第1章　PMMLの概要

1.1　PMMLの基礎

米国を中心とする世界的なIT企業のデータマイニングツール開発部門を母体として，Data Mining Groupというコンソーシアムが構成され，その中でデファクトスタンダードとしてデータマイニングに必要なデータのXMLベース表現規格がPMMLとして提案さてている．PMMLは完成された規格ではなく，データマイニング技術やXML技術の発展に沿って，Verion 1.0から始まり現在Version 2.1がリリースされ，更にVersion 3.0の作成が進められている．なお，これらの規格の策定過程はSOURCEFORGE.NETというサイトのPublic Forumでオープンにされており，第三者が問い合わせや意見を述べることができる．

1.2　マイニング結果モデル表現スキーマ

上記PMML記述内の各マイニング結果モデルを記述するXMLSchemaは以下の通りである．

```
<xs:element name="ExampleModel">
 <xs:complexType>
  <xs:sequence>
   <xs:element ref="Extension" minOccurs="0" maxOccurs="unbounded"/>
   <xs:element ref="MiningSchema"/>
   <xs:element ref="ModelStats" minOccurs="0" maxOccurs="1"/>
   ...
   <xs:element ref="Extension" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="modelName" type="xs:string" use="optional"/>
  <xs:attribute name="functionName" type="MINING-FUNCTION" use="required"/>
  <xs:attribute name="algorithmName" type="xs:string" use="optional"/>
 </xs:complexType>
```

```
</xs:element>
```

$ExampleModel$ 内では，どのような方法でマイニングを行ったかを $MiningSchema$ に記述される．具体的には元データのどの属性を用い，どの属性を用いなかったか，クラスを必要とするモデルの場合にはそのクラス属性名，外れ値や欠値の取り扱い方法などが記述される．$ModelStats$ には，そのモデルを構築するのに使用したデータ数やモデルの平均予測誤差などの統計的情報が記述される．$ExampleModel$ タグでは, モデル名 $modelName$，そのマイニング機能名称 $functionName$，アルゴリズム名称 $algorithmName$ といったものを属性として表記できることが書かれている．属性 $functionName$ には $associationRules$, $sequences$, $classification$, $regression$, $clustering$ の５種類の機能を記述でき，このモデルが如何なる目的に使用可能な機能を有するかを示す．

1.3 データタイプスキーマ

上記ドキュメント記述内に現れるデータには，幾つかの基本タイプが存在する．それらは一般数値タイプ，整数値タイプ，高精度実数値タイプ，確率値タイプ，百分率タイプ，具体的な属性名を指定して値を定義するタイプなどである．例えば高精度実数値タイプのスキーマならば以下のように定義される．

```
<xs:simpleType name="REAL-NUMBER">
 <xs:restriction base="xs:double"/>
</xs:simpleType>
```

また以下のように配列を定義することも可能である．

```
<Array n="3" type="int"> 1 22 3
</Array>
<Array n="3" type="string">
ab  "a b"   "with \"quotes\" "
</Array>
```

