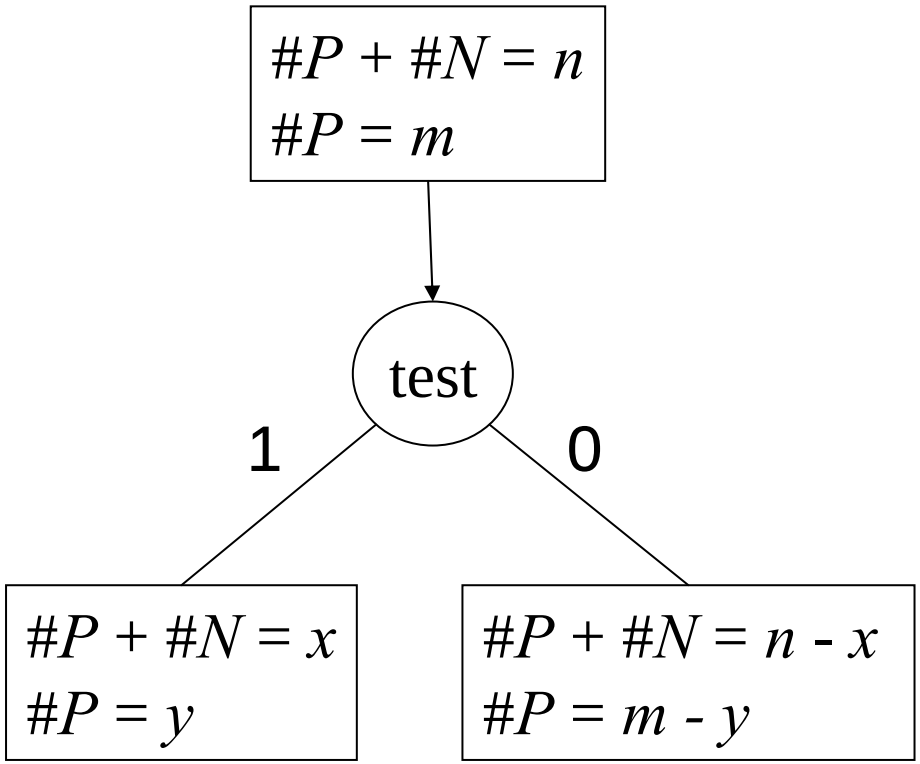
$\#P$: **正レコードの数**　$\#N$: **負レコードの数**

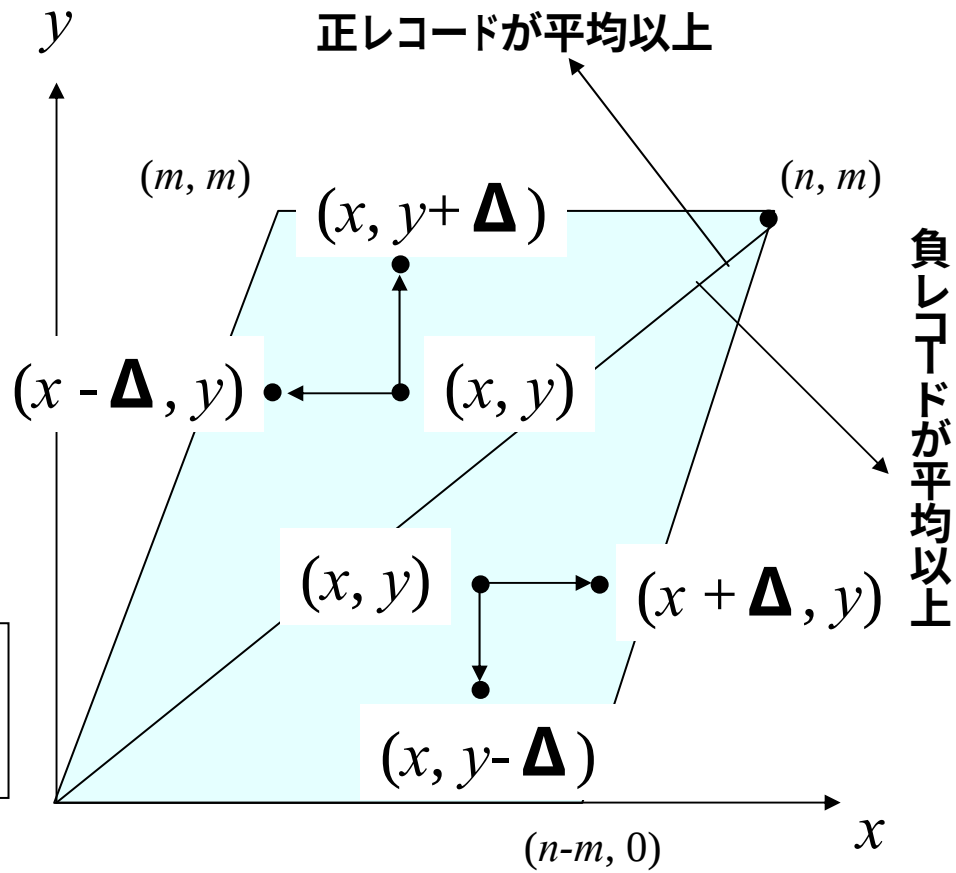
T1	T2	T3	T4	目標属性
1	0	1	1	1
1	0	1	1	1
1	1	1	1	1
1	1	1	0	0
1	0	1	0	0
1	1	0	1	0
1	0	0	1	0
1	1	0	1	0
0	1	0	1	1
0	0	1	1	1
0	1	0	1	1
0	1	0	1	1
0	0	0	1	1
0	0	1	0	0
0	1	0	0	0
0	0	1	0	0











- test を選択すると (x, y) が定まるので、評価関数を $\Phi(x, y)$ とおく
- 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 $m/n > y/x$

  $\Delta > 0$

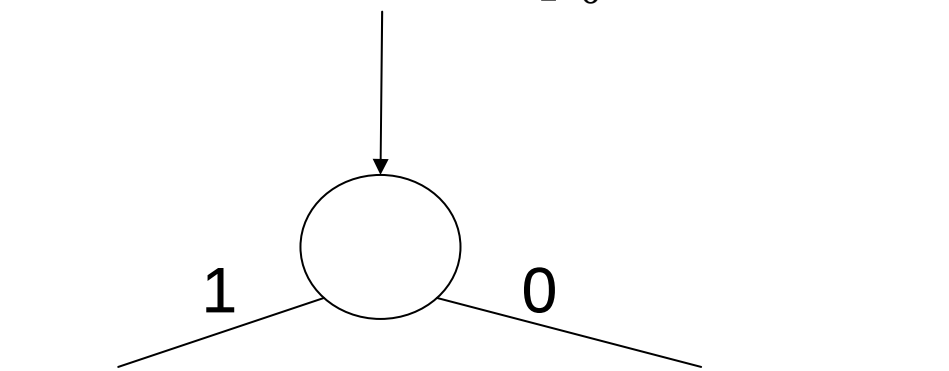
正レコードの割合 $p = \#P / (\#P + \#N)$

負レコードの割合 $1 - p = \#N / (\#P + \#N)$

エントロピー $\mathrm{ent}(p) = - p \log_2 p - (1-p) \log_2 (1-p)$

レコード数 n

正レコードの割合 $p_0 = m / n$



$n_1 = x$ $\quad$ $n_2 = n - x$

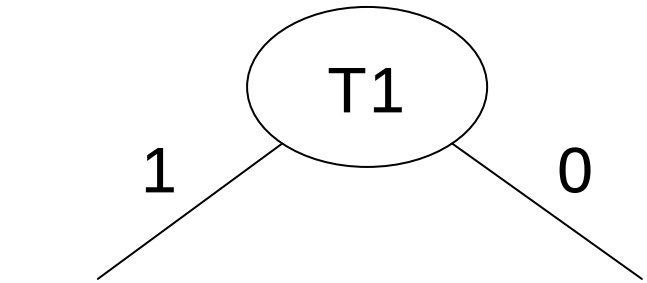
$p_1 = y / x$ $\quad$ $p_2 = (m - y) / (n - x)$

