The Management and Mining of Multiple Predictive Models Using the Predictive Modeling Markup Language (PMML)

Robert Grossman
National Center for Data Mining, University of Illinois at Chicago
& Magnify, Inc.

Stuart Bailey, Ashok Ramu and Balinder Malhi
National Center for Data Mining
University of Illinois at Chicago

Philip Hallstrom, Ivan Pulleyn and Xiao Qin
Magnify, Inc.

Abstract

Keywords: data mining, predictive modeling, data interchange formats, XML, SGML, ensemble learning, partitioned learning, distributed learning

We introduce a markup language based upon XML for working with the predictive models produced by data mining systems. The language is called the Predictive Model Markup Language (PMML) and can be used to define predictive models and ensembles of predictive models. It provides a flexible mechanism for defining schema for predictive models and supports model selection and model averaging involving multiple predictive models. It has proved useful for applications requiring ensemble learning, partitioned learning, and distributed learning. In addition, it facilitates moving predictive models across applications and systems.

1 Introduction

Data mining can be defined as the automatic discovery of patterns, associations, changes and anomalies in large data sets. Broadly speaking there are two cultures emerging in the field: 1) The predictive mining (PM) culture is concerned with exploiting data mining algorithms to produce predictive models automatically [Grossman 1998a]. 2) The knowledge discovery (KD) culture is concerned with exploiting data mining algorithms to extract valid, novel and useful knowledge [Fayyad 1996].

Both approaches have their merits: if one's goal is to improve the real-time detection of fraud, then a tree with 10,000 nodes or an ensemble of 100 trees may yield a very accurate predictive model. On the other hand, if one's goal is to gain knowledge about historical fraud in order to change business processes with the goal of reducing future fraud, then extracting useful knowledge from a simple Bayes net or a single shallow tree may provide more useful knowledge, even it is does not provide as powerful a predictive model.

Our point of view in this paper is to consider data mining from the predictive modeling point of view. Roughly speaking, by a predictive model, we mean a classifier or predictor, such as a tree or neural net.

By an ensemble of models, we mean a collection of predictive models. By selection and averaging rules we mean rules for selecting and combining predictive models or ensemble of predictive models to produce a single result.

As a simple example, a collection of models predicting yes/no may be combined by a simple vote. More sophisticated methods are also useful, such as Bayesian model averaging [Raftery 1996]. See [Dietterich 1997] for a good review of ensemble of models and learning.

Ensembles of models are important in data mining for several reasons. 1) Ensembles of models often yield more accurate predictors [Brieman 1996 and Wolpert 1992]. 2) Large data sets may be partitioned, the different partitions may be mined in parallel to produce an ensemble of models, and then the results combined using model averaging [Grossman 1996]. 3) Distributed data may be mined initially separately to produce an ensemble of models and then a single predictive model produced from this ensemble. Sometimes this is called meta-learning [Chan 1995].

In this paper, we introduce a markup language based upon XML [W3 XML Spec 1997] called the Predictive Model Markup Language (PMML). PMML is designed to provide a convenient mechanism for working with the types of predictive models and ensembles of predictive models which arise in data mining. In particular, we feel that PMML is well suited for partition learning [Bodek 1997], meta-learning [Chan 1995], distributed learning, and related areas.

Specialized formats and languages for the interchange of data are quite common in scientific and engineering computing [Buneman 1995]. Sometimes these are called data exchange formats (DX formats). From this perspective, PMML may be thought of as an interchange format for predictive models, that is a model interchange format or mif.

From another perspective, XML is rapidly emerging as a useful language for working with data and meta-data on the web. The XML-data specification [Layman 1998] is a good example of recent work in this area. For related work, see [W3 XML 1998].

Models described using PMML consist of several parts: 1) a header, 2) a data schema, 3) a data mining schema, 4) a predictive model schema, 5) definitions for predictive models, 6) definitions for ensembles of models, 7) rules for selecting and combining models and ensembles of models, 8) rules for exception handling. Component 5) is required. In addition a schema for the predictive model must be defined. This can be done using one or more of the schemas - components 3, 4, and 5. The other components are optional.

