Enhancing A Production Rule Engine With Predictive Models Using PMML

Davide Sottara [dsotty@gmail.com]
Paola Mello, Claudio Sartori and Emory Fry

PMML Workshop, Aug 21st 2011
Outline

1. Introduction

2. PMML-driven Hybrid Systems
   - Integration Patterns
   - Implementation

3. Conclusions
Outline

1 Introduction

2 PMML-driven Hybrid Systems
   • Integration Patterns
   • Implementation

3 Conclusions
Background: (Business) Rules

- (Semi-)declarative knowledge
- Premise-Conclusion
Background: (Business) Rules

- (Semi-)declarative knowledge
- Premise-Conclusion
- Many applications: medicine, finance, networking, security, ...
Background: (Business) Rules

- (Semi-)declarative knowledge
- Premise-Conclusion
- Many applications: medicine, finance, networking, security, ...

What kind of relationship is there with predictive models?
Background: (Business) Rules

- (Semi-)declarative knowledge
- Premise-Conclusion
- Many applications: medicine, finance, networking, security, ...

What kind of relationship is there with predictive models?

- Online model evaluation
- Model chaining

Rule engine: Drools
“When you have a young, overweight diabetic patient then prescribe a treatment...”
business rules

"When you have a young, overweight diabetic patient then prescribe a treatment..."

rule "Basic"
when
    Patient( problem == "diabetes",
        age < 30, weight > 100 )
then
    insert( new Prescription("insuline") );
end
**Goal**

**rule** "As Patient"
**when**
    $p : \text{Person}(/* \text{conditions apply} */)
**then**
    $p \text{ isA Diabetic}
    \text{update(} $p \text{)};
**end**

**rule** "Treat Diabetic"
**when**
    $p : \text{Diabetic}\text{(age ... , weight ...)}
**then**
    /* start process */
**end**
Classification

- Provided as a fact
- Rule-based inference
- **Predictive Models**
- Semantic reasoning
A vast family

- Decision Trees (DT)
- Neural Networks (NN)
- Bayesian Networks (BA)
- Clustering Algorithms (CL)
- Support Vector Machines (SVM)
- ...

Implicit knowledge learned from data
(Predictive) Models

A vast family

- Decision Trees (DT)
- Neural Networks (NN)
- Bayesian Networks (BA)
- Clustering Algorithms (CL)
- Support Vector Machines (SVM)
- ...

Implicit knowledge learned from data

Classifiers vs Predictors

Inference by model evaluation
# PM: Benefits and Drawbacks

<table>
<thead>
<tr>
<th></th>
<th>Learn</th>
<th>Discover</th>
<th>Adapt</th>
<th>Flexible</th>
<th>Explain</th>
<th>Graceful</th>
</tr>
</thead>
<tbody>
<tr>
<td>CBR</td>
<td>★★</td>
<td>★</td>
<td>★★★</td>
<td>★</td>
<td>★★★</td>
<td>★</td>
</tr>
<tr>
<td>Rules</td>
<td>★★</td>
<td>★★</td>
<td>★</td>
<td>★★★</td>
<td>★★★★</td>
<td>★</td>
</tr>
<tr>
<td>NN</td>
<td>★★★</td>
<td>★</td>
<td>★★★</td>
<td>★★★</td>
<td>★</td>
<td>★★★</td>
</tr>
<tr>
<td>RBF</td>
<td>★★★</td>
<td>★★★</td>
<td>★★★</td>
<td>★★★</td>
<td>★★★</td>
<td>★★★</td>
</tr>
<tr>
<td>FS</td>
<td>★</td>
<td>★</td>
<td>★★★</td>
<td>★★★</td>
<td>★★★★</td>
<td>★★★★</td>
</tr>
<tr>
<td>BN</td>
<td>★★★</td>
<td>★★★</td>
<td>★</td>
<td>★★★</td>
<td>★★★★</td>
<td>★★★★</td>
</tr>
<tr>
<td>CL</td>
<td>★★★</td>
<td>★★★★</td>
<td></td>
<td></td>
<td>★★★★</td>
<td>★</td>
</tr>
</tbody>
</table>

**Table:** Comparison of AI techniques
Case Study: Signal Interpretation

Changes in trend correlated to process state
Event Processing Flow

- Chart
- Trainer
- Track
- Statistics
- Predict
- Analysis
- Policy
- Control
- Actuators
- Probes
- Denoise
- Trace
- Storage
- Router

Detection  Diagnosis  Action
Event Processing Flow

- Chart
- Trainer
- Statistics
- Predict
- Analysis
- Policy
- Control
- Actuators
- Trace
- Storage
- Sampler
- Noise
- Router

Detection  Diagnosis  Action
Event Processing Flow

- **Probes** → **Denoise** → **Sample** → **Analysis** → **Policy** → **Control** → **Actuators**
- **Chart** ↔ **Trainer** ↔ **Statistics**
- **Storage** ↔ **Router**

**Detection**  **Diagnosis**  **Action**
Event Processing Flow

- Chart
- Trainer
- Statistics
- Predict
- Sample_{pred}
- Policy
- Control
- Actuators
- Probes
- Denoise
- Analysis
- Trace
- AckStage
- Storage
- Router

Detection  Diagnosis  Action
Event Processing Flow

- Probes
- Denoise
- Analysis
- Policy
- Control
- Actuators
- Chart
- Trainer
- Statistics
- Predict
- Trace
- Storage
- Router

Detection  Diagnosis  Action
Event Processing Flow

- Probes
- Denoise
- Chart
- Trainer
- Statistics
- Predict
- Sample_{pred}
- Analysis
- TrendChange
- Policy
- Control
- Actuators
- Trace
- AckStage
- Storage
- Router

