Single function CRUG Meetup
Pmml() and onnx()

IBM Developer

Svetlana Levitan, PhD

Developer Advocate and PMML Release Manager
Center for Open Data and AI Technologies (CODAIT)
IBM Cognitive Applications

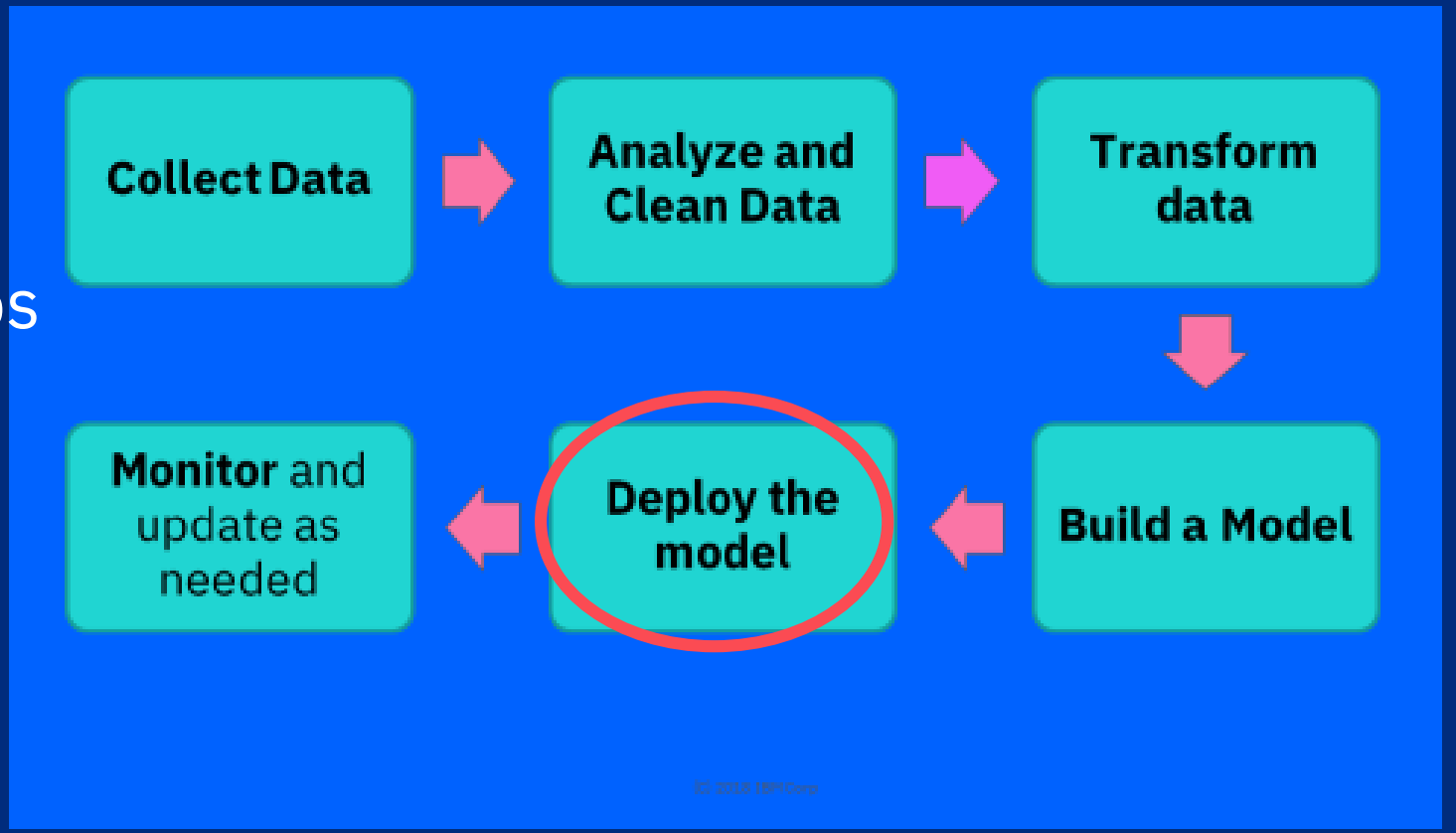
@SvetaLevitan

ML Model deployment

Challenges:

