知 能 と 複 雑 系 128−16
（２００２．５．２４）

XML変形を用いた前処理の事例研究

山田　有吉　木村亮　矢野幸司　沼尾　正行

東京工業大学大学院　情報理工学研究科　計算工学専攻

yukichi@nm.cs.titech.ac.jp

概 要

データマイニングの全プロセスの中で、前処理に要するコストは６０％を占めるといわれている。前処理の一部を支援するアプリケーションは存在しているが、それらを用いて正しく処理するためには、緻密な計画とデータの整合性維持が必要となる。そこで、本研究室ではXML形式のデータを視覚化し、ユーザとのインタラクションによって前処理を行っていくシステムを提案した。また、多くのデータ操作を必要とする前処理プロセスの中で、システムの自動化は大きなテーマであると考えられる。そこで、実際に医学データ[1]に本システムを適用する際に有効であると考えられるユーザ支援手法とその適用事例を紹介する。

Studies of preprocessing using XML transformation

Yukichi Yamada, Ryou Kimura, Kouji Yano, Masayuki Numao

Department of Computer Science, Tokyo Institute of Technorogy

yukichi@nm.cs.titech.ac.jp

Abstract

Datamining requires huge data, which takes a long time to be preprocessed. Although each element of preprocessing is simple, it tends to be quite complecated and it is hard to construct the whole plan. To reduce the load, we propose an interactive and dynamic planning tool for preprocessing, named TransX. This system is based on XML, which enables to visualize the process by using a treelike notation and it allows user to change data easily and understandably. We propose some methods using TransX for semi-automatically executing function, which preprocess especially for medical data set.

1　はじめに

データマイニングではその解析アルゴリズムとして、相関ルール、決定木、クラスタリング、ニューラルネットワーク、遺伝アルゴリズムなど多くのものがあるが、これらの解析アルゴリズムに大量に蓄積されているデータを適用するためには、何らかの前処理が必要となる。前処理には構造の変形や値の標準化などが含まれるが、これらの作業は事例によって処理が異なり、また経験の求められる複雑な作業であるので、熟練した専門家によって処理される必要がある。そのためにデータマイニングではその処理コストの６０％が前処理に費やされていると言われている[2]。

しかし、前処理に特化したシステムや、前処理を専門に扱った研究というのはあまり盛んに行われてい

ないのが実情である。現在、前処理の自動化という観点では、属性若しくはレコードの取捨選択を学習によって自動化する研究 [3] や、前処理をおこなわないまま結果を導出する研究 [4] があるが、現時点で実際に前処理を行う場合は、単純だが有効性が明らかなものを人間であるオペレータが計画を立てて多数組み合わせている。

そこで本研究室では、前処理に特化したシステムである TransX を構築した [5]。このシステムは、前処理で扱うデータをすべて XML 形式 [6] に統一し、それを木構造として可視化することにより、ユーザにより理解しやすい形でデータを表示している。また、データ構造の変更をより容易に、ユーザをサポートしていくシステムとして実現している。本論文では、このシステムと、実際に医療データをもとにデータマイニングを行っていく際にユーザを支援していく半自動化手法について述べる。

2 従来の前処理の問題点

解析アルゴリズムに対して入力するデータの構造は、一部グラフ構造など他の構造をとる場合もあるがほとんどがフラットな表形式のデータ構造をとる。

従って、現時点では前処理に用いるツール、アプリケーションとして表形式のデータ、及び表の関係を扱うことができる関係データベースが用いられる。関係データベースは大量のデータを高速に処理することができる。しかし、関係データベースを用いた前処理では、表同士の関係の生成や修正などに大きなコストがかかる。

さらに、実際に前処理を行う上では、バックトラックの管理が必須となる。通常前処理では明確にそのゴールが決まっておらず、前処理を行ったデータを観察したり、解析を行ってみたりといった作業を行わないと、その前処理への評価が得られないことが多いため、何度も異なった方法で前処理を行う必要があるからである。

