Deploying your machine learning model to unlock its potential

Svetlana Levitan, PhD
Developer Advocate and PMML Release Manager
Center for Open Data and AI Technologies (CODAIT)
IBM Digital Business Group
Who is Svetlana Levitan?

Originally from Moscow, Russia

PhD in Applied Mathematics and MS in Computer Science from University of Maryland, College Park

Software Engineer for SPSS Analytic components (2000-2018)

Developer Advocate with IBM Center for Open Data and AI Technologies (since June)

Married, two daughters who love programming
1500 Drones Are Ready To Fly

IBM Developer Drone Challenge

Challenge runs from November 12 through December 16 and those 18 years of age or older residing in the US and Canada are eligible to enter

Enter at https://developer.ibm.com/contest

Once per week during the 5 weeks of the contest a random drawing will be held to determine the winners (watch for the drawing on Twitch)

Winners will receive a DJI Tello programmable drone, an IBM Developer T-shirt, and an IBM Developer laptop sticker

Winners will also receive a special code to unlock more challenges and given the opportunity to contribute back to the challenge

#IBMDroneDrop
Agenda

• Why Machine Learning?
• Some History
• PMML Internals
• PMML in Python and R
• PMML in IBM products
• PFA
• Other tools
Why is machine learning (ML) so popular now?

Data Scientists are in high demand

Data Science = ML + Business Domain Knowledge

ML is everywhere!

Actuarial Science was first ML
Examples of ML around us

Weather forecast

Chat bots, Alexa, Siri

Identifying fraud in banks, credit cards

Online shopping recommendations

Pattern recognition, Spam filters

Computer vision and speech recognition

Watson playing Jeopardy!
Frequently used terms

**Machine learning** is "learning" from data, or generalizing from examples. **Structured data** - highly organized information that can be stored in row database structures. Columns are **fields** or **variables**, can be **categorical (nominal, ordinal)** or **continuous**.

www.cubicsol.com/machine-learning-algorithms/
Some ML models

- Clustering
- Linear regression
- Logistic regression
- Decision tree

Neural network (Multi-Layer perceptron)

Deep learning
The Iris dataset

- Data set from UCI ML data depository
- 3 classes of iris flower: Setosa, Versicolor, Virginica, 50 cases each
- Four continuous attributes: sepal length, sepal width, petal length, petal width
Typical Stages in Machine Learning

1. Collect Data
2. Analyze and Clean Data
3. Transform data
4. Build a Model
5. Deploy the model
6. Monitor and update as needed
Typical Stages in Machine Learning

1. Collect Data
2. Analyze and Clean Data
3. Transform Data
4. Build a Model
5. **Monitor** and update as needed
6. Deploy the model
Model Deployment Challenges

Environments
- OS and File Systems
- Databases, desktop vs cloud

Languages
- Python or R, various packages, C++ or Java or Scala, Dependencies and versions

Teams
- Data Scientists and statisticians, application developers and IT

Data Prep
- Aggregation and joins
- Normalization, Category Encoding, Binning, Missing value replacement
DMG to the rescue!

Data Mining Group

dmg.org

The Data Mining Group is a consortium managed by the Center for Computational Science Research, Inc., which is an Illinois based 501(c)(3) not-for-profit corporation

Founded in late 1990’s by Professor Robert Grossman
What is PMML?

Predictive Model Markup Language

• An Open Standard for XML Representation
• Developed by DMG

• Over 30 vendors and organizations

• PMML 4.4 Release manager: Svetlana Levitan

dmg.org/pmml
Brief History of PMML versions

- **0.7 in 1997**
  - First

- **1.1 in 2000**
  - Six models

- **2.0 in 2001**
  - Transformations
    - Naïve Bayes, Sequence

- **3.0 in 2004**
  - Functions
    - Output Composition
  - SVM, Ruleset

- **4.0 in 2009**
  - Ensembles, Cox, Time Series, Model Explanation

- **4.4 in 2018**
  - More Time Series, BN, Gaussian Process
Main Components of PMML

- Header
- Data Dictionary
- Transformation Dictionary
- Model(s)
PMML under the hood

- **Header**
  - Application name and version
  - Timestamp and copyright

- **Data Dictionary**
  - Field names and labels, values
  - Data type and measurement level

- **Transformation Dictionary**
  - Define Function
  - Derived Fields

- **Model(s)**
  - Mining Schema
  - Specific model contents
Transformations

- **NormContinuous**: piece-wise linear transform
- **NormDiscrete**: map a categorical field to a set of dummy fields
- **Discretize**: binning
- **MapValues**: map one or more categorical fields into another categorical one
- **Functions**: built-in and user-defined
- **Other transformations**
PMML Models

