Standard per Data Mining

Sistemi informativi per le Decisioni

Slide a cura di prof. Claudio Sartori

Iniziative di Standard per Data Mining

- I modelli di Data Mining e statistici generati dagli strumenti commerciali di Data Mining devono spesso essere utilizzati in altri sistemi
  - customer relationship management (CRM)
  - enterprise resource planning (ERP)
  - risk management
  - intrusion detection
  - dati scientifici
  - dati ingegneristici
- Gli standard semplificano integrazione, aggiornamento e mantenimento dei sistemi che utilizzano i modelli

Cosa Coprono gli Standard

- Modelli
  - Per rappresentare dati di Data Mining e statistici
- Attributi
  - Per rappresentare la pulizia, la trasformazione, l'aggregazione di attributi usati come input nei modelli
- Interfacce ed API
  - Per collegarsi ad altri linguaggi e sistemi
- Configurazioni
  - Per rappresentare i parametri interni richiesti per costruire ed utilizzare i modelli
- Processi
  - Per produrre, installare e usare i modelli
- Dati remoti e distribuiti

Approcci

- PMML
  - Predictive Model Markup Language
  - Derivato da XML
- Microsoft OLE DB for Data Mining
  - È uno standard interno all’ambiente Microsoft

XML

- I parametri di un modello di Data Mining, come una rete neurale, possono essere rappresentati in XML
- Esempio: il nodo 10 di una rete neurale ha connessione in input dal nodo 0 con peso -2.08148

```
<Neuron id="10">
    <Con from="0"
    weight="-2.08148"/>
```

- Gli standard per la definizione di modelli parametrici con XML sono sufficientemente maturi
  - Assumono che gli input al modello siano dati esplicitamente
- Purtroppo però in generale gli input non sono così espliciti
  - I dati devono prima essere puliti e trasformati
  - Non ci sono ancora standard convincenti per la pulizia e la trasformazione
- Gli standard relativi al processo operazionale del Data Mining stanno emergendo

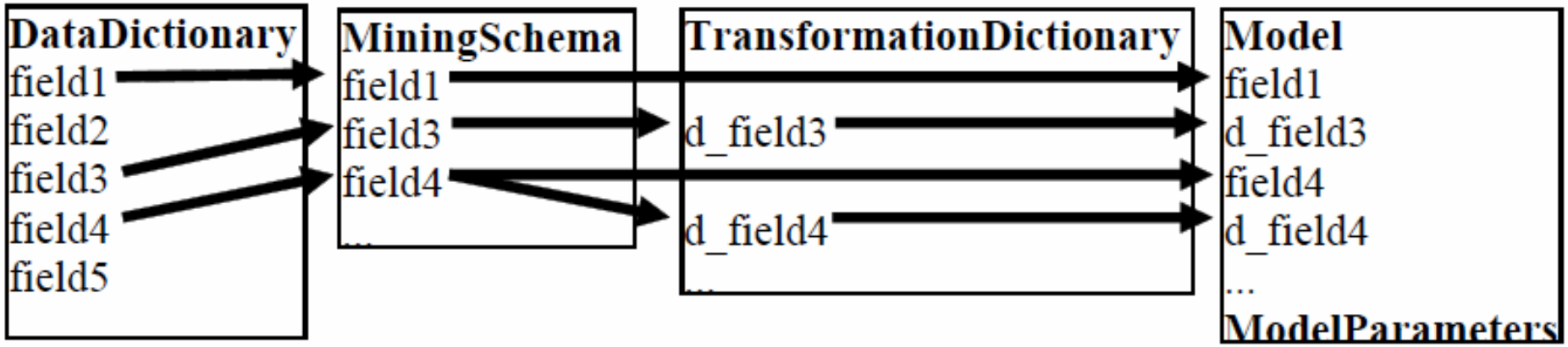
Data Mining Group (`http://www.dmg.org`)