以上をまとめると、関係データベースよりも強力なデータ構造を持ち、バックトラックの容易性を実現する処理系が必要となる。

3 TransX システム

3.1 XML とユニットツリー

本システムでは、主にデータ構造の変形を、ＸＭＬ変形を用いた処理として実現する。この際に、自動的な前処理が可能となるよう配慮する。つまり、変形の単位を設定し、それをフィルタと呼ぶ。ユーザはこのフィルタと、フィルタ群から生成されるフィルタパスをデータに随時適応していくことによりデータ操作を行うことができる。

また、フィルタ数を軽減し、操作を簡易化するために、ＸＭＬ全体を一度に把握が可能なユニットツリーと呼ばれる構造で処理する。ユニットツリーは、そもそもデータマイニングにおいてデータひとつひとつの内容はあまり重要ではなく、全体が表す情報が重要であることに着目し、文書実体から見て同一の階層にある同一の名前を持つ要素を同一とみなす構造である。ＸＭＬとユニットツリーを図 1 に示す。

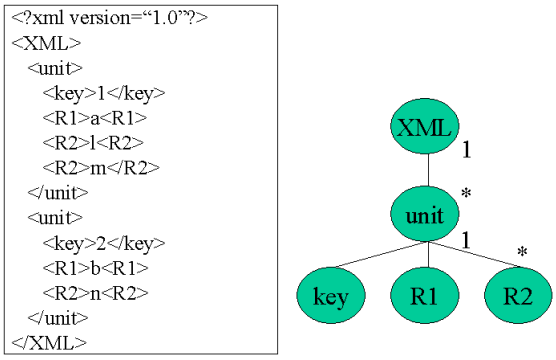

図 1: XML とユニットツリー

3.2 フィルタ

本システムにおける、ユニットツリー中のノードに対する変更操作の単位、それに付随して起きるXML の要素に関する変更操作の単位をフィルタと呼ぶ。このフィルタを逐一保存することで、バックトラックを可能にし、フィルタに対して重み付けを行うことで、フィルタの自動構成をしようとしている。フィルタの種類としては、作成・削除・移動・名前変更・結合の５種類を用意している

4 構成

システム全体の構成を図 2 に示す。

入力には表、関係データベース、テキスト、及びXML などあらゆる入力が考えられるが、それらはすべて単純なプログラムによって XML に変換されてから本システムに投入される。実際に本システムではシステム内に CSV (Comma Separated Value) から XML への簡単な変換プログラムを実装している。

入力された XML ファイルは、JAXP によって解析され、システム内部で DOM として表現される。DOM は XML と同義であるオブジェクトツリーであり、DOM API を用いて入力 XML に対応するユニットツリーが生成される。

このユニットツリーを見ながら、利用者は Web ブラウザ上に用意されたインターフェースを用いてフィルタの組合せであるフィルタパスを構成していく。ユニットツリーに対しては、フィルタパスは即座に適用され、利用者はユニットツリーの状態を見ながら、前処理を選択していく。

ある程度前処理が進んだところでフィルタパスを適用した XML を生成し、その XML ファイルを解析アルゴリズムに入力させることができる。解析結果は Web ブラウザ上で閲覧、またはファイルとして取り出すことができ、それらの結果を見てフィルタパスの修正を行う。システムの概観を図 3 に示す。