- 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
Contents of a PMML Model

- **Mining Schema**: target and predictors, importance, missing value treatment, invalid value treatment, outlier treatment

- **Output**: what to report, post-processing

- **Model Stats**: description of input data

- **Model Explanation**: model diagnostics, useful for visualization

- **Targets**: target category info and prior probabilities

- **Local Transformations**: predictor transformations local to the model

- **...<Specific model contents>...**

- **Model Verification**: expected results for some cases
Specific contents of some models

- **Clustering**
  - ClusteringField, Comparison Measure, Cluster

- **Neural network**
  - NeuralInputs, NeuralLayer, Neuron, NeuralOutputs, Con

- **Regression**
  - RegressionTable
  - NumericPredictor, Categorical Predictor

- **Tree Model**
  - Node, Predicates, Score Distribution
An example PMML – Data Dictionary, Transformations

```xml
<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>
</DerivedField>
```
Example PMML – Neural Network MiningSchema and inputs

```xml
<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>
  </NeuralInputs>
</NeuralNetwork>
```

Predictors
Example PMML - Neural Network hidden layer and outputs

```xml
<NeuralLayer numberOfNeurons="3" activationFunction="identity" normalizationMethod="softmax">
  <Neuron id="1" 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>
  <Neuron id="2" bias="0.101922887283541">
    <Con from="4" weight="-1.05699048855012"/>
    <Con from="5" weight="2.0022889916664"/>
    <Con from="6" weight="3.3127837496491"/>
  </Neuron>
  <Neuron id="3" bias="0.917636281284728">
    <Con from="4" weight="-1.47230776836775"/>
    <Con from="5" weight="0.905795272070893"/>
    <Con from="6" weight="-1.60793177845373"/>
  </Neuron>
</NeuralLayer>
```

**Hidden layer neuron**

- **Output Layer Neurons**

- **Connecting target to the neurons**

Diagram of neural network connections.
Example PMML for a Tree Model

```xml
<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-verseicol" 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-verseicol" recordCount="40.0">
      <SimplePredicate field="petal_length" operator="lessOrEqual" value="4.75"/>
    </Node>
  </Node>
</Node>
```
Mining Model

Segmentation

Segment

Predicate, weight

MODEL

Predicate, weight

MODEL

Multiple model method

Majority vote, average, median, max, sum, selectFirst, modelChain
## PMML Powered


<table>
<thead>
<tr>
<th>Alpine Data</th>
<th>JPMML</th>
<th>R</th>
</tr>
</thead>
<tbody>
<tr>
<td>Angoss</td>
<td>KNIME</td>
<td>Salford Systems (Minitab)</td>
</tr>
<tr>
<td>BigML</td>
<td>KXEN</td>
<td>SAND</td>
</tr>
<tr>
<td>Equifax</td>
<td>Liga Data</td>
<td>SAS</td>
</tr>
<tr>
<td>Experian</td>
<td>Microsoft</td>
<td>Software AG (incl. Zementis)</td>
</tr>
<tr>
<td>FICO</td>
<td>MicroStrategy</td>
<td>Spark</td>
</tr>
<tr>
<td>Fiserv</td>
<td>NG Data</td>
<td>Sparkling Logic</td>
</tr>
<tr>
<td>Frontline Solvers</td>
<td>Open Data</td>
<td>Teradata</td>
</tr>
<tr>
<td>GDS Link</td>
<td>Opera</td>
<td>TIBCO</td>
</tr>
<tr>
<td>IBM (Includes SPSS)</td>
<td>Pega</td>
<td>WEKA</td>
</tr>
<tr>
<td></td>
<td>Pervasive Data Rush</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Predixion Software</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Rapid I</td>
<td></td>
</tr>
</tbody>
</table>
Agenda

- Why Machine Learning?
- Some History
- PMML Internals
- PMML in Python and R
- PMML in IBM products
- PFA
- Other tools
PMML in Python

JPMML package is created and maintained by Villu Ruusmann.