Ever since there has been statistical software, there has been interchange formats for predictive models. We feel though that this paper makes several contributions: First, the role within the data mining process of a well designed interchange format for predictive models has not been emphasized before, as far as we are aware. Second, our experience with a variety of data mining applications has shown the usefulness of providing a flexible mechanism for dealing with the different types of attributes which arise within the data mining process and for supporting not only single models but also ensembles of models. Third, interchange formats for predictive models have tended to be closed and proprietary; our goal here is to encourage the development of an open and flexible interchange format, based upon XML, and specifically designed to support the needs of data mining applications.

Section 1 is an introduction. Section 2 introduces PMML through a simple example. Section 3 provides some background information on the role of predictive modeling in data mining. Section 4 provides additional background about ensemble learning. Section 5 introduces PMML for ensembles of models. Section 6 contains some supplementary information about PMML. Section 7 discusses the implementation status of PMML. Section 8 contains some concluding remarks.

Version. The examples in this paper are illustrated using Version 0.9 of the Predictive Model Markup Language (PMML 0.9) [PMML 1998]. This is currently being revised and Version 1.0 is expected to contain some significant differences.

2 A Simple Example

In this section, we introduce the Predictive Model Markup Language with a simple example. This example contains three components: a data schema, a model schema and the definition of a model, in this case a CART tree. See Figure 1.

This example requires only the most basic XML. In XML, there are two types of tags, a *start tag* (such as `<CART-tree>`) and an *end tag* (such as `</cart-tree>`). The information between these tags is called the *contents*. An XML *element* includes the start tag, the contents, and the end tag. Start tags and end tags are required, although the content is not. The start tag may also contain optional attributes, as in the following example:

```
<data-attribute attribute-number='1' attribute-name='velocity'
value='3' > </data-attribute>
```

A start tag and an end tag with no content may be abbreviated:

```
    <data-attribute attribute-number='1' attribute-name='velocity'
                          value='3' />
```

Recall that a CART tree is built by splitting the learning set of objects into two by asking a simple question, such as:

Is attribute 3 < 4.1?

If so, an object is sent to the left child; otherwise, to the right child. To code this the XML describing a node in a CART tree need only specify the children of the node, the attribute defining the split, and the split value, as in the following fragment:

```
<cart-model model-id='1'   type='binary-classification'
     attribute-predicted='Fraud Indicator' number-nodes='13' depth='3'>
<cart-node node-number='0' model-attribute-number='3'
     cut-value='4.1' left-child='1' right-sibling='6'>

     etc.
```

```
    </cart-model>
```

Notice that the split is defined using what are called *model-attributes*. The current version of PMML supports several types of attributes, including:

> *Data attributes*. These are the most basic types of attributes. Often these are imported from a database system.
>
> *Mining attributes*. A data mining system requires additional information, beyond that required for a database. For example a data attribute which is a string may be treated by a data mining system as a nominal attribute to be excluded, such as a name, or as a categorical attribute, such as M or F.
>
> *Model attributes*. Supporting separate model attributes makes it easier to support multiple models and more convenient when only some of the data attributes are used for a particular model.

Data attributes are defined using the tag <data-schema> and model attributes are defined using the tag <model-schema>. The option *corresponds* is used to indicate the correspondence between model attributes and data attributes, as in the following fragment:

```
    <model-attribute-descriptor attribute-number='1' attribute-name='number
    of transactions past hour' corresponds-to='data-attribute 12' data-
    type='integer'>
```

To summarize, a single model is easily described by specifying 1) a data schema and 2) the parameters of the model. If convenient, a data schema and a model schema may both be used to provide greater flexibility.

```
<pmml>
<data-schema>

<attribute-descriptor attribute-number='1' attribute-name='card number'
use-as='exclude' data-type='string'>
<attribute-descriptor attribute-number='2' attribute-name='timestamp'
use-as='exclude' data-type='string'>
<attribute-descriptor attribute-number='3' attribute-name='dollar amount'
use-as='continuous' data-type='real'>
<attribute-descriptor attribute-number='4' attribute-name='issuer' use-
as='category' data-type='integer'>

     etc.

</data-schema>

<model-schema>

<attribute-descriptor attribute-number='1' attribute-name='number of
transactions past hour' corresponds-to='data-attribute 12' data-
type='integer'>
<attribute-descriptor attribute-number='2' attribute-name='number of
transactions past two hours' corresponds-to='data-attribute 15' data-
type='integer'>

     etc.

</model-schema>

<cart-model type='binary-classification' attribute-predicted='Fraud
     Indicator' number-nodes='13' depth='3'>
<cart-node node-number='0' attribute-number='model-attribute 1'
     cut-value='3' left-child='1' right-child='6'>
<cart-node node-number='1' attribute-number='model-attribute 3'
     cut-value='5' left-child='2' right-child='3'>

     etc.

</cart-model>

</pmml>
```