また、これらの操作時のフィルタパスは自動的に保存されており、利用者は重み付けされて自動的に提案されたフィルタパスを選択することが可能である。

以上の操作を繰り返し行うことで、より興味深い結果を得られる前処理を求めていく。

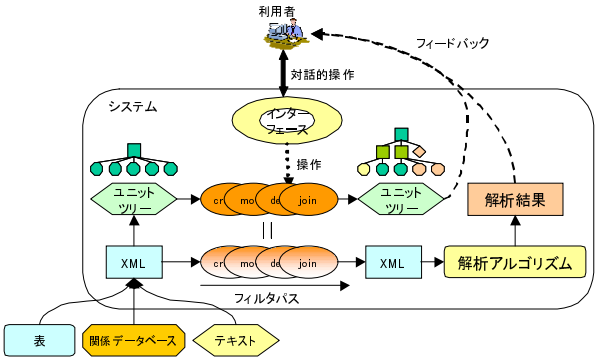

図 2: システム全体の構成

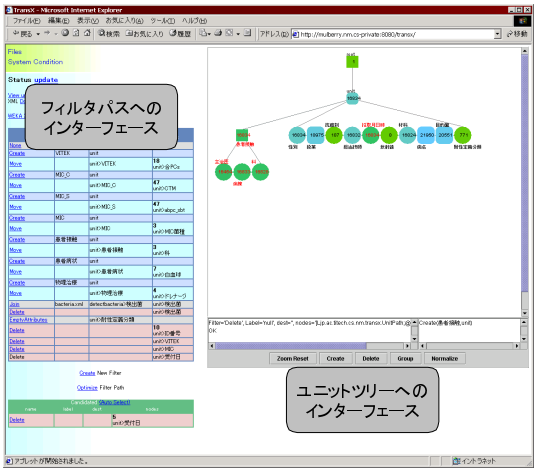

図 3: ユニットツリーとフィルタパス

5 TransX の改良

TransX はデータを木構造として、ユーザに認識しやすい形で可視化してくれるシステムである。し

かし、データマイニングで使用するデータセットは膨大であり、ユーザがその構造を把握し、処理するにはやはり多くの時間と労力を要する。そこで、フィルタパスを適用し、より認識しやすい木構造を提示してくれる自動化が必要となると考えられる。そこで今回、よりユーザフレンドリーなシステムに向けて、煩雑なデータ構造を自動的に再構成してくれる機能を追加した。
エントロピー計算から得られた結果をもとに自動的に木をマージし、元データの構造をより忠実に木構造として提示することを可能にした手法 [7] と、医療データの特性からある種のクラスタリングにより、各検査の関連性を木構造に反映する手法 [8] である。今回行われたこの 2 点の改良点とその出力結果に関する考察を述べる。

5.1 マージ

前処理の重要な作業の一つに、表の再構成が挙げられる。関係データベースを用いる場合には正規化が必要となるし、マイニングアルゴリズムにデータを投入する際においてもデータ形式をあわせる必要がある。しかしデータの再構成は、その意味内容やマイニング結果に影響を与えない変形であっても、ユーザが直感的に理解する意味合いにおいて変化すると考えられる。TransX はユーザフレンドリーな視覚化ツールとして考案されているため、ユーザが直感的に意味をグラフから読み取るということが重要となる。そこで、表の表現形式にとらわれず常に同様のグラフを提示する手段として、次の手法を提案する。

5.1.1 手法 1

図 4 における表 1 と表 2 は、データとしては同じ意味内容であるが、形式は異なっている。そのため、それぞれの表をそのデータ構造からユニットツリーに変換すると、矢印によって導かれるように互いに異なる木構造として表現されてしまう。これはユーザにとって非常に分かりにくい構造であり、このような多数の表現形式を一つに統一する必要がある。そこでマージを行う。
具体的には、同一 key である 1 をひとつの親ノードとしてまとめ、その下の属性についてはユニットツリーの特性からそれぞれを単一ノードとして統合する。このようにして、右のグラフは左のグラフのように変形され、表現形式が統一される。また、TransX はマイニングアルゴリズムとして java で記述された決定木である waka を内蔵している。データをマイニングアルゴリズムに適応する際には、データ形式に制限がかけられることが多いが、決定木に関しては、表 1 の属性 R2 に見られるような一対多の関係を許していない。マイニングアルゴリズムに適応するために行われる表の再構成には通常多くのコストを要するが、表現形式を統一することによってこの問題は解決され、XML データの書き換えという少ないコストで処理することが可能となる。また、実際に決定木に適用する際には、表 3 の出力を使用する。