実データでは，n で配列の長さを指定し，type でそれが整数値配列か，実数値配列か，文字列の配列かを指定し，このタグの中に実データを記述する．

1.4 PMML 記述例

以下に PMML によるアソシエーションルール記述の例を示す．$< DataDictionary >$ 内でデータ記述が 2 つあることを示し，このタグに挟まれた 2 行で transaction と item というデータが記号データであることを記述している．更にその次の $< AssociationModel >$ タグ内で機能がアソシエーションルールであり，4 つのトランザクションデータから作られたモデルであり，アイテムは全部で 3 種類あり，最小支持度，最小確信度がそれぞれ 0.6，0.5，得られた多頻度集合が 3 つあり，得られたアソシエーションルールが 2 つあることを記述している．次の $< MiningSchema >$ タグに挟まれた部分では，$transaction$ と $item$ という 2 つの属性を持つデータからマイニングされたものであることが示されている.$< Itemset >$ タグ部分では $id =$ ”1” のアイテムセットが支持度が 1 で，含まれるアイテム数が 1 であることなどが記述されている．更に $< AssociationRule >$ タグでは $id =$ ”1” のアイテムセットがボディー $(antecedent)$，$id =$ ”2” のアイテムセットがヘッド $(consequent)$ であるアソシエーションルールの支持度，確信度共に 1.0 であることなどが記されている．

```
<?xml version="1.0" ?>
<PMML version="2.1" >
<Header copyright="www.dmg.org" description=
 "example model for association rules"/>
<DataDictionary numberOfFields="2" >
<DataField name="transaction" optype="categorical" />
<DataField name="item" optype="categorical" />
</DataDictionary>
<AssociationModel functionName="associationRules"
  numberOfTransactions="4" numberOfItems="3" minimumSupport="0.6"
  minimumConfidence="0.5" numberOfItemsets="3" numberOfRules="2">
  <MiningSchema>
    <MiningField name="transaction"/>
    <MiningField name="item"/>
  </MiningSchema>
<!-- We have three items in our input data -->
<Item id="1" value="Cracker" />
<Item id="2" value="Coke" />
<Item id="3" value="Water" />
```

```
<!-- and two frequent itemsets with a single item -->
<Itemset id="1" support="1.0" numberOfItems="1">
  <ItemRef itemRef="1" />
</Itemset>
<Itemset id="2" support="1.0" numberOfItems="1">
  <ItemRef itemRef="3" />
</Itemset>
<!-- and one frequent itemset with two items. -->
<Itemset id="3" support="1.0" numberOfItems="2">
  <ItemRef itemRef="1" />
  <ItemRef itemRef="3" />
</Itemset>
<!-- Two rules satisfy the requirements -->
<AssociationRule support="1.0" confidence="1.0" antecedent="1" consequent="2" />
<AssociationRule support="1.0" confidence="1.0" antecedent="2" consequent="1" />
</AssociationModel>
</PMML>
```

第2章　グラフ構造のPMML表現の提案

2.1　スキーマ

スキーマを各タグについて個別に説明する．以下は，グラフモデルの XMLSchema である．グラフモデルは複数の内部タグから成り立ち, その内部は $MiningSchema$，$ModelStats$, $Graph$ タグを順番にかくことで表される.$GraphModel$ タグの属性ではモデル名 ($modelName$), どういった意図で用いられるのか $functionName$, モデルを生み出す特定のアルゴリズム ($algorithmName$), グラフの数 ($recordCount$) を記述できるようになっている．また, 要素 $ModelStats$ が最小無くてもよく最大で１つしか許されないことが記述されている．シーケンスの最初と最後には，任意の個数の拡張データをを使用者が必要に応じて記述できるように工夫されている．

```
<?xml version="1.0" encoding="EUC-JP" ?>
<xs:schema
xmlns:xs="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.dmg.org/PMML-2_1"
xmlns="http://www.dmg.org/PMML-2_1"
elementFormDefault="unqualified">
<xs:element name="GraphModel">
  <xs:complexType>
  <xs:sequence>
  <xs:element ref="Extension" minOccurs="0" maxOccurs="unbounded"/>
  <xs:element ref="MiningSchema"/>
  <xs:element ref="ModelStats" minOccurs="0" maxOccurs="1"/>
  <xs:element ref="Graph"/>
  <xs:element ref="Extension" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="modelName" type="xs:string" use="optional"/>
  <xs:attribute name="functionName" type="MINING-FUNCTION" use="required"/>
```

```
   <xs:attribute name="algorithmName" type="xs:string" use="optional"/>
   <xs:attribute name="recordCount" type="NUMBER" use="optional"/>
   </xs:complexType>
 </xs:element>
```

Graph スキーマの定義は以下の通りである．グラフは $Vertex$ タグと $Edge$ タグを用いて表される．グラフ内には任意個の $Vertex$ や $Edge$ が含まれるので $maxOccurs$ は上限なし ($unbounded$) となっている．Graph タグの属性では各グラフデータを識別するための $graphId$ 名，グラフの種類を記述する $miningStatus$ とグラフマイニング結果として得られるグラフの種類を示すための $graphType$，グラフが部分グラフであった場合の支持度を示すための $support$ が $Graph$ タグの属性として定義されている．いずれも対象がグラフデータなのか，グラフマイニングによって得られた結果としてのモデルなのかによって必要性が異なるので，オプションとなっている．

```
<xs:element name="Graph">
 <xs:complexType>
  <xs:sequence>
   <xs:element ref="Extension" minOccurs="0" maxOccurs="unbounded"/>
   <xs:element ref="Vertex" minOccurs="0" maxOccurs="unbounded"/>
   <xs:element ref="Edge" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="graphId" type="xs:string" use="optional"/>
  <xs:attribute name="miningStatus" type="MINING-STATUS" use="optional"/>
  <xs:attribute name="graphType" type="GRAPH-TYPE" use="optional"/>
  <xs:attribute name="support" type="PROB-NUMBER" use="optional"/>
 </xs:complexType>
</xs:element>
```