- Consorzio di produttori nato nel 1998 con lo scopo di stabilire standard per il Data Mining
- PMML
  - rappresentare e descrivere modelli
  - rappresentare e descrivere operazioni preliminari di pulizia e trasformazione
  - fornire infrastrutture tali da permettere a un'applicazione di produrre modelli
    - PMML Producer
  - usare modelli
    - PMML Consumer
  - memorizzare
    - PMML XML data file

PMML

- Data dictionary
  - Definisce gli attributi di input
    - Anche attributi derivati
    - Supporto di matrici sparse e dati temporali
- Mining schema
  - Attributi dello schema e loro ruolo del modello
  - Sottoinsieme del data dictionary
  - Informazioni specifiche del modello
  - Tipo di uso degli attributi (input, output, supplementare, ignorato)
    - Indicatore di importanza, possibile utilizzo diverso in diversi modelli
- Transformation dictionary
  - Normalizzazione (mapping da continui o discreti verso numeri)
  - Discretizzazione (mapping da continui verso discreti)
  - Traduzione (mapping da discreti a discreti)
  - Aggregazione

Relazioni tra i componenti



PMML (ii)

- Statistiche
  - statistiche univariate riguardo agli attributi del modello
- Modelli
  - Parametri specificati da etichette (versione 2.0)
    - Regressione
    - Cluster
    - Alberi
    - Reti neurali
    - Modelli Bayesiani
    - Regole associative
    - Sequenze

PMML (iii)

- Rule set
  - Modelli di alberi di decisione appiattiti
  - Più potenti degli alberi di decisione
  - Generano predizioni con confidenze associate (scoring)
  - Non sono pensati per sostituire gli alberi di decisione, ma per integrarne l'utilizzo
- Support Vector Machines
  - Iperpiani di separazione tra i valori di un determinato campo target
  - Definiti con l'utilizzo di kernel function
    - Lineari, polinomiali, radiali, sigmoidi
  - Utilizzate per classificazione e regressione

PMML (iv)

- Mining di testi
  - Dizionario di termini
  - Corpus di testi (riferimenti)
  - Matrice documenti-termini
  - Normalizzazione del modello di testo
  - Similarità del modello di testo
    - Come confrontare due vettori che rappresentano documenti?

Infrastruttura – Composizione di modelli

- Un modello può essere visto come una trasformazione
- Più modelli possono essere combinati in sequenza per formarne uno più complesso
  - Adatto anche alla preparazione dei dati
- Il risultato di un modello può essere utilizzato per selezionare quale modello deve essere applicato al passo successivo
  - Ad esempio, un albero di decisione può avere un modello di regressione incluso in un nodo

Infrastruttura – Funzioni Utente

- Il pre-trattamento può essere eseguito con molteplici funzioni *built-in*
  - Somma, differenza, prodotto, divisione, logaritmo, …
  - Funzioni di stringa
- Si possono definire funzioni *custom*
  - Ad esempio, estrarre il numero di giorni dall'inizio dell'anno, a partire dalla data

API standard

- SQL
  - Per permettere ad applicazioni di DM di accedere ai dati su DBMS relazionale
- Java
  - Costruzione di modelli di DM e scoring di dati
  - Creazione, mantenimento e accesso a dati e metadati
- Microsoft
  - OLE DB for DM
    - Supporta interazione con DM nelle applicazioni Microsoft
  - Tassonomie di dati e meccanismi di trasformazione
  - Microsoft Analysis Services
    - Nuovo concetto in SQL Server 2000

Scoring

- Processo di utilizzo di un modello analitico basato su dati storici per effettuare predizioni riguardo al comportamento futuro
- Generalmente produce numeri
- Utilizzato in
  - Market Basket Analysis
  - Churn analysis (analisi delle “perdite”)
  - Sicurezza
  - Frodi
  - Ammissioni
  - Valutazione rischio credito
  - Valutazione rischio medico
  - Diagnosi medica
- Scoring engine