- Different Teams
- Different environments
- Need to keep data prep steps

A Solution: open standards



@SvetaLevitan

DMG and PMML

Data Mining Group (dmg.org) since 1990's

Predictive Model Markup Language

- An Open Standard for **XML** Representation
- Over 30 vendors and organizations
- 17 models + combinations + transforms + …
- dmg.org/pmml

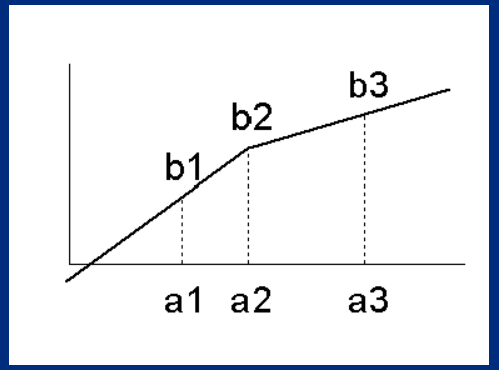
@SvetaLevitan

Main Components of PMML

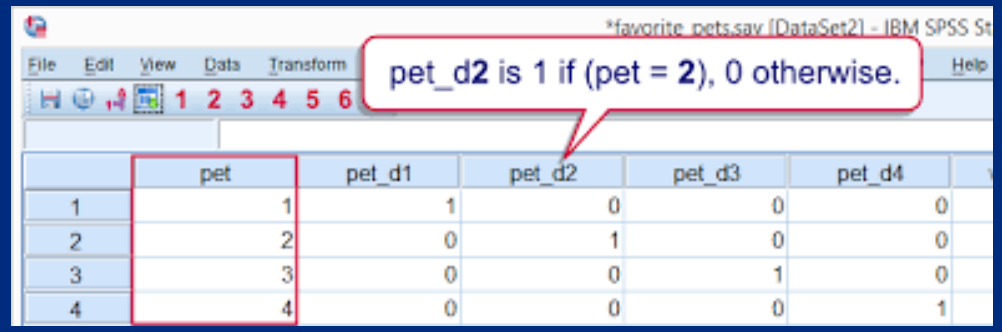
- Header
- Data Dictionary
- Transformation Dictionary
- Model(s)

Transformations

- **NormContinuous:** piece-wise linear transform



- **NormDiscrete:** map a categorical field to a set of dummy fields



	pet	pet_d1	pet_d2	pet_d3	pet_d4
1	1	1	0	0	0
2	2	0	1	0	0
3	3	0	0	1	0
4	4	0	0	0	1

- **Discretize:** binning
- **MapValues:** map one or more categorical fields into another categorical one
- **Functions:** built-in and user-defined
- Other transformations

PMML 4.4 Models

- Anomaly Detection (new)
- Association Rules Model
- Clustering Model
- General Regression
- Naïve Bayes
- Nearest Neighbor Model
- Neural Network
- Regression
- Tree Model

- **Mining Model:** composition or ensemble (or both) of models

- Baseline Model
- Bayesian Network
- Gaussian Process
- Ruleset
- Scorecard
- Sequence Model
- Support Vector Machine
- Time Series

R package pmml

There was also “**pmmlTransformations**”, now merged into pmml
https://cran.r-project.org/package=pmml

Recommends: **ada, amap, arules, caret, clue, data.table, gbm, glmnet, neighbr, nnet, rpart, randomForest, kernlab, e1071, testthat, survival, xgboost, knitr**

Maintained by Dmitriy Bolotov and others from Software AG

Build and save a decision tree (C&RT) model predicting Species class:

```
> irisTree <- rpart( Species~., iris )
> saveXML( pmml( irisTree ), "IrisTree.xml" )
```

More details for pmml()