以下は上記の $miningStatus$ で指定するグラフのタイプ $MINING-STATUS$ を定義するスキーマである．グラフが元データを表す $original$ か，マイニング結果として導かれる誘導部分グラフを表す $induced$ であるか一般部分グラフを表す $general$ であるかのいずれかでなければならないことが記述されている．

```
<xs:simpleType name="MINING-STATUS">
 <xs:restriction base="string">
```

```
  <xs:enumeration value="original"/>
  <xs:enumeration value="induced"/>
  <xs:enumeration value="general"/>
 </xs:restriction>
</xs:simpleType>
```

以下は同様にグラフマイニング結果として得られる構造の種類を$GRAPH-TYPE$として定義している．ルート無しツリー，ルート付きツリー，順序木，パス，閉路を含むグラフのいずれかであること示している．

```
<xs:simpleType name="GRAPH-TYPE">
  <xs:restriction base="string">
   <xs:enumeration value="unRootedTree"/>
   <xs:enumeration value="rootedTree"/>
   <xs:enumeration value="orderedTree"/>
   <xs:enumeration value="path"/>
   <xs:enumeration value="graph"/>
 </xs:restriction>
</xs:simpleType>
```

更に以下２つは，グラフの構成基本要素である$Vertex$と$Edge$を定義するスキーマである．$Vertex$は内部に$VertexLabel$タグをもつ．また属性として$Vertex$の識別名である$VertexId$，次数$dimension$を持つ．次数によって$VertexLabel$が何次元のラベルであるのかを明記できる．更に下部には$VertexLabel$の定義も与えられている．これはラベル名称$field$とその値$value$からなる．$Edge$の定義はより複雑である．同様に$EdgeLabel$タグを内部にもち，$edgeId$や$EdgeLabel$の$dimension$を有するが，それ以外にも有向エッジであるか無向エッジであるかを示す$edgeType$，いずれの$Vertex$間を結ぶかを示す$bgnVertexId$，$endVertexId$を属性で指定する．

```
<xs:element name="Vertex">
 <xs:complexType>
 <xs:sequence>
  <xs:element ref="VertexLabel" minOccurs="0" maxOccurs="unbounded"/>
 </xs:sequence>
  <xs:attribute name="VertexId" type="xs:string" use="required"/>
```

```
  <xs:attribute name="dimension" type="xs:int"/>
 </xs:complexType>
</xs:element>
<xs:element name="VertexLabel">
 <xs:attribute name="field" type="xs:string"/>
 <xs:attribute name="value" type="xs:string"/>
</xs:element>

<xs:element name="Edge">
 <xs:complexType>
  <xs:sequence>
   <xs:element ref="EdgeLabel" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="edgeId" type="xs:string" use="required"/>
  <xs:attribute name="edgeType" type="EDGE-TYPE" default="undirected"/>
  <xs:attribute name="dimension" type="xs:int"/>
  <xs:attribute name="bgnVertexId" type="xs:string"/>
  <xs:attribute name="endVertexId" type="xs:string"/>
 </xs:complexType>
</xs:element>

<xs:simpleType name="EDGE-TYPE">
 <xs:restriction base="string">
  <xs:enumeration value="directed"/>
  <xs:enumeration value="undirected"/>
 </xs:restriction>
</xs:simpleType>

<xs:element name="EdgeLabel">
 <xs:attribute name="field" type="xs:string"/>
 <xs:attribute name="value" type="xs:string"/>
</xs:element>
</xs:schema>
```

2.2 グラフ表現の実例

以上のスキーマに基づき，図 2.1 に示すエチレンを PMML で表現した例を以下に示す．$< DataDictionary >$ タグに挟まれた部分では，用いられるデータに *atomy* と *bondtype* が