Entropy 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)
$= 2\ (-\ (1/2)\ \log_2(1/2))$
$= 1$



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

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

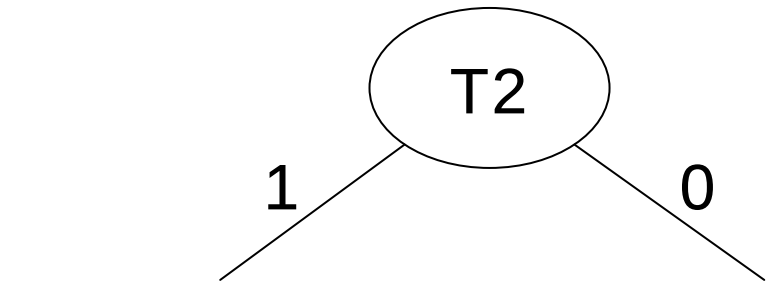
ent(3/8)
$= -\ (3/8)\log_2(3/8) - (5/8)\log_2(5/8)$
$= 0.95444$

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

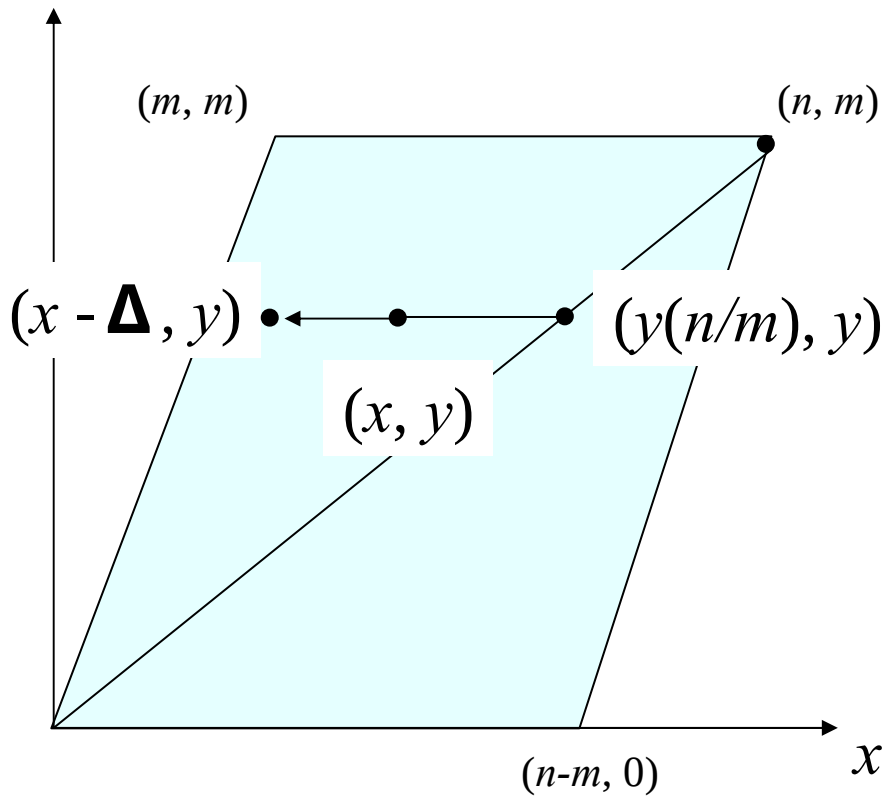
Entropy Gain = ent(8/16) - (8/16)ent(4/8) - (8/16)ent(4/8)
= 0

- 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

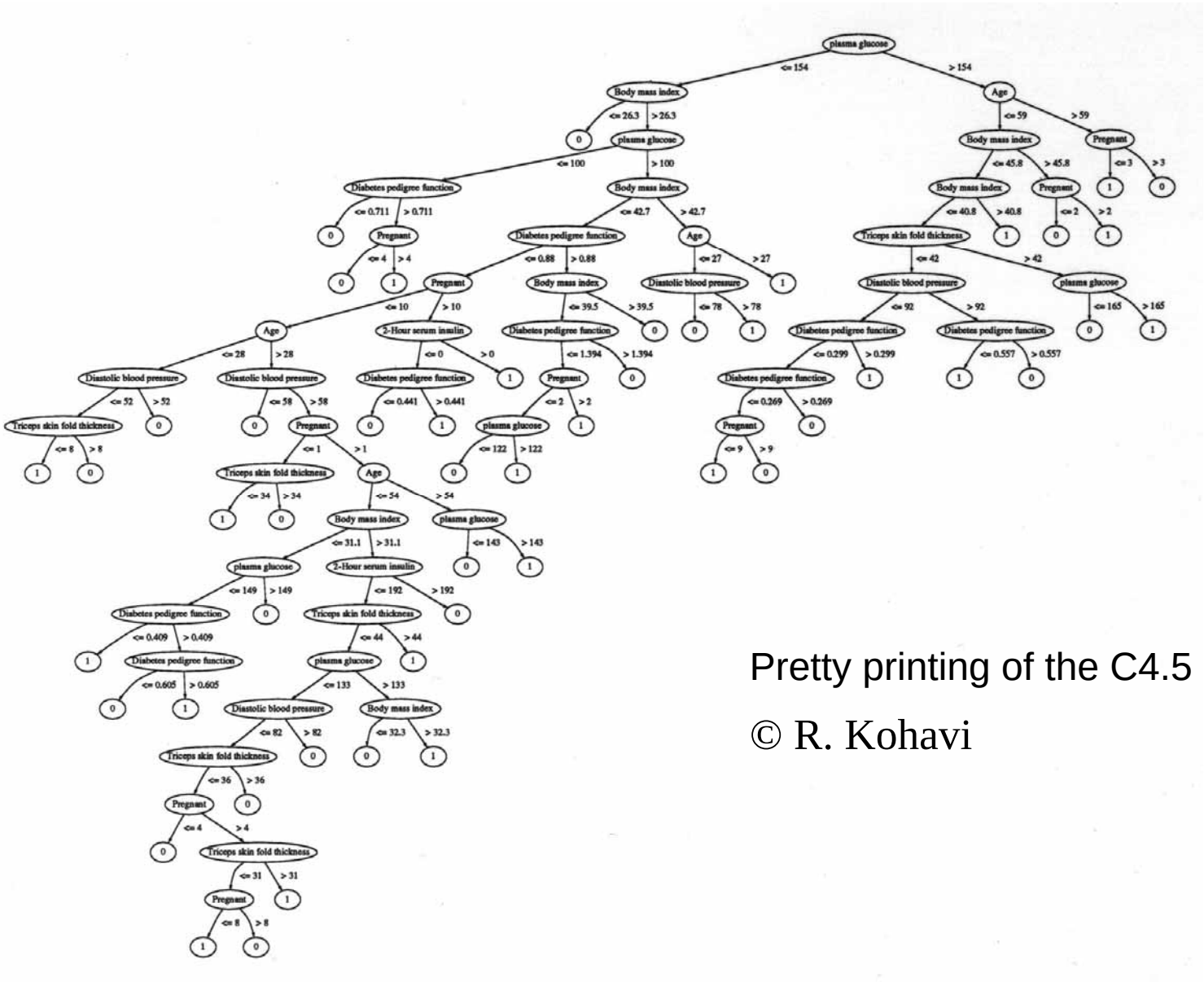
$$\mathrm{Ent}(\lambda \mathbf{v1} + (1-\lambda)\mathbf{v2}) \leqq \max\{\mathrm{Ent}(\mathbf{v1}), \mathrm{Ent}(\mathbf{v2})\}$$