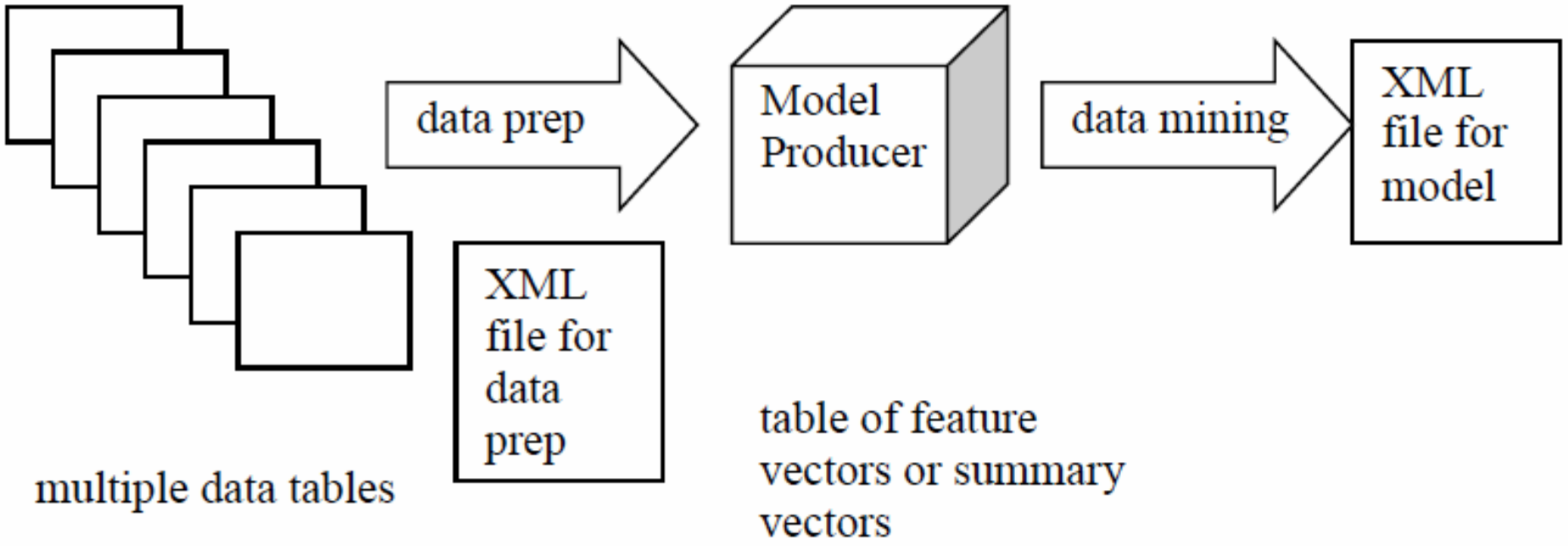
Data Mining Metadata

- Common Warehouse Model for Data Mining (CWM DM)
- Configurazioni per la costruzione di modelli
- Rappresentazione di modelli
- Rappresentazione di risultati
- Generazione di XML DTD