Usage

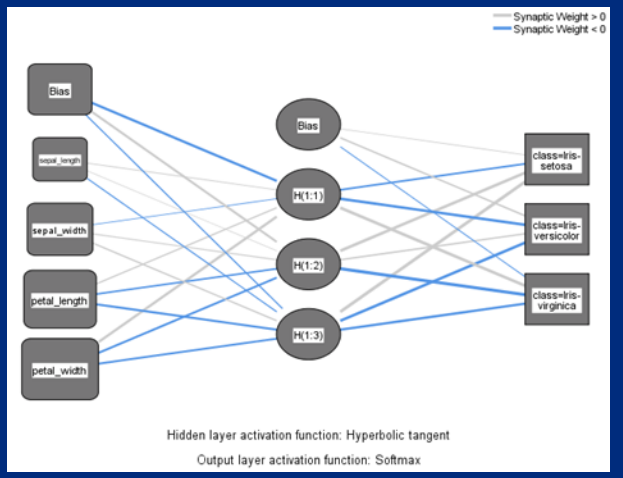
```
pmml(
  model = NULL,
  model_name = "R_Model",
  app_name = "SoftwareAG PMML Generator",
  description = NULL,
  copyright = NULL,
  transforms = NULL,
  ...
)
```

Method **pmml()** can be used to do any of the following:

- Export a model in PMML
- Export data transforms in PMML
- Merge data transforms into a PMML model

An example PMML – Data Dictionary, Transformations

```
▼<DataDictionary numberOfFields="5">
 ▼<DataField name="class" optype="categorical" dataType="string">
    <Value value="Iris-setosa"/>
    <Value value="Iris-versicolor"/>
    <Value value="Iris-virginica"/>
  </DataField>
  <DataField name="sepal_length" optype="continuous" dataType="double"/>
  <DataField name="sepal_width" optype="continuous" dataType="double"/>
  <DataField name="petal_length" optype="continuous" dataType="double"/>
  <DataField name="petal_width" optype="continuous" dataType="double"/>
</DataDictionary>
```

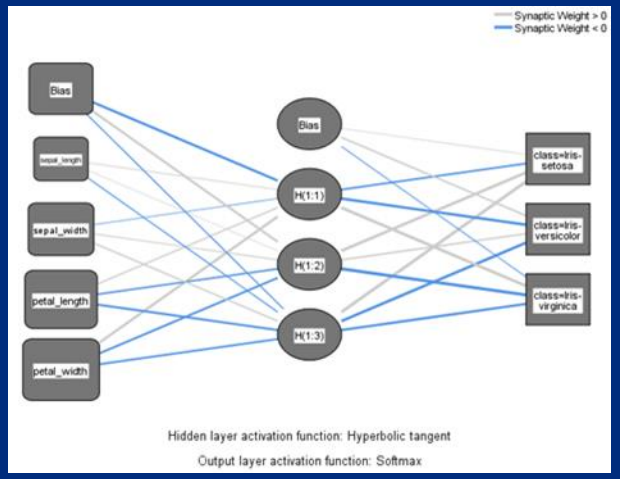



```
▼<DerivedField optype="categorical" dataType="double" name="classValue2">
    <NormDiscrete field="class" value="Iris-virginica"/>
  </DerivedField>
▼<DerivedField optype="continuous" dataType="double" name="sepal_lengthNorm">
 ▼<NormContinuous field="sepal_length">
    <LinearNorm orig="4.3" norm="-1.84285714285714"/>
    <LinearNorm orig="7.7" norm="2.3204081632653"/>
  </NormContinuous>
  </DerivedField>
▼<DerivedField optype="continuous" dataType="double" name="sepal_widthNorm">
 ▼<NormContinuous field="sepal_width">
    <LinearNorm orig="2" norm="-2.48539690378995"/>
    <LinearNorm orig="4.4" norm="3.13131926296699"/>
  </NormContinuous>
```