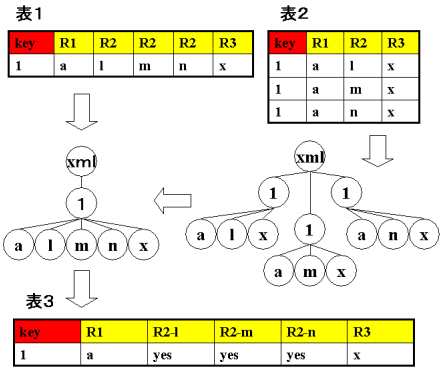

図 4: 表の再構成

5.1.2 手法 2

次に、より複雑な構造をもったユニットツリーに対するマージの方法について述べる。通常データマイニングで使用されるデータは属性間の関係性が

密であり、先に述べたような手法では木構造における意味内容が十分に反映されないというケースが考えられる。この問題を解消するために、平均情報量（エントロピー）を使用する。
平均情報量を元に、決定木作成に使用される情報利得比を算出することにより、マージされる確率が高いと思われるノードを選択し、そのノードに対してマージを繰り返すことにより、矛盾の少ないと思われる木構造を形成していく。
また、この平均情報量を用いたマージは、各属性間（図 4 における R1、R2）の出現値の差が大きく異なるという今回使用したデータの特性にも拠っている。

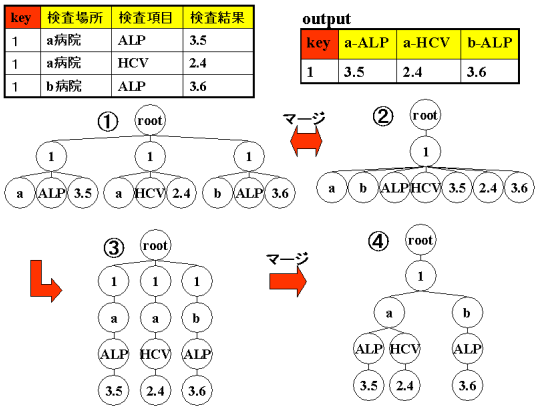

図 5: マージ

次に、実際にマージを行っていく過程を具体的に説明する。
図 5 の左上の表は、医療データにおける代表的な例である。この表をユニットツリーとしてあらわすと、図中 (1) のようになる。この場合平均情報量がもっとも少ないのは key であるから、ノード 1 がマージされて (2) になる。しかし、全ての属性が並列になることに矛盾を感知し、(1) に戻る。次に、平均情報量の順に key を除いた各属性に対してソートを行う (3)。そしてその中で平均情報量の少ない属性である「検査場所」について再度マージを行い、最終的に (4) の形になって終了する。出力は右上の表になる。
この結果から、元の表における行と列の属性間の関係を保持しつつ、かつあいまいさを取り除いた形でユニットツリーを再構成していることが分かる。

5.2 クラスタリング

今回使用したデータは、主に検査項目や検査結果等がリストされた医療データである。このようなデータには、ある臓器または疾患の状態を知りたい時に行われる検査に大きな偏りがあり、その結果同じタイミングで行われる傾向が強い検査項目群が多数存在する。このような関係性はマイニングをする上で自明な関係として不要なデータとなることが多く、それらをカテゴライズし、削除することが必要になると考えられる。

使用したデータを模式的に表にまとめた例を図 6 に表す。このように、データには属性の分布から明らかに関連性を見出せる検査項目が多く発見できる。このような項目群を、評価関数を基準にしてカテゴライズし、一連の木構造としてユーザに認識しやすい形として提示する。

MD	検査日	ALB	ALP	CL	COT	CPT	K	NA	PL
1	19811111	上限異常	正常値		上限異常	上限異常			
1	19811223	正常値	正常値		上限異常	上限異常			
1	19820127	上限異常	正常値		上限異常	上限異常			
1	19820210	正常値	正常値	正常値	上限異常	上限異常	正常値	正常値	
1	19820303	正常値	正常値		上限異常	上限異常			
1	19820512	正常値	正常値		上限異常	上限異常			正常値
1	19820811	正常値	正常値		上限異常	上限異常			正常値
1	19821003	上限異常	正常値	正常値	上限異常	上限異常	正常値	正常値	
1	19821006	正常値	正常値		上限異常	上限異常			正常値
1	19821208	正常値	正常値		上限異常	上限異常			正常値
1	19830720								
1	19830818	正常値	正常値	正常値	上限異常	上限異常	正常値	正常値	正常値
1	19830818								
1	19830907	上限異常	正常値		上限異常	上限異常			正常値