図 2.1: エチレンの平面構造

あり，両者は記号属性であって，$atomy$ の値は H と C のいずれかであり $bondtype$ の値は単結合，芳香族結合，二重結合，三重結合のいずれかであることを記述している．その次の $< GraphModel >$ タグ部分によって実際のグラフが記述されている．モデル名は仮に $sample$ としてある．$< MiningSchema >$ タグではこのモデルが $atomy$ と $bondtype$ という属性を扱うことを示しである．$< Graph >$ タグ内ではエチレンの構造が記述されている．1 つの $< Graph >$ タグで 1 つのグラフが表現される．同様に, 各 $< Vertex >$ タグは 1 つの $Vertex$ を表し，それぞれに識別番号が振られ，かつ $atomy$ というスカラーのラベルが付与されていること ($dimension$ が 1)，またそれぞれの元素名 (H または C) が記述されている．各 $< Edge >$ タグは 1 つの $Edge$ を表し，それぞれに識別番号が振られ，結合には方向性がないので $edgeType$ として $undirected$，$bondtype$ として単結合か二重結合，始点 $bgnVertexId$ 及び終点 $endVertexId$ の $Vertex$ の識別番号が記述されている．エチレンでは二重結合は 2 つの炭素 C 間にしか存在しないので，対応する 3 番と 4 番の $Vertex$ 間の結合だけが二重結合となっている．これは 1 番目のグラフの記述であるが，以下続けて他の識別番号で複数のグラフを記述することが可能である．

```
<?xml version="1.0" encoding="EUC-JP"?>
<PMML version="2.1" >
 <Header copyright="www.dmg.org"
  description="Sample Graph Model"/>
 <DataDictionary numberOfFields="2">
  <DataField name="atomy" optype="categorical">
   <Value value="H"/>
   <Value value="C"/>
  </DataField>
  <DataField name="bondtype"
   optype="categorical">
   <Value value="singlebond"/>
```

```
  <Value value="aromaticbond"/>
  <Value value="doublebond"/>
  <Value value="triplebond"/>
 </DataField>
</DataDictionary>
<GraphModel modelName="sample">
 <MiningSchema>
  <MiningField name="atomy"/>
  <MiningField name="bondtype"/>
 </MiningSchema>
 <Graph graphId="1">
  <Vertex VertexId="1" dimension="1">
   <VertexLabel field="atomy" value="H"/>
  </Vertex>
   <Vertex VertexId="2" dimension="1">
   <VertexLabel field="atomy" value="H"/>
  </Vertex>
  <Vertex VertexId="3" dimension="1">
   <VertexLabel field="atomy" value="C"/>
  </Vertex>
  <Vertex VertexId="4" dimension="1">
   <VertexLabel field="atomy" value="C"/>
  </Vertex>
  <Vertex VertexId="5" dimension="1">
   <VertexLabel field="atomy" value="H"/>
  </Vertex>
  <Vertex VertexId="6" dimension="1">
   <VertexLabel field="atomy" value="H"/>
  </Vertex>
  <Edge edgeId="1" edgeType="undirected"
   dimension="1" bgnVertexId="1" endVertexId="3">
    <EdgeLabel field="bondtype" value="singlebond"/>
   </Edge>
  <Edge edgeId="2" edgeType="undirected"
   dimension="1" bgnVertexId="2" endVertexId="3">
    <EdgeLabel field="bondtype" value="singlebond"/>
```

```
    </Edge>
   <Edge edgeId="3" edgeType="undirected"
    dimension="1"  bgnVertexId="3" endVertexId="4">
     <EdgeLabel field="bondtype" value="doublebond"/>
    </Edge>
   <Edge edgeId="4" edgeType="undirected"
    dimension="1"  bgnVertexId="4" endVertexId="5">
     <EdgeLabel field="bondtype" value="singlebond"/>
    </Edge>
   <Edge edgeId="5" edgeType="undirected"
    dimension="1"  bgnVertexId="4" endVertexId="6">
     <EdgeLabel field="bondtype" value="singlebond"/>
    </Edge>
  </Graph>
    . . .
 </GraphModel>
</PMML>
```