Example PMML – Neural Network MiningSchema and inputs

```
<NeuralNetwork functionName="classification" activationFunction="tanh">
  <MiningSchema>
    <MiningField name="sepal_length"/>
    <MiningField name="sepal_width"/>
    <MiningField name="petal_length"/>
    <MiningField name="petal_width"/>
    <MiningField name="class" usageType="predicted"/>
  </MiningSchema>
  <NeuralInputs>
    <NeuralInput id="0">
      <DerivedField optype="continuous" dataType="double">
        <FieldRef field="sepal_lengthNorm"/>
      </DerivedField>
    </NeuralInput>
    <NeuralInput id="1">
      <DerivedField optype="continuous" dataType="double">
        <FieldRef field="sepal_widthNorm"/>
      </DerivedField>
    </NeuralInput>
```

Predictors



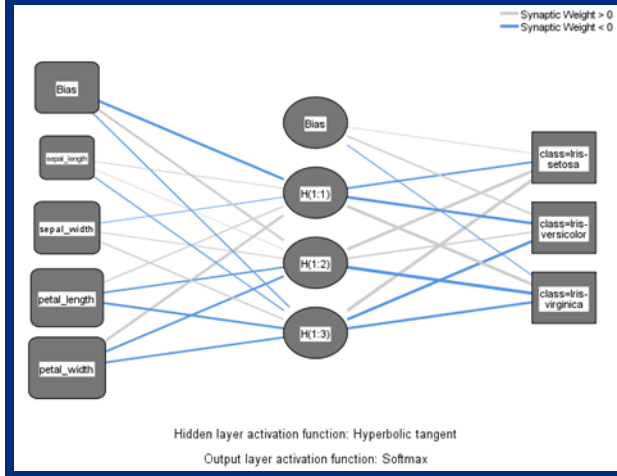
Example PMML - Neural Network hidden layer and outputs

```
      <Neuron id="6" bias="-0.69138649428932">
        <Con from="0" weight="-0.57324998362272"/>
        <Con from="1" weight="0.892806772564007"/>
        <Con from="2" weight="-1.23192787546061"/>
        <Con from="3" weight="-1.19705013526962"/>
      </Neuron>
    </NeuralLayer>
    <NeuralLayer numberOfNeurons="3" activationFunction="identity" normalizationMethod="softmax">
      <Neuron id="7" bias="0.101922887283541">
        <Con from="4" weight="-1.05690948855012"/>
        <Con from="5" weight="2.00228899161664"/>
        <Con from="6" weight="3.31278374396491"/>
      </Neuron>
      <Neuron id="8" bias="0.917636281284728">
        <Con from="4" weight="-1.47230776836775"/>
        <Con from="5" weight="0.905795272070893"/>
        <Con from="6" weight="-1.60793177845373"/>
      </Neuron>
      <Neuron id="9" bias="-0.2772471777484">
        <Con from="4" weight="2.22290439134024"/>
        <Con from="5" weight="-2.43960637239511"/>
        <Con from="6" weight="-1.32214182019044"/>
      </Neuron>
    </NeuralLayer>
    <NeuralOutputs>
      <NeuralOutput outputNeuron="7">
        <DerivedField optype="categorical" dataType="double">
          <FieldRef field="classValue0"/>
        </DerivedField>
      </NeuralOutput>
```