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

巨大な決定木と Overfitting 問題

決定木は巨大になりがち



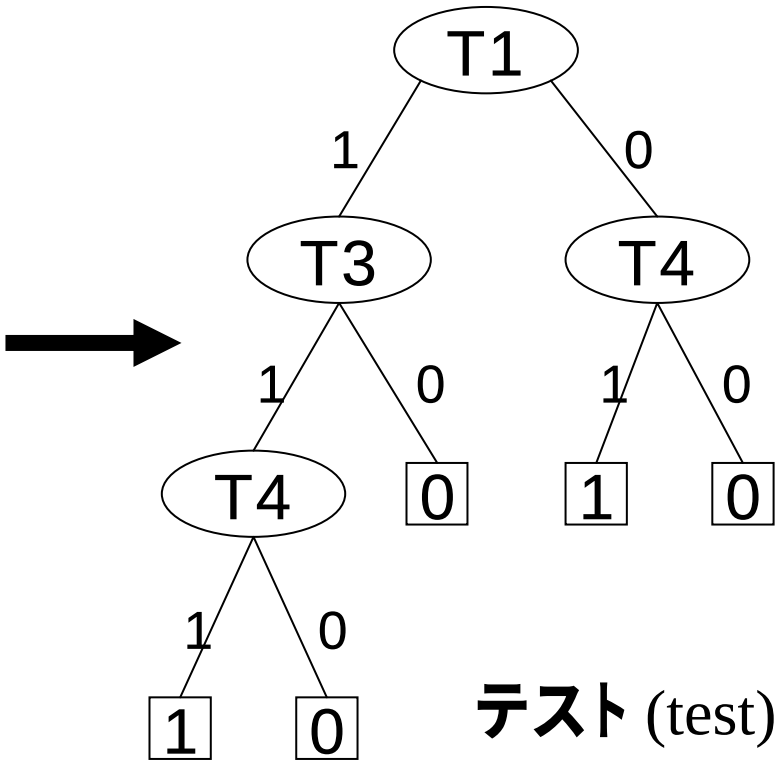
Pretty printing of the C4.5 output

© R. Kohavi

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

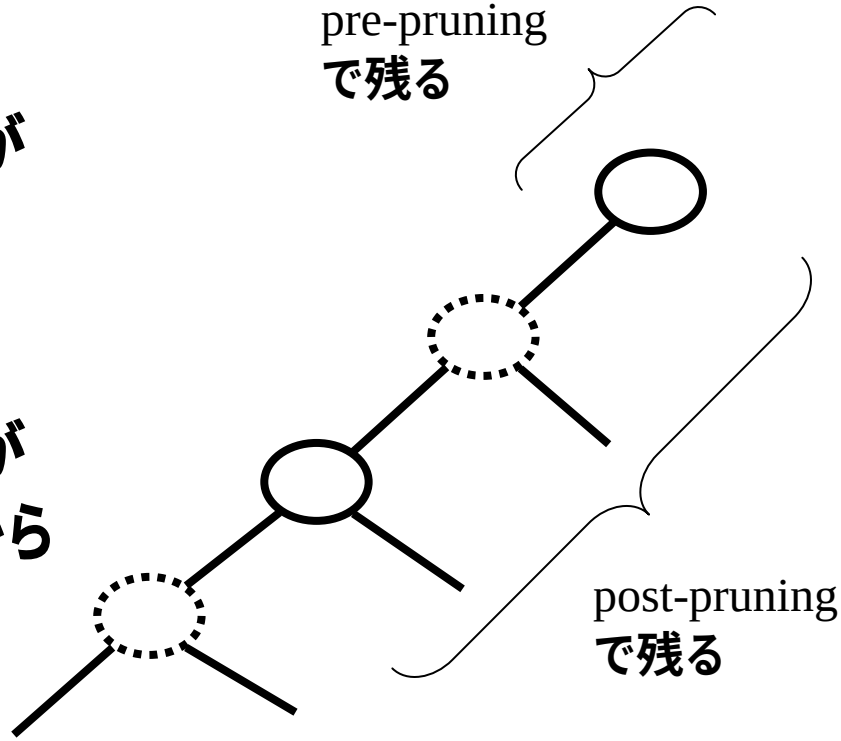
Overfitting **問題**

- **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**

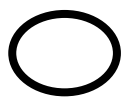
Overfitting の回避方法1

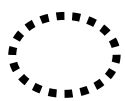
- pre-pruning 法
  決定木を構成中に Entropy Gain が
  小さいテスト属性は生成しない