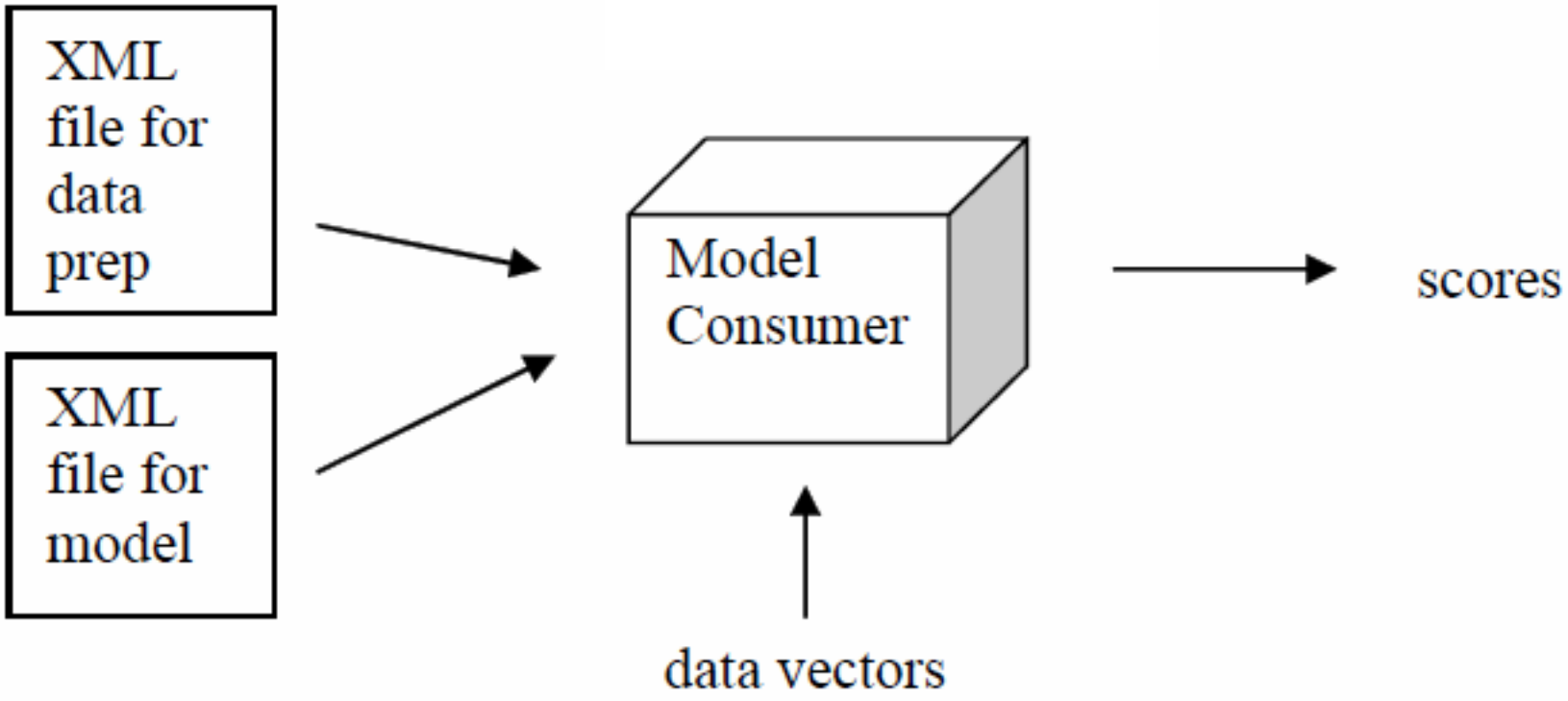
PMML Producers and Consumers

- Hanno diverse necessità
- Un analista dei dati (producer) può utilizzare un sistema di Data Mining per mesi, fino a ricavare un modello PMML soddisfacente
- Uno scoring engine (consumer) a basso impatto di calcolo potrebbe essere inserito in un sistema operazionale per assegnare valori in tempo reale
- Una chiara interfaccia tra produttori e consumatori fornisce un meccanismo semplice di separazione fra la fase di modellazione e quella di deployment
- Per lo stesso motivo è opportuno avere una chiara separazione nella fase di preparazione

Produttore di Modello



Consumatore di Modello



Composizione di Modelli – Approfondimento

- Quale è l'architettura più appropriata e lo standard più opportuno per permettere la composizione di:
  - Azioni di preparazione dati?
  - Applicazione di modelli di Mining?
- Esempio 1
  - Un albero di classificazione può essere utilizzato per scegliere tra due o più modelli di regressione
- Esempio 2
  - Un modello di regressione logistica può essere utilizzato come input di un albero di classificazione

Problemi da Considerare

- La composizione di operazioni di Data Mining è definita soltanto parzialmente, in generale non è ben definito come connettere output e input
- L’idea di composizione si esplicita in diversi casi di utilizzo pratico
- Come la chiara separazione tra produttori e consumatori di modelli ha permesso lo sviluppo di un'ampia varietà di scoring engine e applicazioni di Data Mining embedded, si intende giungere a una chiara architettura per la preparazione dei dati
  - Bilanciare generalità, complessità e potenza