Hidden layer neuron

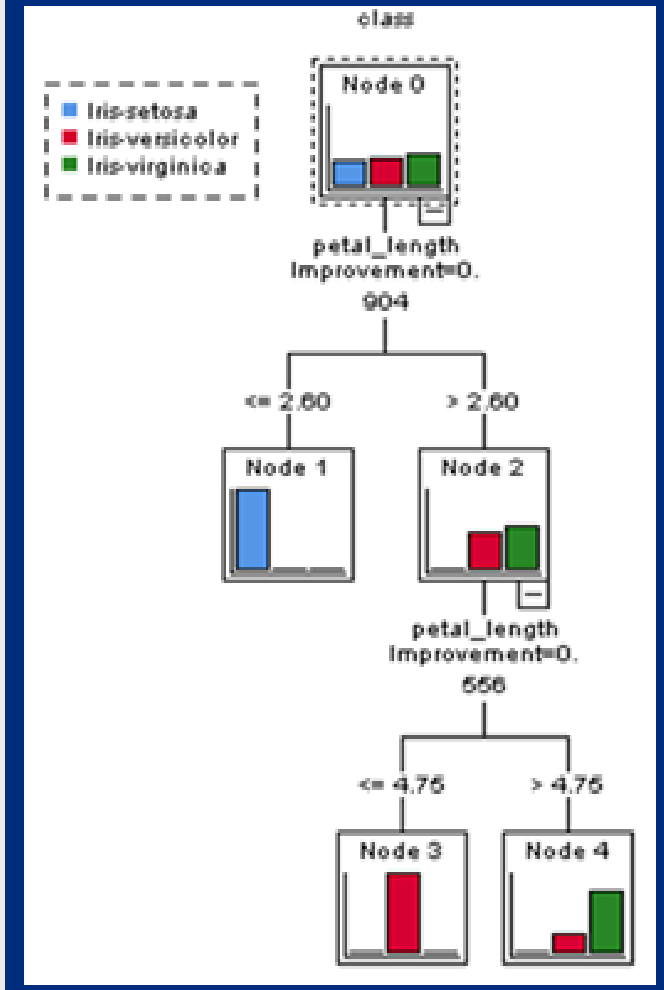
Output Layer Neurons

Connecting target to the neurons



Example PMML for a Tree Model

```
<Node id=“0"> <True/>
  <Node id=“1" score="Iris-setosa" recordCount="50.0">
     <SimplePredicate field="petal_length" operator="lessOrEqual“
                              value=“2.6"/>
     <ScoreDistribution value="Iris-setosa" recordCount="50.0"/>
     <ScoreDistribution value="Iris-versicolor" recordCount="0.0"/>
     <ScoreDistribution value="Iris-virginica" recordCount="0.0"/>
   </Node>
   <Node id=“2">
     <SimplePredicate field="petal_length" operator="greaterThan“
                              value=“2.6"/>
     <Node id=“3“score="Iris-versicolor" recordCount=“40.0">
        <SimplePredicate field="petal_length"
                                 operator="lessOrEqual" value=“4.75"/>
```



PMML Powered

From http://dmg.org/pmml/products.html:

Alpine Data

Angoss

BigML

Equifax

Experian

FICO

Fiserv

Frontline Solvers

GDS Link

IBM (Includes SPSS)

JPMML

KNIME

KXEN

Liga Data

Microsoft

MicroStrategy

NG Data

Open Data

Opera

Pega

Pervasive Data Rush

Predixion Software

Rapid I

R

Salford Systems (Minitab)

SAND

SAS

Software AG (incl. Zementis)

Spark

Sparkling Logic

Teradata

TIBCO

WEKA

ONNX: Open Neural Network eXchange – onnx.ai

Since 2017; Protobuf format

Covers DL and traditional ML

Active work by many companies

@SvetaLevitan

R package onnx

Github.com/onnx/onnx-r

```
library(onnx)

graph_def <- make_graph(
    nodes = list(
      make_node("FC", list("X", "W1", "B1"), list("H1")),
      make_node("Relu", list("H1"), list("R1")),
      make_node("FC", list("R1", "W2", "B2"), list("Y"))
    ),
    name = "MLP",
    inputs = list(
      make_tensor_value_info('X' , onnx$TensorProto$FLOAT, list(1L)),
      make_tensor_value_info('W1', onnx$TensorProto$FLOAT, list(1L)),
      make_tensor_value_info('B1', onnx$TensorProto$FLOAT, list(1L)),
      make_tensor_value_info('W2', onnx$TensorProto$FLOAT, list(1L)),
      make_tensor_value_info('B2', onnx$TensorProto$FLOAT, list(1L))
    ),
    outputs = list(
      make_tensor_value_info('Y', onnx$TensorProto$FLOAT, list(1L))
    )
)
print_readable(graph_def)
```

Conclusions

- Model deployment is an important part of ML lifecycle
- Data Mining Group works on open standards for model deployment
- PMML eases deployment for supported models and data prep
- ONNX is becoming a de-facto standard for Deep Learning

Links and contact

Dmg.org/pmml dmg.org/pfa onnx.ai codait.org

@SvetaLevitan slevitan@us.ibm.com