最適相関ルールとその応用

福田 剛志

日本アイ・ビー・エム (株)
東京基礎研究所

1 はじめに

昨今のデータ入力技術の進歩・普及と大容量記憶装置の劇的な低価格化により，データの収集・集積は容易なこととなり，集積されたデータの規模はギガバイト単位からテラバイト単位へと成長を続けている．この文字通り山のようなデータはビジネスの対象である顧客や市場の性質を反映しているので，そこに内在する未知の規則やパターンを発見し，パフォーマンスの改善に役立てたいというのは自然な要求である．

そのための第一段階として，収集された生データをデータベースに格納して置けば，人間がデータベースに仮説・検証的な問い合わせを繰り返してデータを分析することで，目的をある程度達することが出来る．この分析的な問い合わせを効率的に処理する研究が OLAP (On-Line Analytical Processing [1]) の名の下に盛んに行われている．

さらに進んで，この人間が行う分析作業を出来るだけ自動化し，大規模データから規則やパターンを発見する「データマイニング」がデータベースと人工知能の境界分野として盛んに研究されている [2, 3, 4].

本稿では，相関ルールの数値属性への拡張を通じて最適相関ルールの概念を導入し，さらにその決定木・回帰木への応用について述べる．

2 相関ルール

例としてスーパーマーケットのキャッシュレジスタで収集されるデータに注目しよう．顧客が買い物かごに入れた商品のラベルを，店員がバーコードリーダを使って次々に読み込んでいく．このように収集されたデータは，どのような商品が同時に買われたかという事実の集まりであると考えることができる．そこから例えば「目玉商品 A と日用品 B を購入した顧客は，同時に高級品 C も高い確度で購入することが多い」という事実が判れば，

- A，B，C のセット商品を発売する．
- A や B の特売を行う際には C の在庫を増やしておく．
- 顧客の利便性を考えて，商品の配置を近づける．
- 逆に顧客に店内を長く歩き回るってもらうため，商品の配置を遠ざける．

などといった販売戦略を立てることができそうである．さらに目玉商品 A と日用品 B を購入した顧客の内，何パーセントが高級品 C を購入するか (後で定義する確信度) が判明すれば，目玉商品 A と日用品 B の売り上げ傾向から，高級品 C の売り上げをある程度予測することができるだろう．この事実は，

$$\{\text{目玉商品 A}, \text{日用品 B}\} \Rightarrow \{\text{高級品 C}\}$$

という式で表現できる．

一般に X, Y を集合として

$$X \Rightarrow Y$$

と記述される事実を相関ルール (association rule) と呼ぶ．

対象とする問題は小売店で売られる商品の併買関係に限る必要は無い．顧客の特徴 (年齢，性別，職業，趣味) の間の関係を調べるために用いても良いし，テキストマイニングでは，文書データを対象として，キーワードの間の相関ルールを求めることもある [5]．時間的に引き続いて起こる事柄の間の関係を「順序パターン (sequential pattern)」として発見する問題にも類似の考え方が応用できる [6].

「相関」は本来 “correlation” に対する訳語として用いられており，筆者らは当初 “association rule” に対して「関連ルール」とか「結合ルール」という訳語をあててきた．実際，correlation と association は区別すべき概念だが，どうやら相関ルールという語が一般に定着してしまったようなので，ここでもそれを採用する．

商品の集合を好き勝手自由に組み合わせれば，形式的には非常に多くの相関ルールを作ることができる．あまりに大量にあるので個々に調べていくことはできないし，その中で役に立つものはほんの僅かであるに違いない．ではどういう相関ルールに価値があるのだろうか．

価値ある相関ルールはある程度以上確からしいことが必要なのは当然だろう．相関ルール $X \Rightarrow Y$ の確からしさは，$Pr[Y|X]$ の推定値すなわち，商品の集合 X を購入した顧客数 a のうち，Y をも購入した顧客数 b の割合 b/a で計り，この値を相関ルールの確信度 (confidence) と呼ぶ．さらに，相関ルールが適用できるデータの量がある程度以上大きいことが重要である．なぜなら，あまりに僅かなデータにしか適用できない相関ルールは出番が少なく，役に立つ機会が少ないからである．適用できるデータ量は相関ルールの X と Y を同時に購入した顧客数 b の全顧客数 N に対する割合 b/N で図るのが普通で，この値を相関ルールのサポートと呼ぶ．そこで確信度とサポートがある程度以上大きい相関ルールが有効で，価値があると考えることにする．