Idee Base

- I modelli possono essere inclusi in altri modelli
  - In questo caso molte delle infrastrutture possono essere semplificate, poiché si mettono in comunicazione oggetti che parlano la stessa lingua
  - PMML 3.0 embedded model
- La sequenza di modelli può essere supportata con la tecnica delle trasformazioni usata per gli attributi derivati
  - Un modello embedded può generare un ResultField, che può essere input di un altro modello
- Fornire un contenitore per una collezione di modelli: MiningModel
  - La selezione può essere supportata con un decision tree all'interno di un MiningModel
  - Il voting può essere supportato come modello di regressione all'interno di un MiningModel

Preparazione dati

```
<Discretize field="Profit">
  <DiscretizeBin binValue="negative">
    <Interval closure="openOpen" rightMargin="0" />
    <!-- left margin is -infinity by default -->
  </DiscretizeBin>
  <DiscretizeBin binValue="positive">
    <Interval closure="closedOpen" leftMargin="0" />
    <!-- right margin is +infinity by default -->
  </DiscretizeBin>
</Discretize>
```

Composizione di Modelli

- Usare modelli in forma embedded e usare un albero di decisione per selezionare il modello appropriato
  - Il container include i diversi modelli di regressione
  - Il DT fornisce la logica di selezione
- Sequenza: usare un attributo derivante da un modello di regressione come attributo di scelta in un albero di decisione
  - L'elemento PMML "Regression" ha le informazioni necessarie alla definizione di un nuovo attributo "ResultField" come modello di regressione
  - ResultField può essere input al modello che lo contiene

```
<PMML>
...
  <MiningModel function="regression">
  <MiningSchema>
     as usual
   </MiningSchema>

   ... derived fields as usual ...

  <DecisionTree>  <!-- wrapper for content as in TreeModel -->

    <Node><True/>  <!-- root node in a DecisionTree is always True -->

    <Node>  <!-- 1st sub-tree, is a leaf node -->
      <Predicate age<=50 .../>

      <Regression> <!-- embedded regression equation -->
        <RegressionTable intercept="2.34">
          ... predictors:  0.03*income + 1.23*age ...
        <RegressionTable>
      </Regression>
    </Node>

    <Node> <!-- 2nd sub-tree, is a leaf node -->
      <Predicate age>50 .../>

       <Regression> <!-- embedded regression equation -->
         <RegressionTable intercept="2.22">
          ... predictors:  0.01*income -0.11*age*mc ...
        <RegressionTable>
       </Regression>
     </Node>

     </Node> <!-- end of root node -->

   </DecisionTree>

  </MiningModel>
</PMML>
```

DecisionTree contains nodes that can contain embedded models

regression model

regression model

```
<PMML>
...
<TreeModel function='regression'>
   <MiningSchema>
   <!-- declare fields "age", "income", "married", ...  as usual -->
   <MiningSchema>

  <!-- encode a categorical input field using a PMML 2.1 transformation -->
  <DerivedField name="mc" optype="continuous">
     <MapValues ... >

     <!-- map "yes" to 1.0
     map "no" to -1.0 -->

     </MapValues>
  </DerivedField>

  <!-- derive a new input term -->
  <!-- use an embedded regression model as a transformation -->
  <Regression>
    <ResultField name="term" feature="predicted">
     <!-- The ResultField selects the predicted value from
       this regression element  and binds
       it to the name "term" -->

     <!-- RegressionTable as defined in RegressionModel -->
     <RegressionTable>
       <!—with  intercept="2.34"
       and predictors:  0.03*income + 1.23*age*mc  -->
      <RegressionTable>
  </Regression>

    <Node> <!-- this is the root node of the TreeModel  -->
    <!-- A decision tree as usual,
     it can refer to fields "age", "income", "married"
     as well as to the derived attribute "mc"
     and regression attribute "term"  -->
     ...

    </Node>
  </TreeModel>
</PMML>
```

This code defines a derived field called “mc”.

This code defines a result field called “term” using a regression model.

This code defines a node that can use the derived field “mc” and the result field “term”.