Detection   Diagnosis   Action
Event Processing Flow

1. Chart
2. Trainer
3. Statistics
4. Predict
5. Control
6. Actuators
7. Probes
8. Denoise
9. Analysis
10. Trace
11. Storage
12. Router

Detection, Diagnosis, Action
Event Processing Flow

- Probes → Denoise → Analysis → Trainer → Statistics
- Trace → Analysis → Policy → Switch → Control → Actuators
- Storage

Detection  Diagnosis  Action
Outline

1. Introduction

2. PMML-driven Hybrid Systems
   - Integration Patterns
   - Implementation

3. Conclusions
Outline

1. Introduction

2. PMML-driven Hybrid Systems
   • Integration Patterns
   • Implementation

3. Conclusions
Flame Wars

**PM**
- Compact
- Data

**Rules**
- Pragmatic
- Policies

**Semantics**
- Conceptual
- Relations
Flame Wars

- PM
  - Compact
  - Data

- Rules
  - Pragmatic
  - Policies

- Semantics
  - Conceptual
  - Relations

- Are we forced to choose?
Are we forced to choose?
No Integration - External Call

```plaintext
rule "No integration"
when
  $pm : PredictiveModel(...)
  $x : ... from accumulate( ... )
then
  insert( $pm.invoke($x) );
end
```

- Rules, at best, select the PM/Semantic module
- Module is invoked in RHS
No Integration - External Call

**Rule** "No integration"

**When**

$pm : \text{PredictiveModel}(\ldots)

$x : \ldots \text{from \ accumulate}(\ldots)$

**Then**

$\text{insert}(pm\text{.invoke}(x));$

**End**

- Rules, at best, select the PM/Semantic module
- Module is invoked in RHS
- Not declarative
- External lib dependency
- User-managed
- Mapping issues
Loose Integration - Wrapper

```plaintext
rule "Delegate"
when
  Person( this ~isA[ custom ] "Diabetic" )
  ...
then
  ...
end
```

- PM/Semantic module is embedded in a **custom evaluator**
- Only **classifiers** can be used in this way
Loose Integration - Wrapper

**rule** "Delegate"
**when**
  `Person( this ~isA[ custom ] "Diabetic")`...
**then**
  ...
**end**

- PM/Semantic module is embedded in a *custom evaluator*
- Only *classifiers* can be used in this way
- Data mapping
- External lib dependency
Strong Integration - Emulation

- PM/Semantic modules are implemented using **rules**
- Constraint evaluation triggers **rule chaining**
- Interface ← Trigger facts
Integration Patterns

Homogeneous Integration - Reactive

```
rule "Invoke"
when
  ...
then
  insert( new Input( ... ) );
end

rule "Model"
when
  Input( ... )
then
  insert( ... );
end
```
Homogeneous Integration - On Demand

```plaintext
query model(Input $in, Output $o)
    $o := Output($in);
end

rule "Main"
    when
        $in : Input(...)
        ?model($in, $out)
    then
        insert($out);
end

rule "On Demand"
    when
        DroolsQuery("model", _, $out)
        not Output(this == $out)
    then
        insert(new Trigger());
end
```
Outline

1. Introduction

2. PMML-driven Hybrid Systems
   - Integration Patterns
   - Implementation

3. Conclusions
Behind the scenes

- PMML Resources are parsed
- A (Drl) visitor creates a set of rules using templates
- The resulting rules emulate the Predictive model
  - Model evaluation (Dataflow)

PMML --> AST --> DRL Compiler --> DRL

% Templates
Data Types are mapped to input/output facts:

```xml
<DataField dataType="integer" name="fieldName">
    declares FieldName
    value : int
    valid : boolean = false
    missing : boolean = false
    ...
end
```
Restrictions become validation rules:

```xml
<DataField dataType="integer" name="fieldName">
  <Value>5</Value>
  <Value>6</Value>
</DataField>
```

```plaintext
rule "Valid_FieldName"
  when $f: FieldName( value == 5 || value == 6 )
  then
    modify ($f) {
      valid = true;
    }
end
```
Rules define transformations:

```xml
<DerivedField dataType="double" name="Field_norm">
  <NormContinuous field="FieldName"
    mapMissingTo="0" outliers="asExtremeValues">
    <LinearNorm norm="0" orig="0"/>
    <LinearNorm norm="1" orig="80"/>
    <LinearNorm norm="2" orig="100"/>
  </NormContinuous>
</DerivedField>
```

**rule** "Linear_field"

**when** FieldName( valid == true, missing == false, $x : value > 0 \&\& < 80 )

**then**

```
  double y = ... $x ...
  FieldNorm f = new FieldNorm( y );
  insert( f );
```

**end**
Mining Schema

- Rules bind fields to models
- Additional validation
- Outlier and missing values
Models

- Model structure and parameters → Facts
- Model evaluation → Rules
- Output is generated when all inputs are available

```plaintext
rule "Neuron Fire"
  when
    Number( $val: doubleValue ) from accumulate ( $c : Charge( context == "NN", index == 4, $in : value )
      sum ( $in )
      count( $c )
    )
  then
    insertLogical( new Stym( f($val) ) );
end
```

(Currently, only neural networks are fully supported)
Derived features

- Knowledge-based input/output validation
- Adaptive training
- Interaction (GUI)
Outline

1. Introduction

2. PMML-driven Hybrid Systems
   - Integration Patterns
   - Implementation

3. Conclusions
Conclusions

Drools-PMML

In development, available in Drools 6

- Homogeneous integration of rules and predictive models

Derived features can be provided:

- Support declarative integration patterns
- Knowledge-based input/output validation
- Adaptive training
- Interaction (GUI)
Questions?