- 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


Entropy Gain が
大きいテスト属性


Entropy Gain が
小さいテスト属性

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}）
  O(k) k **個の離散値**

- **テスト属性が数値の場合**
  例 21 $\leqq$**年齢** $\leqq$ 44
  多項式時間

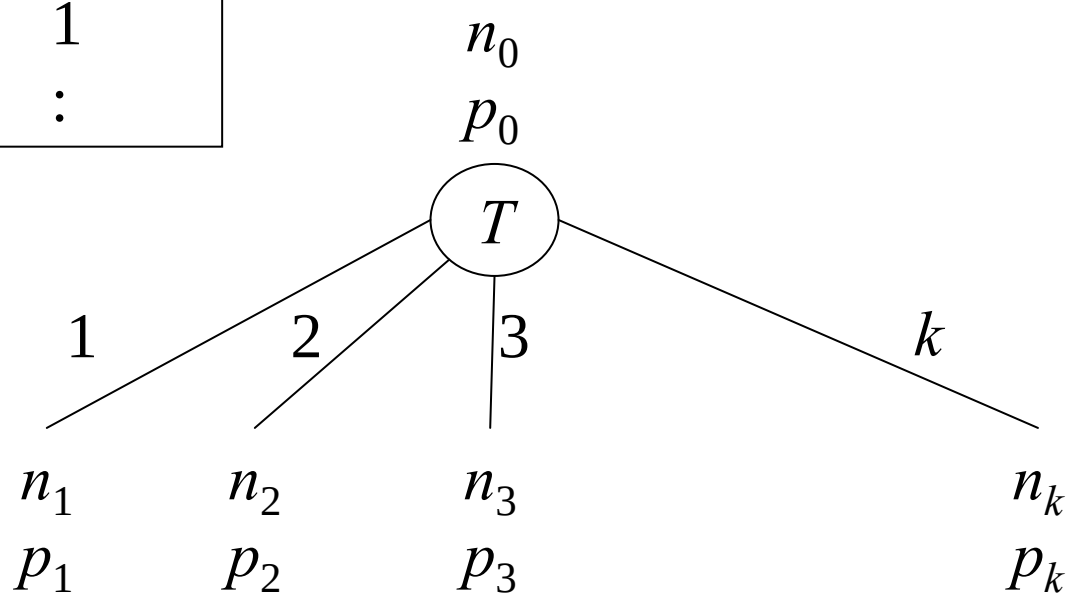
テスト属性 T の領域に3つ以上の異なる離散値がある場合

- T の領域を $\{1, 2, \ldots, k\}$

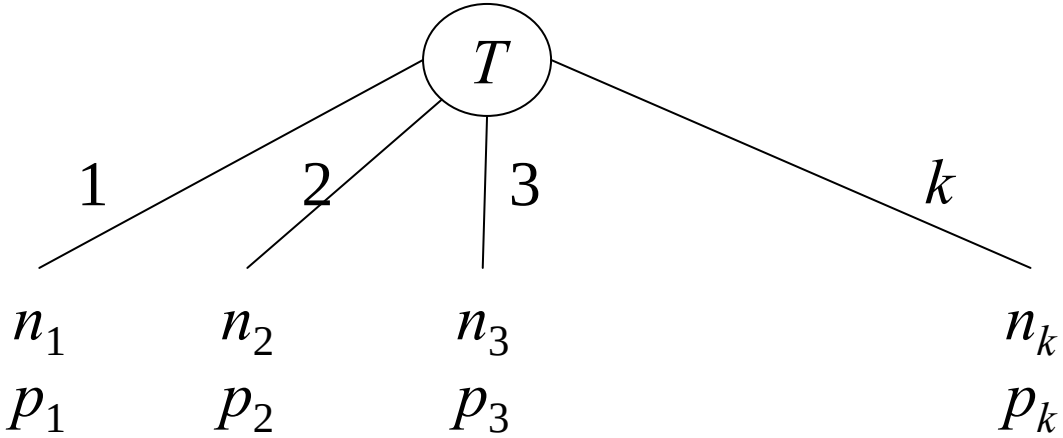
T ...	目標属性A
2	1
3	0
1	1
:	:

<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



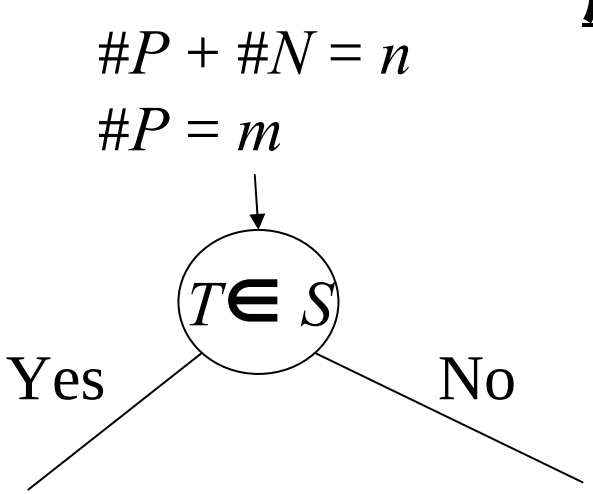
$$\mathrm{ent}(p_0) - \sum \left(n_i / n_0\right) \mathrm{ent}(p_i)$$



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

属性値の部分集合で分割する方式

$S \subseteq \{1, 2, \ldots, k\}$



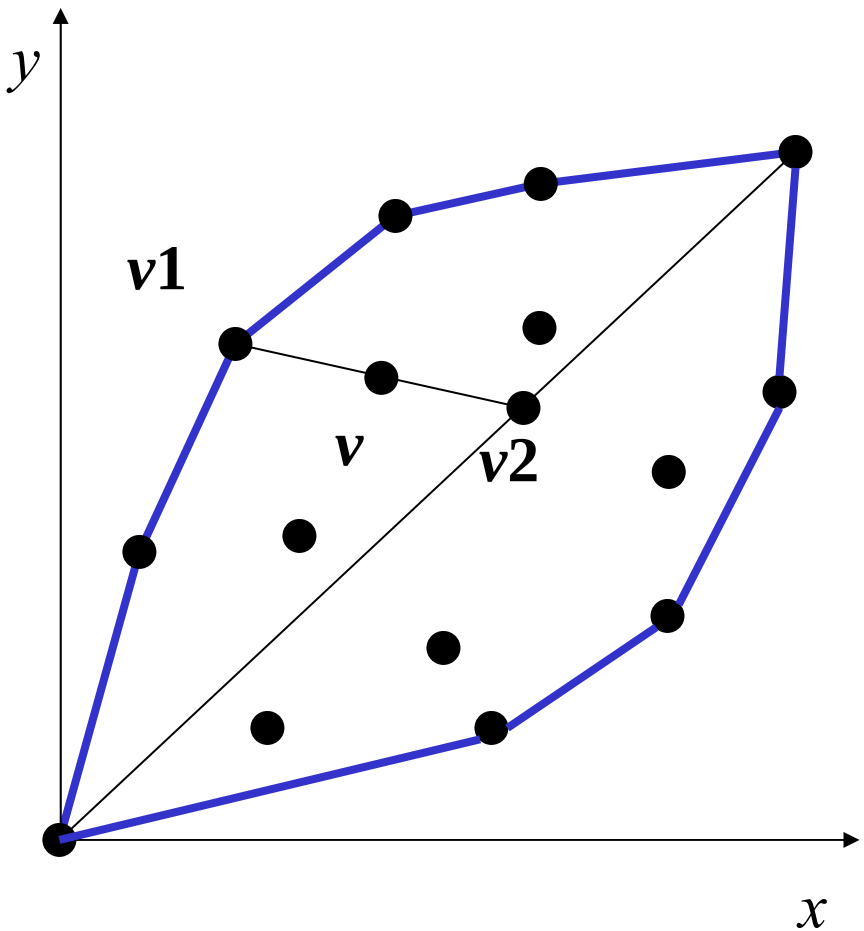
$\#P + \#N$
$= \sum \{n_i \mid i \in S\} = x$
$\#P$
$= \sum \{p_i n_i \mid i \in S\} = y$

$\#P + \#N$
$= \sum \{n_i \mid i \notin S\}$
$\#P$
$= \{p_i n_i \mid i \notin S\}$

定理
ある $1 \leqq j \leqq k$ **が存在し**
$S = \{i \mid 1 \leqq i \leqq j\}$ **もしくは**
$S = \{i \mid j \leqq i \leqq k\}$
が Entropy Gain **を最大化**

各テスト $T \in S$ に 2次元座標の点(x, y) を対応させる

凸包 (Convex Hull)
すべての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)