価値の高い相関ルールを作るアイテム集合が予め分かっていれば，データベースに対する簡単な問い合わせで，相関ルールの確信度とサポート (あるいはその他の指標による価値) を知ることが出来る．商品集合毎に購入した顧客の数を数えて集計すればよいのだから，これは OLAP の得意とする演算である．しかし通常は，どのアイテムを組み合わせれば価値のある相関ルールが出来るかは事前には分からない．もし分かっていれば，そもそもマイニングの必要などないのであるから，ユーザが価値のありそうな相関ルールの候補を与えることを仮定するわけにはいかない．すべての相関ルールを調べて重要なものを選ぶという方法は，潜在的な相関ルールの数があまりにも多いため，役に立たない．そこで，自動的にデータベースから価値のある相関ルールを効率的に，しかも漏れなく発見する方法が必要となる．

IBM アルマデン研究所の Rakesh Agrawal らは 1994 年，ユーザが最小確信度と最小サポートを与えて，それ以上の確信度，サポートを持つ相関ルールをすべて発見する効率的なアルゴリズム “アプリオリ” (apriori) を発表した [7]．これは後に数多くの派生研究を生むこととなり，データマイニングの研究に火をつけたともいえる重要なアルゴリズムである．同研究所で開発された Quest システム [8, 9] はアプリオリを含む数々のデータマイニング手法を実装した世界初の本格的なデータマイニングシステムである．

アプリオリアルゴリズムは次のような単純な観察に基づいて，探索すべき相関ルールの枝刈りを行う．商品の集合 I を購入した顧客の全体に対する割合 (I のサポートと言う) を support(I) とすると，I を含む商品集合 $J \supseteq I$ のサポート support(J) は support(I) を超えることはない．従って，もし I のサポートがユーザの与えた最小サポートよりも小さければ，I を含むようなどんな商品集合も最小サポート以上のサポートを持つことはありえない．相関ルール $X \Rightarrow Y$ のサポートと確信度を求めるために必要な support(X) と support($X \cup Y$) は X と $X \cup Y$ を購入した顧客をそれぞれ数えれば求めることができるが，support($X \cup Y$) は相関ルールのサポートそのものなので，この値が最小サポートより小さい場合には，$X \cup Y$ とそれを含む商品集合は相関ルールを作るために用いる商品集合の候補から外すことができる．

注意しなければならないのは，データベースは膨大で主記憶には普通入らず，2 次記憶に置かなければならないことである．一方，相関ルールの数は通常は高々数千程度で，相関ルールだけは上手に主記憶内に保持したい．そこで候補となる商品集合を主記憶内に置き，顧客一人一人が購入した商品のリストを 2 次記憶から逐次的に読み出して，候補となっている商品集合がそこに含まれていれば購入顧客数を 1 増やすということを繰り返す．データベースのスキャンはコストがかかるので，その回数を減らすため 1 回のスキャンで複数の商品集合の候補を同時に処理する．オリジナルのアルゴリズムは，k 回目のスキャンで k 個の商品からなる候補をまとめて処理している．

ここで処理上のボトルネックとなるのは，候補となっているたくさんの商品集合の中から，各顧客が購入しているものを見つける部分である．この高速化には，ハッシュ木を使う方法による実装方法が用いられている．実装方法の詳細は，[10] にある解説や [11] に公開されている実装例を参照すると良いだろう．

さて，ひとつの相関ルールを取り出してみると，あまりにプリミティブな情報に見える．しかし人間が持っている知識や先入観と対比させたり，複数の相関ルールを比較することにより，相関ルールは役立つ知識となる．このため，一度にたくさんの相関ルールを見比べるて解釈することができるような視覚化ツール (例えば [12]) は，相関ルールを活用する上で非常に役に立つ．

3 最適数値属性相関ルール

これまで見てきた相関ルールは，アイテムの集合であるトランザクションが集まったデータベースを対象として，そこからトランザクションにアイテムが出現するか否か (の連言) だけを取り扱っており，その表現力には限界がある．例えば表 1 のリレーションが入力として与えられたとする．このような場合でも「属性名 = その値」を一つのアイテムだと思って取り扱えば，従来のアルゴリズムで相関ルールを求めることができるそうである．しかし数値属性の場合は，以下の理由でそう簡単には行かない．