図 6: 医療データ

空データを含む属性同士のカテゴライズに使用した評価関数を次に示す。

R : 全レコードの数
Ra, Rb : それぞれ要素 A,B にのみに実データが存在するレコードの数
Rab : 要素 A,B の実データが共に存在するレコードの数

Rnl　：　要素 A,B ともに空データのレコードの数
W　：　評価の重み (0 ¡ W ¡ 1)

```
  if ( average(Ra, Rb) > 2/3 * R & W * Rnl > Ra + Rb)
relate( A, B);
 else if ( average(Ra, Rb) < 1/3 * R & W * Rab > Ra + Rb)
relate( A, B);
 else if( W * (Rab + Rnl) > 2 * (Ra + Rb))
relate( A, B);
```

この評価関数を使用し、属性間の所在に強い関連性を見出せたものについてのカテゴライズ機能を TransX に追加した。実際のデータセットにこの手法を適用した結果を次に示す。

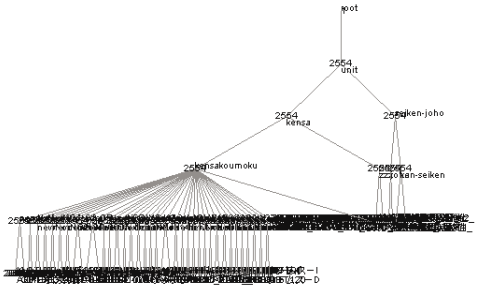

図 8: クラスタリング結果

5.2.1　結果

図 7 は、データを TransX にそのまま適用したグラフである[1]。これに、本手法を適用したのが次の図 8 である。木が自動的に再構成され、各要素がユーザに認識しやすい形で階層化されているのが分かる。
図 8 を詳しく分析してみると、フィルタの自動構成の結果、検査項目について約 30 のカテゴライズが与えられた。上記図 6 の ALB[2] と ALP[3]、GOT[4] と GOP[5] など、互いに関係性が深いと考えられていた属性同士が正しくカテゴライズされていることが分かる。これらは、いわゆるルーチン検査で肝臓の状態を見るためにしばしば一緒に検査されている検査項目である。

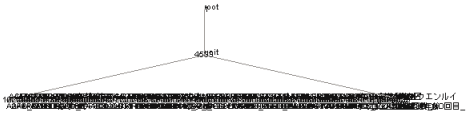

図 7: オリジナルグラフ

[1]実際のシステムでは、それぞれの属性を拡大表示することができる
[2]重症肝障害で低値
[3]肝硬変・肝細胞癌・胆道系疾患・慢性肝不全で高値
[4]さまざまな肝炎・肝障害、肝癌・肝硬変・胆汁うっ滞・閉塞性黄疸で高値
[5]さまざまな肝炎・肝障害、肝癌・肝硬変・胆汁うっ滞・閉塞性黄疸で高値

6　今後の課題

XML を使用する欠点として、データの増幅が考えられる。CSV 形式のファイルを XML で記述することによりデータ量が約１０倍に膨れ上がるし、その処理系も十分ではない。
また、これまでの手法の確認と、学習等を用いたフィルタパス生成の自動化が必要であると考えられる。

参考文献

[1] 医療データ提供: 千葉大医学部付属病院医療情報部, 千葉大医学部付属病院第一内科

[2] Peter Cabena, PabloHadjinian, “*Discovering Data Mining*” Prentice Hall PTR, 1998.

[3] Xindong Wu,“Induction as Pre-processing” *PAKDD*, 114-122, 1999.

[4] Ragel A, Cremilleux B,“Treatment of Missing Values for Association Rules” *PAKDD*, 258-270, 1998.

[5] 五十嵐 建平,“XML 変形を用いたデータマイニングにおける前処理の自動化” 東京工業大学 修士論文, 2001.

[6] World Wide Web Consotium,“External Makeup Language(XML)” http://www.w3.org/XML/

[7] 矢野 幸司, “XML 変形を用いたデータマイニングにおける前処理の自動化” 東京工業大学 学士論文 2002.

[8] 木村 亮, “データマイニングの前処理におけるデータベースの構造変換に関する研究” 東京工業大学 学士論文 2002.