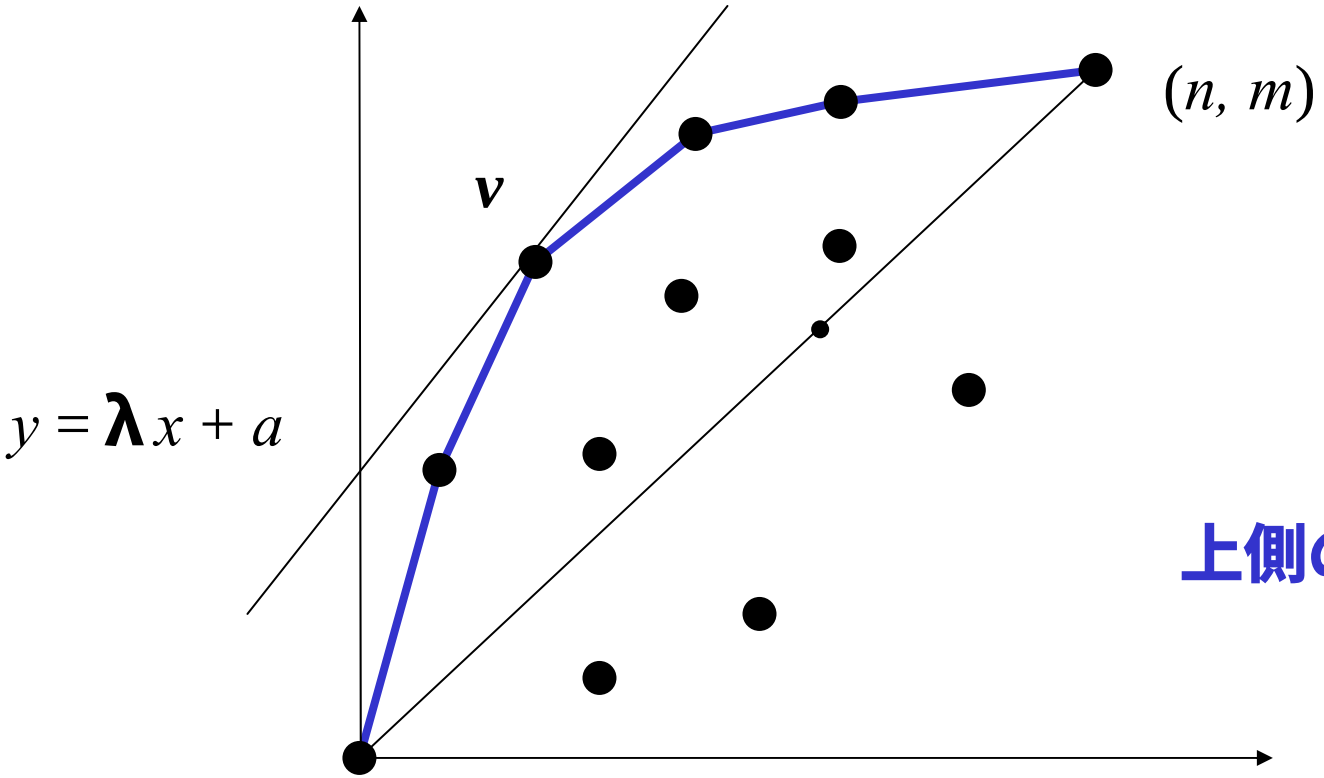


Ent(***v***) ≦ max{Ent(***v1***), Ent(***v2***)}

Ent(***v2***) **は最小値**

Ent(***v***) ≦ Ent(***v1***)

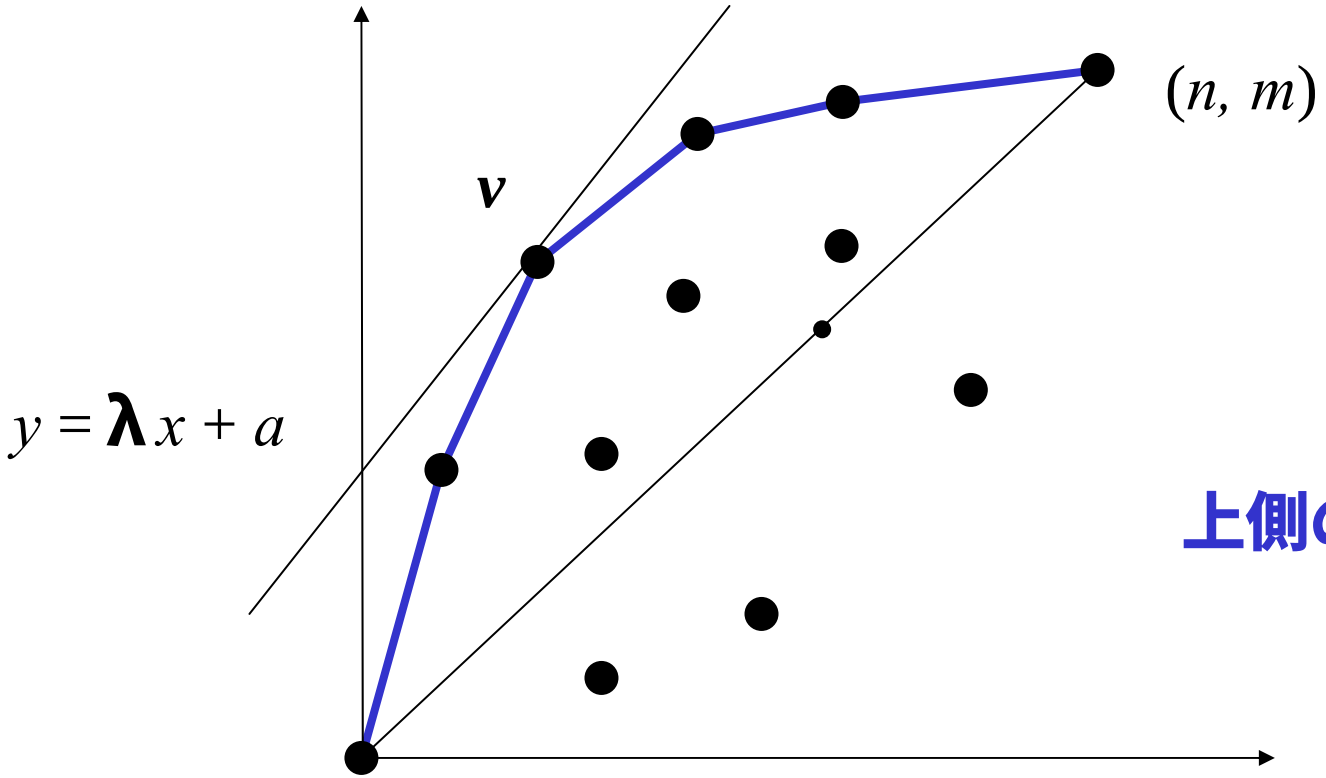
Entropy Gain **を最大化する点は凸包上に存在**



上側の凸包の場合

上側の凸包上の点 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

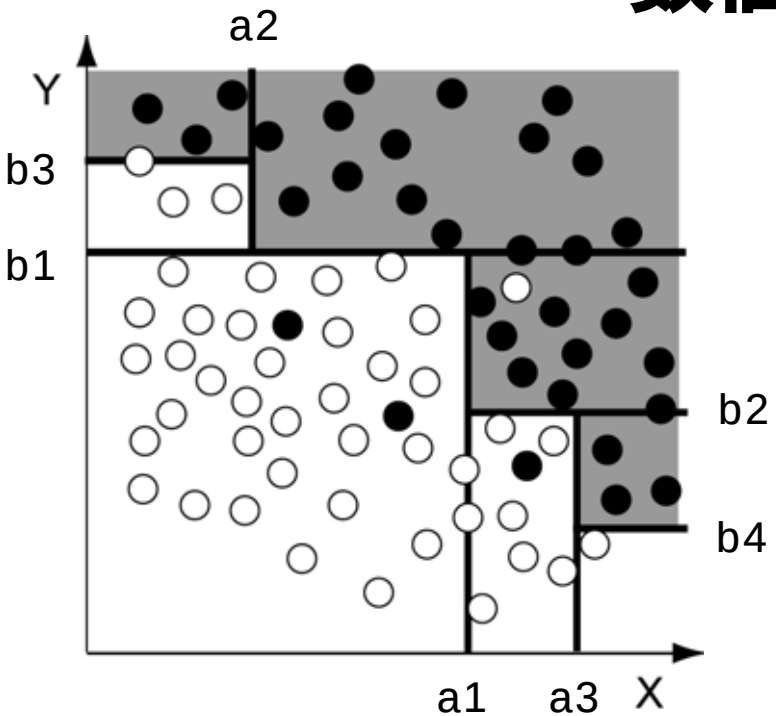
凸包上の点 $v = (\Sigma\{n_i \mid i \in S\},\ \Sigma\{p_i n_i \mid i \in S\})$ **に対して傾き** λ **の直線が存在し、全点中で** v **は** y **切片** a **を最大化すると仮定**

$$a = \Sigma\{p_i n_i \mid i \in S\} - \lambda\ \Sigma\{n_i \mid i \in S\} = \Sigma\{(p_i - \lambda) n_i \mid i \in S\}$$