Figure 1. A PMML fragment defining a CART model. Note that the attributes for the predictive model are defined in terms of the data attributes.

3 Predictive Models and the Data Mining Process

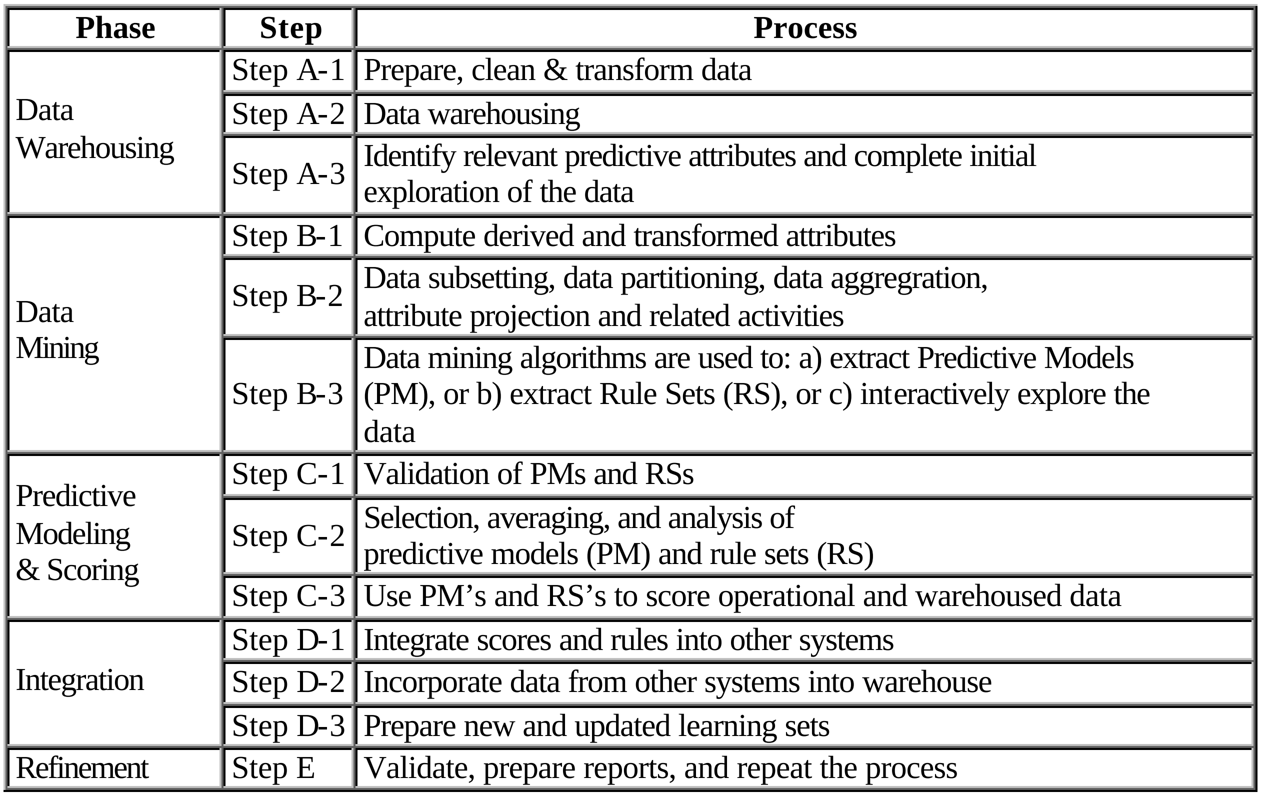
Table 1 and Figure 1 are adapted from [Grossman 1998b] and summarizes the major steps in what is usually called the data mining process. The following observations are relevant for the purposes here:

- For many problems, mining may be viewed as the extraction of a learning set from a data warehouse and the production of a predictive model by the data mining system.
- More than one predictive model may be produced. Selecting and combining predictive models is an important activity. This is the role of the data modeling system.
- For data mining to be useful in decision support the predictive models produced must be incorporated into operational systems.
- There are several different types of attributes that are part of the data mining process. Keeping track of them is an important.