数値属性は値の順序に意味があり，定義域が大きいことが多い．例えば，銀行の顧客の「預金残高」は 0 円から数十億

表 1: 銀行顧客

顧客番号	年収	預金残高	年齢	...	職業	サービス A
100	600	120	30		公務員	no
200	1000	500	40		会社員	yes
300	100	500	35		主婦	no
400	400	50	24		会社員	yes
⋮	⋮	⋮	⋮		⋮	⋮

円の間の値を 1 円刻みで取りうる．従って，カテゴリ属性で考えたような「預金残高 $= v$ 円」という形式の条件は数十億個も存在し，そのほとんどがごく小さなサポートしか持たないだろうから，相関ルールに使う条件としてあまり意味がない．一方，「預金残高が近い銀行顧客は似通った振る舞いをするだろう」という，数値属性に関するデータの性質の連続性を仮定すると，「$v_1 \leq$ 預金残高 $\leq v_2$」のような，数値の区間を条件として相関ルールに利用することができれば役に立ちそうである．このような区間を表す条件一つ一つをアイテムとして取り扱うことができれば良いのだが，例えば預金残高の区間は (その始まりと終わりがそれぞれが何十億通りあるから) 何十億の 2 乗個くらい存在するので，このアイデアは現実的でない．

そこでまず，数値属性の区間を使った最も簡単な次の形式の相関ルールに注目しよう:

$$(A \in [v_1, v_2]) \Rightarrow C \tag{1}$$

ここで，A は数値属性，$v_1 \leq v_2$ は A の定義域中の値，C はある与えられた条件である．この形式の相関ルールを，1 次元数値属性相関ルール [13, 14, 15] と呼ぶ．

例えば表 1 のリレーションが与えられたとする．各タプルは一人の顧客を表しており，年収，預金残高，年齢，職業，サービスを利用しているか否かなどの属性がある．このデータから「預金残高が 20 万以上 150 万以下の顧客は高い確率でサービス A を利用する」という事実は，

$$(\text{預金残高} \in [20\text{万}, 150\text{万}]) \Rightarrow (\text{サービス A} = \text{yes}) \tag{2}$$

という相関ルールとして表現でき，この相関ルールによってサービス A を売り込むべき顧客層が浮かび上がってくる．

区間を予め決めてしまえば，簡単なデータベース問い合わせで，この相関ルールの確信度とサポートを求めることが出来る．具体的には，例えば式 (2) の相関ルールの場合データベースを 1 度スキャンして，顧客の総数 N，預金残高が 20 万以上 150 万以下の顧客の人数 s と，その中でさらにサービス A を利用している顧客の人数 h をそれぞれ数えれば，この相関ルールの確信度が h/s，サポートが h/N として計算できる．

今「20 万以上 150 万以下」という区間を天下り的に与えたが，実際にはサービス A を利用する顧客を特徴づけるような預金残高の区間自体を見つけ出したいのである．ここで問題となるのは，預金残高のような値域の大きい数値属性の区間の取り方はたくさんあると言うことである．しかも，単に確信度が高ければ良いと言うことではない．例えば，サービス A を利用しているある顧客ただ一人だけの預金残高を含むようなごく狭い区間があったとすると，その区間を使った相関ルールの確信度は 100 パーセントとなるが，サポートは非常に小さく，このような相関ルールに意味があるとは言えない．逆に，区間を大きくすればサポートは 100 パーセントになるまでいくらでも大きくできるが，確信度は顧客全体がサービス A を利用する人の割合に近くなってしまう．では，どのような区間が数値属性相関ルールの目的に良く適っているだろうか．

最適確信度相関ルール　相関ルールはある程度の大きさのサポートを持っているべきであるから，ユーザは条件「預金残高 $\in [v_1, v_2]$」のサポート (区間のサポートと言う) の最低値を与えることにする．サポートがその最低値以上であるような区間の中で，相関ルールの確信度が最も高くなる区間は，サービス A を利用する顧客を最も良く特徴づけていると言えるだろう．そこで，このような区間を確信度最大化区間と呼び，それを用いた相関ルールを最適確信度相関ルールと呼ぶ．