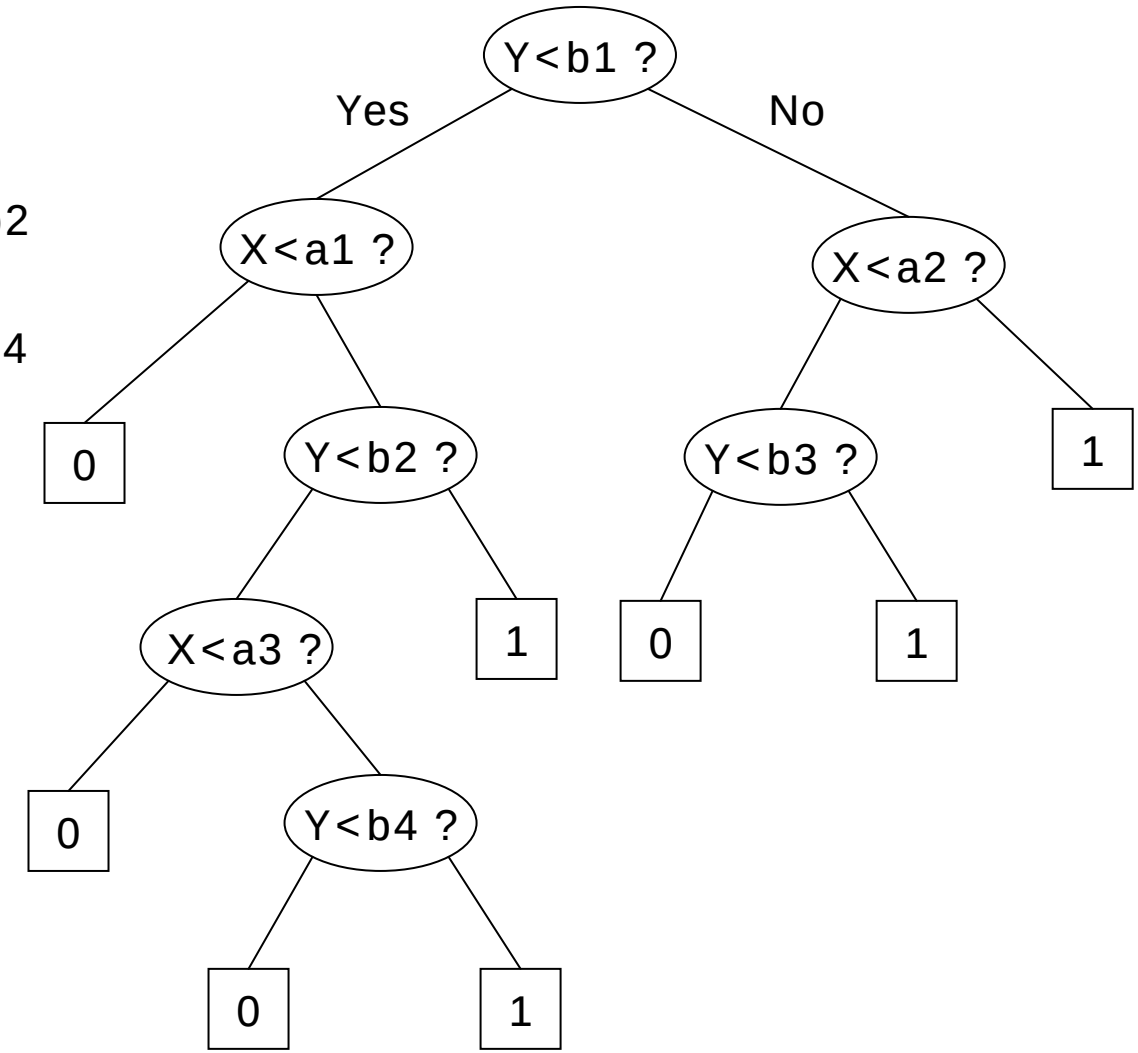
$p_j \geqq \lambda \geqq p_{j+1}$ **のとき最大化するのは** $p_i - \lambda \geqq 0$ **を満たす** $S = \{1, \ldots, j\}$

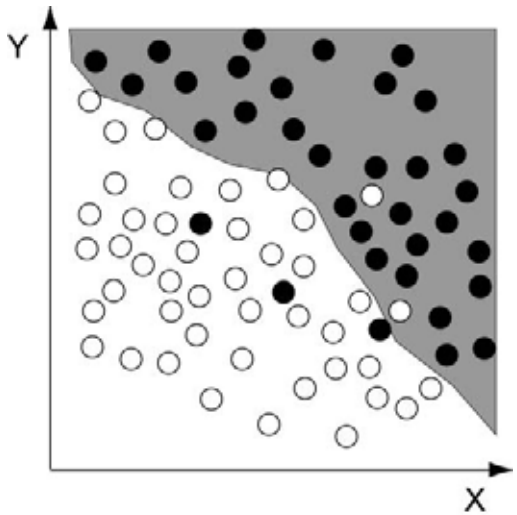
$\lambda > p_j$ **のときは** $S = \phi$, $p_k > \lambda$ **のときは** $S = \{1, \ldots, k\}$.

数値属性の場合

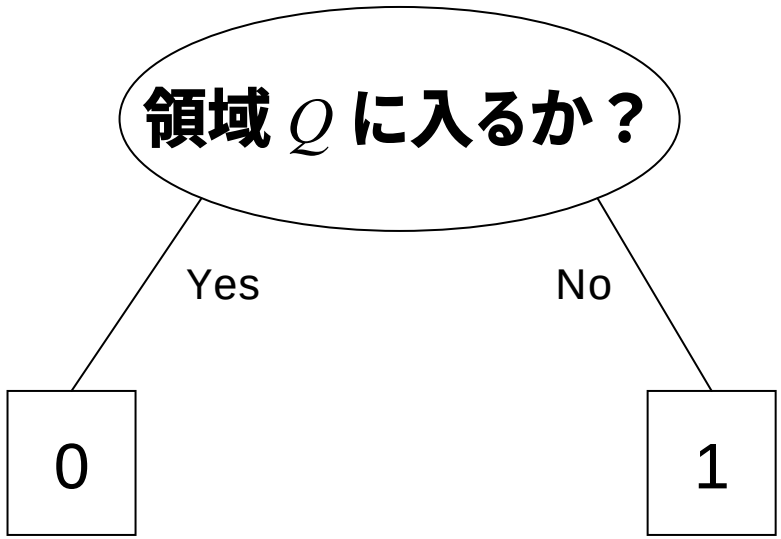


● 正レコード
○ 負レコード





領域 Q



多項式時間で効率的に計算できる領域 Q のクラスが存在

Takeshi Fukuda, Yasuhiko Morimoto, Shinichi Morishita and Takeshi Tokuyama: "Data Mining with Optimized Two-Dimensional Association Rules." *ACM Transactions on Database Systems (TODS)*, Volume 26 , Issue 2, pp. 179 - 213, June 2001.