Referring to Figure 2, the output of the data mining system, the input and the output of the data modeling system, and the input to the scoring system are all predictive models. PMML is a convenient language for importing and exporting predictive models between these different systems. By using this type of common infrastructure, the complexity of the total system may be dramatically reduced. In other words, a key role of PMML is to facilitate the importing and exporting of predictive models between the various subsystems that comprise a typical decision support environment, including the data warehouse, the data mining system, the data modeling system, and the operational systems.

A major contributing factor towards the complexity is that each subsystem typically requires a different set of attributes. For example, a data warehouse may aggregate 1000 attributes about each customer. A specific data mining algorithm may produce 5 models, each using only about 100 attributes, but each model may use a slightly different set of attributes. The PMML must therefore keep track of:

- *The Data Schema*. This is the list of attributes used in the data warehouse. PMML refers to these attributes as *data* attributes.
- *The Data Mining Schema*. This is the list of attributes used by a specific data mining algorithm. PMML refers to these attributes as *mining* attributes.
- *The Predictive Modeling Schema*. This is the list of attributes used by a specific predictive model. PMML refers to these attributes as *model* attributes.



Phase	Step	Process
Data Warehousing	Step A-1	Prepare, clean & transform data
	Step A-2	Data warehousing
	Step A-3	Identify relevant predictive attributes and complete initial exploration of the data
Data Mining	Step B-1	Compute derived and transformed attributes
	Step B-2	Data subsetting, data partitioning, data aggregration, attribute projection and related activities
	Step B-3	Data mining algorithms are used to: a) extract Predictive Models (PM), or b) extract Rule Sets (RS), or c) interactively explore the data
Predictive Modeling & Scoring	Step C-1	Validation of PMs and RSs
	Step C-2	Selection, averaging, and analysis of predictive models (PM) and rule sets (RS)
	Step C-3	Use PM's and RS's to score operational and warehoused data
Integration	Step D-1	Integrate scores and rules into other systems
	Step D-2	Incorporate data from other systems into warehouse
	Step D-3	Prepare new and updated learning sets
Refinement	Step E	Validate, prepare reports, and repeat the process

Table 1. The data mining process, adapted from [Grossman 1998b].