例えば，顧客にダイレクトメールを発送するのだが，予算が限られていて全体の顧客の 10 パーセントにしかメールを出すことができないとする．このような時は

$$\text{収入} \in [v_1, v_2] \Rightarrow \text{商品に興味を持つ} \tag{3}$$

という相関ルールを考て，最小サポート 10%とした時の最適確信度相関ルールを求めればよい．収入が確信度最大化区間 $[v_1, v_2]$ に入っているような顧客は，ダイレクトメールを発送すれば最も高い確率で商品に興味を持つ 10 パーセントの顧客層である．

最適サポート相関ルール　最適確信度相関ルールと双対の概念として，ユーザーが確信度の最低値を与え，その最低値以上の確信度を持つ相関ルールを作るような区間の中で，最大のサポートを持つ区間をサポート最大化区間と呼び，それを用いた相関ルールを最適サポート相関ルールと呼ぶ．

先のダイレクトメールの例で，今度はメールを受け取った顧客が反応する割合がある値 τ 以下だと赤字が出てしまうとしよう．1 枚のダイレクトメールのコストとダイレクトメールに反応した顧客から得られる利益の割合から τ の値が決ま

る．赤字がでない限り出来るだけ多くの顧客を取り込みたいとすると，式 (3) にある相関ルールの最小確信度を τ とした時の最適サポート相関ルールを求めれば，反応率が τ 以上であるような最大の顧客層を見つけ出したことになる．

最適ゲイン相関ルール　もう一つ重要な最適相関ルールとして最適ゲイン相関ルール (optimized gain association rule) がある．これはダイレクトメール 1 通のコスト c と，反応した顧客一人に対する売り上げ s が決まっていると仮定したときの利益 (ゲイン; gain),

$$s\times(\text{区間のサポート})\times(\text{確信度})-c\times(\text{区間のサポート}) \quad (4)$$

を最大化するような区間を用いた相関ルールである．

確信度，サポート，ゲインを最大化する区間を求める素朴な方法は，全ての区間を列挙して，最大の確信度，サポート，ゲインとなる区間を選ぶことである．この方法の最悪計算量はタプル数の 2 乗に比例することになり，データベースが巨大なときには実用的ではない．[13, 14, 15] は，データベースが予め数値属性に従ってソートされていれば，計算幾何学的方法を用いて，計算量がレコード数に比例するようなアルゴリズムを示した．

4　2 次元最適数値属性相関ルール

前節では，数値属性を 1 つだけ用いた相関ルールを考えたが，1 つだけの属性を用いたルールでは，必ずしも結論条件を特徴づけるのに十分とは言えない．複数の数値属性を前提条件に用いるような相関ルールを求めたいと考えるのが自然である．

まず次のような形式の相関ルールを思い付く．

$$(\text{預金残高} \in [20\text{万}, 150\text{万}])\wedge(\text{年齢} \in [25, 35]) \Rightarrow (\text{サービス A} = \text{yes}) \quad (5)$$

この相関ルールは，新しい条件「年齢 $\in [25, 35]$」を用いることで，1 つしか数値属性を用いていなかった式 (2) より強くサービス A を利用する顧客層を特徴づけている．このような 2 つの数値属性の区間の直積は，2 つの数値属性が作る平面上の矩形領域である (図 1 左 参照)．矩形領域は制限が強く，例えば対角線上に分布するようなデータや曲がった領域に分布するようなデータを上手く表現することが出来ない．

一般に，数値属性 A, B とそれらが張る平面上の (矩形に限らない) 領域 R を用いて，

$$((A, B) \in R) \Rightarrow C \quad (6)$$

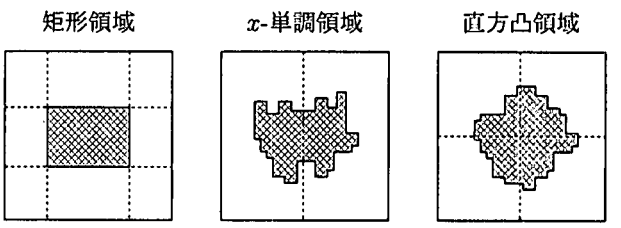

図 1: 領域の例

という形式で書かれる相関ルールを，2 次元数値属性相関ルール [16] と呼ぶ．この形式の相関ルールに対しても，1 次元の場合と同様に最適確信度相関ルール，最適サポート相関ルール，最適ゲイン相関ルールを自然に定義することが出来る．