論理積でレコード集合を分割する場合

$(T1 = 1) \wedge (T2 = 1) \wedge \ldots \wedge (Tn = 1)$

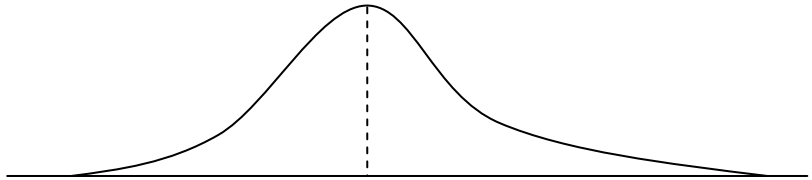
Entropy Gain を最大化する論理積を計算する問題はNP困難
（Set Cover 問題に帰着）

Branch-and-bound による探索空間の枝刈り解法が提案されている

目標属性が数値の場合の拡張

遺伝子の状態

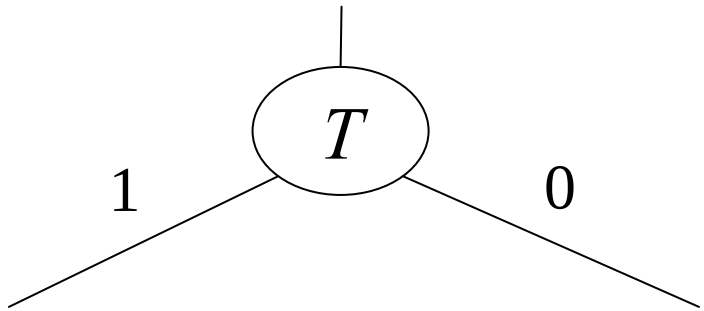
T1	T2 …	**目標属性** A **(血圧)**
1	0	125
0	1	145
1	1	113
:	:	:



レコード数 $= n$

$\Sigma\, t[A] = m$

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 $= x$

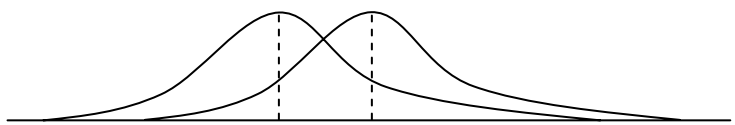
$\Sigma\, \{t[A] \mid t[T]=1\} = y$

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 **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**

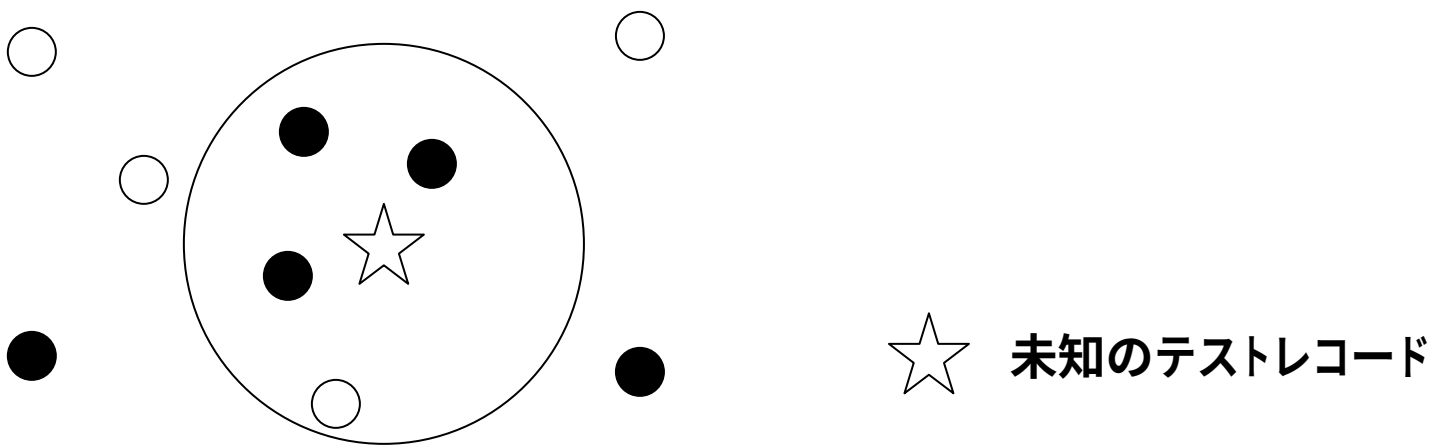
$$\mathrm{Var}(\lambda \mathbf{v1}+(1-\lambda)\mathbf{v2}) \leqq \lambda \mathrm{Var}(\mathbf{v1}) + (1-\lambda)\mathrm{Var}(\mathbf{v2})$$

- 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**

クラス分類周辺の技法
K nearest neighbors

K nearest neighbors

- 規則を学習しない
- 未知のテストレコードと最も類似した（適切な距離を定義し、近い距離にある）K 個の訓練レコードの集合 S を求める
- 未知のテストレコードの目標属性値を S から予測
  例　2値の場合：S の目標属性値の多数決
  　　数値の場合：平均値
- レコード間の類似性を適切に定義することが重要で
  ユークリッド距離やハミング距離など
- K の選択も学習能力を左右



クラス分類周辺の技法
AdaBoost

参考文献

Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119-139, 1997.

三人寄れば文殊の知恵

Boosting **とは正答率の低いクラス分類器を組合せて、**
高い正答率を示すクラス分類器を構成する技法。

訓練用データ

	T1	T2	T3	T4	目標属性
$\vec{x}_1$	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

y_1

$(\vec{x}_1, y_1)$

$\vdots$

$(\vec{x}_N, y_N)$

AdaBoost の基本的考え方

- 初期状態では各レコードに均等な重みを割当てる
- 次のステップを繰返す

1. ランダムに推測するよりは正答率の高い
   (50%を超える) クラス分類器を生成
2. 目標属性の値の予測を誤ったレコードの重みを
   相対的に上げる
   (予測が困難であることを覚えておく)

註:参考論文中ではクラス分類器(classifier)のことを仮説(hypothesis)と呼んでいる

クラス分類器

T1	T2	T3	T4	目標	重み	if T1=1 then Ob=0 else Ob=1	新しい重み
1	0	1	1	1		0	
1	0	1	1	1		0	
1	1	1	1	1		0	
1	1	1	0	0		0	
1	0	1	0	0		0	
1	1	0	1	0		0	
1	0	0	1	0		0	
1	1	0	1	0		0	

 の大きさが重みを表現

新たな
クラス分類器

T1	T2	T3	T4	目標	重み	if T3=1 then Ob=1 else Ob=0	新しい重み
1	0	1	1	1	■	1	■
1	0	1	1	1	■	1	■
1	1	1	1	1	■	1	■
1	1	1	0	0	■	1	■
1	0	1	0	0	■	1	■
1	1	0	1	0	■	0	■
1	0	0	1	0	■	0	■
1	1	0	1	0	■	0	■

T1	T2	T3	T4	目標	重み	新たな クラス分類器 if T4=1 then Ob=1 else Ob=0	新たな 重み
1	0	1	1	1	■	1	■
1	0	1	1	1	■	1	■
1	1	1	1	1	■	1	■
1	1	1	0	0	■	0	■
1	0	1	0	0	■	0	■
1	1	0	1	0	■	1	■
1	0	0	1	0	■	1	■
1	1	0	1	0	■	1	■

T1	T2	T3	T4	Ob	クラス分類器 if T1=1 then Ob=0 else Ob=1	 if T3=1 then Ob=1 else Ob=0	 if T4=1 then Ob=1 else Ob=0	単純な 多数決
1	0	1	1	1	0	1	1	1
1	0	1	1	1	0	1	1	1
1	1	1	1	1	0	1	1	1
1	1	1	0	0	0	1	0	0
1	0	1	0	0	0	1	0	0
1	1	0	1	0	0	0	1	0
1	0	0	1	0	0	0	1	0
1	1	0	1	0	0	0	1	0

AdaBoost **は重みを使った多数決**

AdaBoost

入力

訓練データ $(\vec{x}_1, y_1), \ldots, (\vec{x}_N, y_N)$

初期の重み $w_i^1 = D(i) = 1/N$ for each i =1,2, …, N

WeakLearn：　エラー率が0.5**未満の**
クラス分類器を常に出力する学習アルゴリズム

T: **最終的に使うクラス分類器の個数**

各 $t = 1, ..., T$, について以下のステップを繰返す:

1: **各レコードの重みを正規化して各レコードの分布 p_i^t を計算**

$$p_i^t = w_i^t \Big/ \sum_{i=1}^{N} w_i^t$$

2: **WeakLearn を呼び出し次の条件を満たすクラス分類器 h_t を生成**

$$\varepsilon_t = \sum_{i=1}^{N} p_i^t \left| h_t(\vec{x}_i) - y_i \right| < 1/2$$

$$\left| h_t(\vec{x}_i) - y_i \right| = \begin{cases} 0 & h_t \text{が} \vec{x}_i \text{の正解を返す場合} \\ 1 & \text{otherwise} \end{cases}$$

3: **重みを更新　重みは正解だと軽くなり、不正解だとそのまま**

$$\beta_t = \varepsilon_t / (1 - \varepsilon_t) \qquad \beta_t < 1 \qquad w_i^{t+1} = w_i^t \beta_t^{1 - \left| h_t(\vec{x}_i) - y_i \right|}$$

出力: 最終のクラス分類器 h_f (h_t の重みつき多数決)

$$h_f(\vec{x}) = \begin{cases} 1 & \text{if} \quad \sum_{t=1}^{T}(-\log\beta_t)h_t(\vec{x}) \geq \sum_{t=1}^{T}(-\log\beta_t)\dfrac{1}{2} \quad \cdots(*) \\ 0 & \text{otherwise} \end{cases}$$

$\varepsilon_t \to 0, \quad \beta_t \to 0, \quad (-\log\beta_t) \to +\infty \qquad \varepsilon_t \to 1/2, \quad \beta_t \to 1, \quad (-\log\beta_t) \to 0$

定義 $\varepsilon = \sum_{\{i \mid h_f(\vec{x}_i) \neq y_i\}} D(i)$

最終クラス分類器 h_f の初期分布に対するエラー率

定理 $\varepsilon \leq \prod_{t=1}^{T} 2\sqrt{\varepsilon_t(1-\varepsilon_t)} = \prod_{t=1}^{T} \sqrt{1-(1-2\varepsilon_t)^2}$

$\varepsilon_t < 1/2, \quad 1-2\varepsilon_t < 1, \quad \sqrt{1-(1-2\varepsilon_t)^2} < 1$

証明のロードマップ

$$\varepsilon\left(\prod_{t=1}^{T}\beta_t\right)^{1/2} \le \sum_{i=1}^{N} w_i^{T+1} \qquad \text{補題 2}$$

$$\le \left(\sum_{i=1}^{N} w_i^{T}\right)\times 2\varepsilon_t \qquad \text{補題 3} \quad \sum_{i=1}^{N} w_i^{t+1} \le \left(\sum_{i=1}^{N} w_i^{t}\right)\times 2\varepsilon_t$$

$$\vdots$$

$$\le \prod_{t=1}^{T} 2\varepsilon_t \qquad \sum_{i=1}^{N} w_i^{1} = 1$$

$$\varepsilon \le \prod_{t=1}^{T}\left(2\varepsilon_t \times \beta_t^{-1/2}\right) = \prod_{t=1}^{T} 2\sqrt{\varepsilon_t(1-\varepsilon_t)}$$

$\beta_t = \varepsilon_t/(1-\varepsilon_t)$ に注意

補題 2　$\sum_{i=1}^{N} w_i^{T+1} \geq \varepsilon \left(\prod_{t=1}^{T} \beta_t \right)^{1/2}$

$\sum_{i=1}^{N} w_i^{T+1} \geq \sum_{\{i \mid h_f(\vec{x}_i) \neq y_i\}} w_i^{T+1}$　　h_f が誤るレコードの重みの総和

$= \sum_{\{i \mid h_f(\vec{x}_i) \neq y_i\}} \left(D(i) \prod_{t=1}^{T} \beta_t^{1-|h_t(\vec{x}_i)-y_i|} \right)$　　$w_i^{t+1} = w_i^t \beta_t^{1-|h_t(\vec{x}_i)-y_i|}$　より

$\geq \sum_{\{i \mid h_f(\vec{x}_i) \neq y_i\}} \left(D(i) \left(\prod_{t=1}^{T} \beta_t \right)^{1/2} \right)$　　補題 1 より

$= \left(\sum_{\{i \mid h_f(\vec{x}_i) \neq y_i\}} D(i) \right) \left(\prod_{t=1}^{T} \beta_t \right)^{1/2}$

$$h_f(\vec{x}) = \begin{cases} 1 & \text{if} \quad \sum_{t=1}^{T}(-\log\beta_t)h_t(\vec{x}) \geq \sum_{t=1}^{T}(-\log\beta_t)\dfrac{1}{2} \quad \cdots(*) \\ 0 & \text{otherwise} \end{cases}$$

補題1　$h_f(\vec{x}_i) \neq y_i$ ならば $\displaystyle\prod_{t=1}^{T}\beta_t^{1-|h_t(\vec{x}_i)-y_i|} \geq \left(\prod_{t=1}^{T}\beta_t\right)^{1/2}$

コメント　最終のクラス分類機で予測が失敗すると重みが十分に減らない

$h_f(\vec{x}_i)=1, y_i=0$ ならば：　$\sum_{t=1}^{T}(\log\beta_t)$ を(*)の両辺に加算.

$$\sum_{t=1}^{T}(\log\beta_t)(1-h_t(\vec{x}_i)) \geq \sum_{t=1}^{T}(\log\beta_t)\frac{1}{2}$$

さらに $1-h_t(\vec{x}_i)=1-\left|h_t(\vec{x}_i)-y_i\right|$ に注意

$h_f(\vec{x}_i)=0, y_i=1$ ならば：　$\sum_{t=1}^{T}(-\log\beta_t)h_t(\vec{x}_i) < \sum_{t=1}^{T}(-\log\beta_t)\frac{1}{2}$

$$\sum_{t=1}^{T}(\log\beta_t)h_t(\vec{x}_i) > \sum_{t=1}^{T}(\log\beta_t)\frac{1}{2}$$

さらに $h_t(\vec{x}_i)=1-\left|h_t(\vec{x}_i)-y_i\right|$ に注意

補題3 $\sum_{i=1}^{N} w_i^{t+1}$

$$= \sum_{i=1}^{N} w_i^t \beta_t^{1-|h_t(\vec{x}_i)-y_i|}$$

$$\leq \sum_{i=1}^{N} w_i^t (1-(1-\beta_t)(1-|h_t(\vec{x}_i)-y_i|))$$

$\alpha^\gamma \leq 1-(1-\alpha)\gamma,$
if $\alpha \leq 1$ and $\gamma = 0$ or 1.

$$= \sum_{i=1}^{N} w_i^t - (1-\beta_t)\sum_{i=1}^{N} w_i^t (1-|h_t(\vec{x}_i)-y_i|)$$

$$1-\varepsilon_t = \sum_{i=1}^{N} p_i^t - \sum_{i=1}^{N} p_i^t |h_t(\vec{x}_i)-y_i|$$
$$= \sum_{i=1}^{N} p_i^t (1-|h_t(\vec{x}_i)-y_i|)$$
$$= \sum_{i=1}^{N} w_i^t (1-|h_t(\vec{x}_i)-y_i|) \Big/ (\sum_{i=1}^{N} w_i^t)$$

$$= (\sum_{i=1}^{N} w_i^t)(1-(1-\beta_t)(1-\varepsilon_t))$$

$$= (\sum_{i=1}^{N} w_i^t) \bullet 2\varepsilon_t$$

$$1-\beta_t = 1-\frac{\varepsilon_t}{1-\varepsilon_t} = \frac{1-2\varepsilon_t}{1-\varepsilon_t}$$