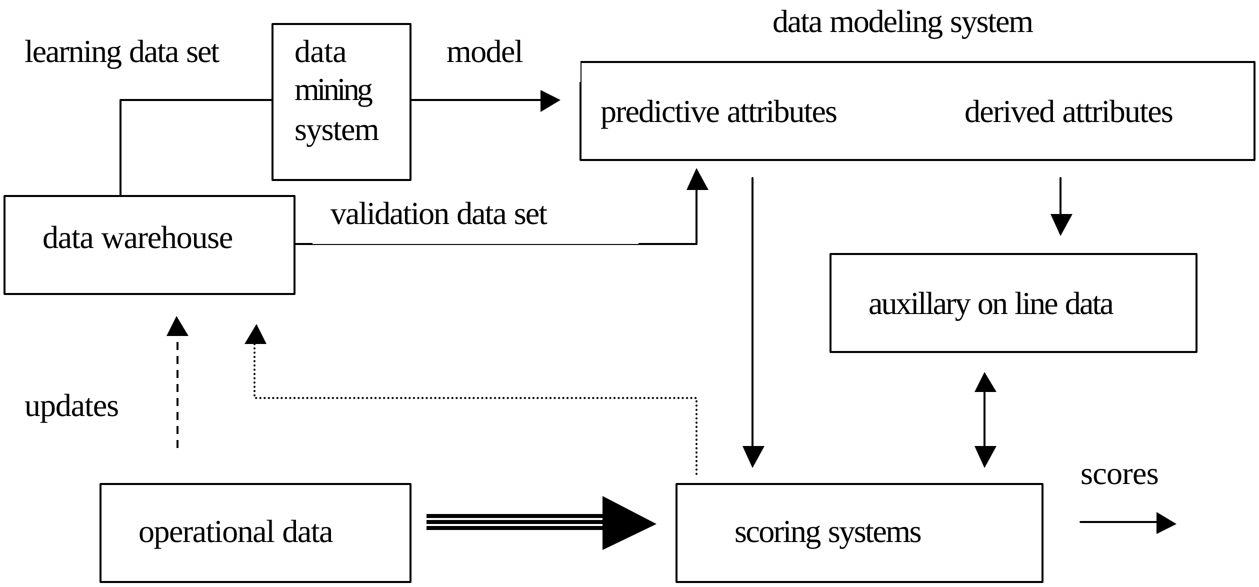


Figure 2. A typical decision support environment contains a data warehouse, a data mining system, a data modeling system, and a scoring system. Predictive models are the outputs of the data mining system and the data modeling system and the inputs of the data modeling system and the scoring system. Using PMML as a common language can dramatically reduce the complexity of decision support environments. This Figure is from [Grossman 1998].

4 Ensembles of Models

For large data sets, or for data sets which are very heterogeneous, it is generally more effective to use a collection of predictive models rather than a single predictive model. Suppose the goal is to attach a score y to each customer or transaction x. With a single predictive model f, the score y is a simple function

$$y = f(x).$$

With *model selection*, one constructs an ensemble of models

$$\{f[1], ..., f[k]\}$$

and selects the appropriate model f[j] based upon the characteristics of the customer x:

$$j = j(x), \qquad y = f[j](x).$$

In other words, the predictive model f[j] selected from the ensemble depends upon x. The score y then depends upon the model as usual. Here are some examples:

Example 1. Suppose the goal of the predictive model is to predict bankruptcy and three models have been developed:

Model	**Rule**
f[1]	x is a home owner
f[2]	x does not own a home and x has held current job for at least three years
f[3]	x does not own a home and x has held current job for less than three years

Note that the rules assign each x to precisely one model and can easily be interpreted as a tree, with three leaves, corresponding to the three models.

With *model averaging*, one assigns weights w[1], ..., w[k] to an ensemble of models {f[1], ..., f[k]} based upon the attributes of x and uses the score:

$$w[1] = w_1(x), \ldots, w[k]=w_k(x), \qquad y = f[0](w[1]\, f[1](x), \ldots, w[k]\, f[k](x)),$$

where f[0] is a rule for combining the various models.

Example 2. The simplest means of combining classifiers is with majority vote. Given a ensemble of classifiers {f[1], …, f[k]}, simply compute the score f[j](x) for each classifier individually and score the ensemble using a majority vote:

$$y = \begin{cases} 1 & \text{at least k/2 classifiers f[j](x) vote 1} \\ 0 & \text{otherwise} \end{cases}$$

Example 2a. *Partitioned Learning*. Suppose that 100 Gigabytes of data has been partitioned into 20 partitions, each containing 5 Gigabytes and a separate classifier f[j] is built for each partition. With *partition learning*, in the simplest case, equal weights w[j] = 1 are used, and a majority vote is used to score the ensemble.

Partitioned learning is important since it is very easily parallelized using a master-slave paradigm [Gropp 1994] and requires no communication between the different slave processes and very little communication between the master and the slave processes. PMML is a convenient way for the slave processes to return the predictive models built. The master process then simply creates an ensemble model by concatenating the individual models built by the slave processes. For further details, see [Bodek 1997 and Grossman 1998c].

Example 2b. *Distributed Learning*. Suppose that a data set is distributed over a wide area network. With distributed learning a separate classifier is built using the data in each location and then the classifiers are sent back to a central location.

Again PMML is a convenient way to return the predictive models. Often distributed learning is done over wide area networks using agents. A query agent dispatches agents to each of the data sites. The dispatched agents simply return PMML models which are assembled into an ensemble by the query agent. See [Chan 1995] and [Grossman 1998c] for further details.