最適確信度・サポート相関ルールに用いる最適化領域を求めるために，先ず 2 つの数値属性が張る平面を適当な粒度のグリッドに分解し，グリッド上のピクセルをつなぎあわせて領域を作ることにする．データベースを数値属性ごとにソートして，各ピクセルにタプルが 0 または 1 つだけ入るような細粒度のグリッドを作ることも可能だが，20×20 から 1000×1000 程度のグリッド数で十分な精度が得られるので，3 章と同様にバケット分割しておけば良い．以後，バケット分割が既に計算され，各ピクセル $G(i, j)$ に含まれるタプル数 $u_{i,j}$ とその中で結論条件 C を満たすタプル数 $v_{i,j}$ が求まっていることを前提とする．

領域を矩形に限ると，最適化領域を求める計算量は 3 節で説明した 1 次元最適相関ルールの問題に還元する方法が知られている限り最善で，ピクセルの個数を n として $O(n^{1.5})$ である．

任意の連結領域の中で最適な領域を求めようとすると，NP 困難となってしまう [16]．そこで矩形よりは柔軟で，しかも計算しやすい領域の種類 (領域族; region family) に制限する必要がある．

領域族に x 単調領域 (x-monotone region; x 軸に垂直な直線との交わりが 1 つの区間か空であるような連結領域) を採用した場合，x 単調領域の中での最適確信度・サポート領域は $O(nN)$ (N はタプル数) で求めることが出来るが，$O(n \log N)$ では P=NP でない限り求めることが出来ない [16]．タプル数は数千万以上にも及ぶかもしれず，この計算量は実用的でない．幸い，画像処理の分野でピクセルに割り当てられた濃淡度情報から x 単調領域を切り出す高速のアルゴリズムが知られており，この方法で最適ゲイン領域を $O(n)$ 時間で求めることができる．最適確信度，サポート領域も最適ゲイン領域を計算する仕組みを利用して，近似解を $O(n \log N)$ 時間で求めることが出来る [16]．

x 単調領域は自由度が高いので，領域の生成に用いた学習データに対する最適な領域がギザギザした不自然な形となることがある．これは，学習データに含まれる本来は意味のない僅かなノイズを拾っているためで，学習データに対しては良い結果を示しても，将来の未知データに対する良い予測を与えない．そこで，領域族として領域の境界がより滑らかになる直方凸領域 (rectilinear convex region; x 単調かつ y 単調な連結領域) を用いる方法が提案されている．最適ゲイン直方凸領域はダイナミックプログラミングにより，$O(n^{1.5})$ 時間で求めることが出来る [17, 18]．x 単調領域の場合と同様に，これを利用して最適確信度，サポート領域の近似解 $O(n^{1.5}\log N)$ で求めることが出来る．直方凸領域を用いた相関ルールは，未知のデータに対しても安定した予測を与えることが実験的に確認されている．

数値属性を用いた条件に場合に限らず一般に，相関ルールの結論条件を最もよく特徴付ける前提条件を持つような相関ルールを，最適相関ルールと考えることができる．文献 [19, ?, ?] はカテゴリ属性を用いた最適相関ルールを効率的に求める方法について論じている．

5 決定木・回帰木

これまで見てきた相関ルールは単純で理解しやすかったが，それゆえ，単独では複雑な実世界を表現するには力不足であることが多い．

例えば，たくさんの企業の財務データがデータベース化されているとする．データベース中の企業の中には業績不振に陥り，倒産したものも含まれている．このデータから経験的に倒産しやすい企業の財務内容を学習し，新たな財務データからその企業が将来倒産する可能性が高いかどうかを判定できれば，金融業務でのリスク管理に有効である．倒産企業を特徴づける相関ルールを求めれば，

$$(\text{自己資本比率} = \text{低}) \Rightarrow (\text{倒産} = \text{yes}) \qquad (7)$$

といった相関ルールがいくつも得られる．このような相関ルールは倒産する確率の高い企業を出来るだけ捉えようとしているのだが，倒産するか否かは複雑な要因で決まるので，単独の相関ルールだけではそれを十分高い精度でモデル化することは出来ない．

そこで，次のような 2 分木構造を作成する．木の各ノードはデータベースのレコードに対するテストで，テストの結果 yes となったレコードは左の枝へ，no となったレコードは右の枝へ進む．テストは相関ルールの前提条件に相当し，先の例では，そのノードにやってくるデータの中で倒産する確率の高い企業を左へ，低い企業を右へと振り分ける．各レコードは何段階かのテストの後，最終的に倒産・非倒産のラベルがつけられている木の葉に到着し，そのラベルにしたがって判定される．このような木構造で，学習の対象となる概念 (目的属性) が離散値を取る場合を決定木 (decision tree) と呼ぶ．これに対して，目的属性が株価のような数値を取る場合，回帰木 (regression tree) と呼ぶ．

一般に学習されるモデルは単純であるほうが良いので，木の高さ (1 つのレコードに対するテストの最大数) が小さく，木の葉数 (分類パターンの数) 小さいことが望ましい．どのようなテストをどういう順番で選ぶかによって，木の大きさや形は大きく変わってくる．最小の決定木を構成する問題は NP 困難 [20] と言われているため，何らかの近似的な解法が必要となる．そのための 1 つのヒューリスティックスとして，決定木の場合エントロピ [21] や gini インデックス [22]，回帰木の場合平均自乗誤差 [23, 24] といったデータの偏りの度合いを評価する指標を用いて，木の根から順番に，各時点で最も良く倒産企業と非倒産企業を分離するようなテストを貪欲に選ぶ方法が有名である．

2 次元数値属性相関ルールを用いた決定木　決定木・回帰木を構成するテストには，離散属性の場合は “離散属性 $\in$ 値域の部分集合”，数値属性の場合は “数値属性 $<$ 値” と言った，属性を 1 つ使ったテストを用いるのが普通である．この方法は，属性が少数の離散値を取る場合や，数値属性でも各属性の独立性が高い場合には良い木を導く．しかし，数値属性間には相関があることが普通であり，そのような場合には上手く行かないことが指摘されている [25]．例えば，健康な人の身長と体重には，次の関係が成り立つことが知られている．

$$18.7 * (\text{身長})^2 < \text{体重} < 25.3 * (\text{身長})^2 \qquad (8)$$

1 つの属性だけを用いたテストではこのような関係を簡単に表現することは出来ず，結果として決定木が非常に大きくなってしまう．

そこで，データの偏りの指標を最適化する領域をテストに用いる方法が提案されている [26, 24]．例えば，(自己資本比率, 有利子負債比率) $\in R$ をテストとして用いると，属性間に相関があっても，個々の属性を別々に使ったテストに比べて，1 回のテストで倒産企業と非倒産企業を良く分離することが出来る．その結果，決定木の大きさは従来方法に比べて劇的に小さくなる．木が小さくなれば，人間が理解することが容易になるので，要因分析の有効な手段ともなり得る．データの偏りの指標を最適にする 2 次元領域を求める問題は，2 次元のピクセル数を n，レコード数を N として，x 単調領域族の場合平均 $O(n\log N)$ 時間 [26]，直方凸領域族の場合平均 $O(n^{1.5}\log N)$ 時間 [17] で計算できる．

倒産確率算出システム 日本IBMはは，最適相関ルールを用いて，過去の数千件の企業の財務データから学習した決定木などのモデルを用いて，未知の企業の財務データから将来の倒産確率を求めるソフトウェアを開発した．このデータベースは，実際の企業のある時点での "自己資本比率", "税引前売上利益率" などといった約80種類の数値属性と，その後 1, 2, 3 年の間に実際その企業が倒産したか否かという情報からなっている．このような財務データの各属性は互いに相関しているので，2次元数値属性相関ルールを用いない従来の方法に比べて，デフォルトメータの2次元数値属性相関ルールを用いた決定木は，大きさが小さい上，高い精度を達成している [19].

参考文献

[1] E. F. Codd, S. B. Codd, and C. T. Salley. Beyond decision support. *Computerworld*, Vol. 27, No. 30, July 1993.

[2] G. Piatetsky-Shapiro and W. J. Frawley, editors. *Knowledge Discovery in Databases.* AAAI Press, 1991.

[3] Michael Stonebraker, Rakesh Agrawal, Umeshwar Dayal, Erich J. Neuhold, and Andreas Reuter. DBMS research at a crossroads: The vienna update. In *Proceedings of the International Conference on Very Large Data Bases*, pp. 688–692, 1993.

[4] Peter Cabena, Pablo Hadjinian, Rolf Stadler, Jaap Verhees, and Alessandro Zanasi. *Discovering Data Mining: from Concept to Implementation.* Prentice Hall, September 1997. 河村 佳洋, 福田 剛志 (監訳): データマイニング活用ガイド概念から実践まで, トッパン, 1999.