5 An Example Involving Multiple Models

In this section, we illustrate using two simple examples how PMML works with multiple predictive models. An ensemble is a collection of predictive models delimited by an `ensemble` tag. Each model is delimited by a `model` tag. The default is for each model in an ensemble to be used to score a given object. The score is simply the sum of the scores of each of the models. If weights are provided as optional arguments to the model tag, then a weighted sum is used. See Figure 3 for an example.

Alternatively, the ensemble may provide selection rules for using one or more of the models. Currently, the selection rules must be in the form of a tree, called a selection tree. Each leaf of the selection tree has a value associated with it, corresponding to the model-id of the corresponding model. Note that for this to work model-ids must be defined in the model tags of the appropriate models. See Figure 4.

```
<ensemble ensemble-number="0">

     <data-schema>
          etc.
     </data-schema>

     <model model-number="0" type="cart" weight="0.5">
          <model-schema>
               etc.
          </model-schema>
          <cart-model>
               etc.
          </cart-model>
     </model>


     <model model-number="1" type="cart" weight="0.5">
          <model-schema>
               etc.
          </model-schema>
          <cart-model>
               etc.
          </cart-model>
     </model>

</ensemble>
```

Figure 3. A code fragment in PMML illustrating an ensemble containing two CART models. Note that in this example a data schema is defined for the ensemble and each model defines its own model-schema by selecting the appropriate attributes from the data schema. Each model is assigned a weight of 0.5.

```
<ensemble>

<selection-tree number-nodes='4'>
<!-- attribute 1 is home owner (1 = homeowner; 0 otherwise)
     attribute 2 number of years current job -->
     <selection-tree-node node-id='0' attribute-number='1'
     attribute-value='1' split-rule='Equal' left-child='1'
     right-sibling='2'>
     <selection-tree-leaf node-id='1' leaf-value='1'>
     <selection-tree-node node-id='2' attribute-number='2'
     attribute-value='3' split-rule='LessThanEqual' left-
     child='3' right-sibling='4'>
     <selection-tree-leaf node-id='3' leaf-value='2'>
     <selection-tree-leaf node-id='4' leaf-value='3'>
</selection tree>

<model model-id='1'>
     etc.
</model>

     etc.

<model model-id='3'>
     etc.
</model>

</ensemble>
```

Figure 4. An PMML fragment showing the use of model selection. A tree is used to describe the rules for selecting the bankruptcy model describe in Example 1 of Section 4.

6 PMML - Additional Concepts

Our goal in this paper has been to illustrate the usefulness of PMML and to introduce it with several examples. We have not described several features of the language, including:

- In PMML, a header can be used to describe the learning set, the algorithm used, the data mining application, and related information.
- Although we have mentioned several types of attributes including data attributes, mining attributes and model attributes, it is also useful to extend the language to support additional types of attributes, including predictive attributes and control attributes [Grossman 1996].
- Not only are there a variety of different types of attributes, but it is also useful to extend the language to support the transformation of attributes. For example, rules for cleaning data attributes can be conveniently expressed as transformation rules.
- Rules for exception handling are currently under discussion and will probably be included in some form in the next draft of PMML.

7 Implementation Status

To date, there have been two implementations of PMML: one by the National Center for Data Mining (NCDM) at the University of Illinois at Chicago and one by Magnify, Inc.

Version 0.8 of PMML was defined using SGML. Both NCDM and Magnify developed tree-based classifiers and related tools incorporating PMML. This technology was demonstrated at the Internet 2/Highway 1 Workshop on October 7 and 8, 1997 in Washington, D.C. and at the Supercomputing 97 Conference on November 17-20, 1997 in San Jose, California.

The National Center for Data Mining developed tools using PMML for mining data distributed over the internet and over specialized high performance networks, such as the vBNS.

Magnify uses PMML in its product PATTERN™ as a portable mechanism for moving predictive models between its data mining system and operational systems which require the predictive models for scoring data.

Version 0.9 of PMML is the current version and is defined using XML. Both the NCDM and Magnify have partial implementations of this version.