[5] 川原稔, 河野浩之, 長谷川利治. 文献データベース情報検索に対するデータマイニング技術の適用. 情報処理学会論文誌, Vol. 39, No. 4, pp. 878–887, 1998.

[6] Ramakrishnan Srikant and Rakesh Agrawal. Mining sequential patterns: Generalizations and performance improvements. In *Proceedings of the International Conference on Extending Database Technology*, 1996.

[7] Rakesh Agrawal and Ramakrishnan Srikant. Fast algorithms for mining association rules. In *Proceedings of the International Conference on Very Large Data Bases*, pp. 487–499, 1994.

[8] Rakesh Agrawal, Andreas Arning, Toni Bollinger, Manish Mehta, John Shafer, and Ramakrishnan Srikant. The Quest data mining system. In *Proceedings of the International Conference on Knowledge Discovery and Data Mining*, August 1996.

[9] http://www.almaden.ibm.com/cs/ quest.

[10] 福田剛志, 森本康彦, 徳山豪. データマイニング. データサイエンス シリーズ, No. 3. 共立出版, 2001.

[11] http://fuzzy.cs.uni-magdeburg.de/ ~borgelt/.

[12] 福田剛志, 森下真一. 相関ルールの可視化について. 電子情報通信学会技術研究報告 95-81, pp. 41–48, May 1995.

[13] Takeshi Fukuda, Yasuhiko Morimoto, Shinichi Morishita, and Takeshi Tokuyama. Mining optimized association rules for numeric attributes. In *Proceedings of the ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, pp. 182–191, June 1996.

[14] 福田剛志. 数値属性の最適結合ルールを発見する効率的なアルゴリズム. 情報処理学会論文誌, Vol. 37, No. 6, pp. 945–953, June 1996.

[15] Takeshi Fukuda, Yasuhiko Morimoto, Shinichi Morishita, and Takeshi Tokuyama. Mining optimized association rules for numeric attributes. *Journal of Computer and System Sciences*, Vol. 58, No. 1, pp. 1–15, February 1999.

[16] Takeshi Fukuda, Yasuhiko Morimoto, Shinichi Morishita, and Takeshi Tokuyama. Data mining using two-dimensional optimized association rules: Scheme, algorithms, and visualization. In *Proceedings of the ACM SIGMOD Conference on Management of Data*, pp. 13–23, June 1996.

[17] Kunikazu Yoda, Takeshi Fukuda, Yasuhiko Morimoto, Shinichi Morishita, and Takeshi Tokuyama. Computing optimized rectilinear regions for association rules. In *Proceedings of the International Conference on Knowledge Discovery and Data Mining*, August 1997.

[18] 依田邦和, 福田剛志, 森本靖彦, 森下真一, 徳山豪. 数値データからの直方凸領域結合ルール発見. 情報処理学会研究報告 97-68, pp. 17–24, July 1997.

[19] Yasuhiko Morimoto, Takeshi Fukuda, Yasuhiko Morimoto, Kunikazu Yoda, and Takeshi Tokuyama. Algorithms for mining association rules for binary segmentations of huge categorical databases. In *Proceedings of the International Conference on Very Large Data Bases*, August 1998.

[20] L. Hyafil and R. L. Rivest. Constructing optimal binary decision trees is NP-complete. *Information Processing Letters*, Vol. 5, pp. 15–17, 1976.

[21] J. Ross Quinlan. Induction of decision trees. *Machine Learning*, Vol. 1, pp. 81–106, 1986.

[22] Manish Mehta, Rakesh Agrawal, and Jorma Rissanen. SLIQ: A fast scalable classifier for data mining. In *Lecture Notes in Computer Science, Proceedings of the Fifth International Conference on Extending Database Technology*, Vol. 1057, pp. 18–32. Springer, 1996.

[23] Leo Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees.* Wadsworth, 1984.

[24] Yasuhiko Morimoto, Hiromu Ishii, and Shinichi Morishita. Efficient construction of regression trees with range and region splitting. In *Proceedings of the International Conference on Very Large Data Bases*, August 1997.

[25] J. Ross Quinlan. *C4.5: Programs for Machine Learning.* Morgan Kaufmann, 1993.

[26] Takeshi Fukuda, Yasuhiko Morimoto, Shinichi Morishita, and Takeshi Tokuyama. Constructing efficient decision trees by using optimized association rules. In *Proceedings of the International Conference on Very Large Data Bases*, pp. 146–155, 1996.