8 Summary and Conclusion

PMML is a markup language based on XML which is designed to support the types of predictive models which arise in data mining. PMML supports not only single predictive models, but also ensembles of predictive models, as well as several mechanisms for selecting and combining predictive models. This makes PMML particularly suited for ensemble learning, partition learning, and meta-learning. PMML also supports a variety of different types of attributes and provides several mechanisms for working with different subsets of attributes. This makes PMML particularly well suited for applications involving large data sets and large numbers of attributes.

As Figure 2 above illustrates, the data mining process typically involves several different systems, including a data warehouse, a data mining system, a predictive modeling system, and a variety of operational systems. The output of the data mining system, the inputs and outputs of the predictive modeling system, and one of the inputs to the operational systems all involve predictive models. Using PMML can substantially simply the design of a complete data mining system and at the same time increase its flexibility.

Finally, PMML is a good data exchange format for predictive models and as such can provide a portable means for moving predictive models between heterogeneous systems and for archiving them.

Version 0.9 is the current version of PMML. There are two partial implementations of this version and the preliminary results are quite promising.

References

[Bodek 1997] H. Bodek, R. L. Grossman, and I. Pulleyn, Detecting Network Intrusions through the Data Mining of Network Packet Data Using the ACT Algorithm, Mathematical Modeling and Scientific Computing, Volume 8, 1997, to appear.

[Brieman 1996] Bagging Predictors, Machine Learning, Volume 24, Number 2, pages 123-140.

[Buneman 1995] P. Buneman, S. Davidson, R. Grossman and D. Maier, "Interoperating with Non-Database Data," University of Pensylvania Computer Science Department Technical Report, 1995.

[Chan 1995] P. Chan and S. Stolfo, A Comparative Evaluation of Voting and Meta-Learning on Partitioned Data, Proceedings of the Twelfth International Conference on Machine Learning, pages 90-98.

[Dietterich 1997] Machine Learning Research: Four Current Directions, to appear.

[Fayyad 1996] U. M. Fayyad, G. Piatetsky-Shapiro, and P. Smyth, "From Data Mining to Knowledge Discovery: An Overview," in Advances in Knowledge Discovery and Data Mining, edited U. M Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, AAAI Press/MIT Press, pp. 1-34, 1996.

[Gropp 1994] W. Gropp, W. Lusk, and A. Skjellum, Using MPI: Portable Parallel Programming with the Message-Passing Interface, MIT Press, Cambridge, Massachusetts, 1994.

[Grossman 1996] R. L. Grossman, H. Bodek, D. Northcutt, and H. V. Poor, Data Mining and Tree-based Optimization, Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, E. Simoudis, J. Han and U. Fayyad, editors, AAAI Press, Menlo Park, California, 1996, pp 323-326.

[Grossman 1998a] R. Grossman, Data Mining: The Two Cultures, Magnify Technical Report 98-1, submitted for publication.

[Grossman 1998b] R. L. Grossman, M Cubed: A Framework for Integrating Data Management, Data Mining and Predictive Modeling within the Knowledge Discovery Process, Magnify Technical Report 98-2, submitted for publication.

[Grossman 1998c] R.L. Grossman, S. Bailey, A. Ramu and B. Malhi, Experimental Results of a System for High Performance, Distributed Data Mining, submitted for publication.

[Layman 1998] A. Layman, J. Paoli, S. De Rose, H. S. Thompson, Specification for XML-Data, http://www.microsoft.com/xml.

[Microsoft XML FAQ 98] Frequently Asked Questions About Extensible Markup Language (XML), http://www.microsoft.com/xml.

[PMML 1998] The Predictive Model Mark Up Language Version 0.9, The Data Mining Group.

[Raftery 1996] A. E. Raftery, D. Madigan, and Jennifer A. Hoeting, Bayesian Model Averaging for Linear Regression Models, submitted for publication.

[W3 XML 1998] For information about XML, see http://www.w3.org/XML.

[W3 XML Spec 1997] Working Draft Specification for XML, http://www.w3.org/pub/WWW/TR/WD-xml-lang-970331.html

[Wolpert 1992] Stacked Generalization, Neural Networks, Volume 5, pages 241-259.