From https://stackoverflow.com/questions/33221331/export-python-scikit-learn-models-into-pmml

```
pip install git+https://github.com/jpmml/sklearn2pmml.git
```

**Example of how to export a classifier tree to PMML. First grow the tree:**

```
from sklearn import datasets, tree
iris = datasets.load_iris()
clf = tree.DecisionTreeClassifier()
clf = clf.fit(iris.data, iris.target)
```

**SkLearn2PMML conversion takes 2 arguments: an estimator (our clf) and a mapper for preprocessing. Our mapper is pretty basic, since no transformations.**

```
from sklearn_pandas import DataFrameMapper
default_mapper = DataFrameMapper([(i, None) for i in iris.feature_names + ['Species']])
from sklearn2pmml import sklearn2pmml
sklearn2pmml(estimator=clf, mapper=default_mapper, pmml="IrisClassificationTree.pmml")
```
PMML in R

R is a programming language and software environment for statistical computing and graphics supported by the R Foundation for Statistical Computing.

R packages “pmml” and “pmmlTransformations”

https://cran.r-project.org/package=pmml

Depend on XML package

Supports a number of R models: ada, amap, arules, gbm, glmnet, neighbr, nnet, rpart, randomForest, kernlab, e1071, testthat, survival, xgboost, knitr, rmarkdown

Maintained by Dmitriy Bolotov and Tridivesh Jena from Software AG

JPMML also has a package that augments “pmml” and provides PMML export for additional R models
Create PMML in R (using R Studio)

```r
> library(XML);
> library(pmml);
> data(iris);

Build and save a linear regression model predicting Sepal length:
> irisLR<-lm(Sepal.Length~.,iris)
> saveXML( pmml(irisLR), "IrisLR.xml" )

Build and save a decision tree (C&RT) model predicting Species class:
> irisTree <- rpart( Species~., iris )
> saveXML( pmml( irisTree ), "IrisTree.xml" )
```
Agenda

• Why Machine Learning?
• Some History
• PMML Internals
• PMML in Python and R
• PMML in IBM products
• PFA
• Other tools
IBM SPSS Statistics

1968
Statistical Package for Social Sciences

Acquired by IBM in 2009

Release 25 in August 2017
Subscription option

Integration with Python and R
**IBM SPSS Statistics**

*Transformation PMML from:*
ADP (Automatic Data Preparation)
TMS Begin/TMS End

*Model PMML from:*
COXREG, CSCOXREG
CSGLM, CSLOGISTIC, CSORDINAL
GENLIN, Logistic regression, NOMREG
GENLINMIXED
LINEAR, KNN
MLP, RBF neural networks
NAÏVE BAYES
REGRESSION
TREE, TSMODEL
TWOSTEP CLUSTER

**IBM SPSS Modeler**

Apriori, CARMA, Association Rules
C5, CART, Chaid decision trees
Cox regression
GENLIN
Decision List
K-Means Cluster
KNN
LINEAR, Regression
Logistic Regression
MLP and RBF
NOMREG
Random Trees
Regression
Two Step Cluster
Requesting PMML export in a decision tree analysis and Bayesian regression in IBM SPSS Statistics
Score PMML in IBM SPSS Statistics

Utilities->Scoring Wizard
Watson Studio (formerly Data Science Experience)

PMML export enabled in Jupyter notebooks, also possible in R Studio. PMML scoring can be done in Flows, notebooks, Watson Machine Learning.
Using SPSS Two Step Cluster model in Python notebook

Create a Jupyter notebook with Spark service
Load the data into Cloud Object Storage, then into the notebook as a Spark data frame, specifying which fields are numeric-valued

```python
from spss.ml.clustering.twostep import TwoStep
from spss.ml.clustering.twostep import TwoStepModel

cluster = TwoStep().
    setInputFieldList(["sepal_length","sepal_width","petal_length","petal_width"]).
    setDistMeasure("LOGLIKELIHOOD").
    setFeatureSelection(False).
    setAutoClustering(True)

clusterModel = cluster.fit(df_data_1)
cePMML = clusterModel.toPMML()
```
Watson Studio Flows
Scoring PMML in Watson Machine Learning

IBM Watson

PMML Scorer

Overview  Evaluation  Deployments  Lineage

Summary
- Machine learning service: pm-20-dix
- Model Type: pmml4.5
- Runtime environment: java-1.8
- Training date: 11-Sep-2015, 4:48PM
- Label column: Sepal
- Latest version: fcd4ab05-6525-4550-a518-8510b5dd10e50

Input Schema
- COLUMN  TYPE
  - Sepal.Length  double
  - Sepal.Width  double
  - Petal.Length  double
  - Petal.Width  double

PMML Scorer

Implementation

Code Snippets

URL  Java  JavaScript  Python  Scala

```
import urlib3, requests, json

# retrieves your url_service_credentials_username, url_service_credentials_password
# service credentials associated with your IBM Cloud Watson Machine Learning

url_service_credentials={
    "url": url_service_credentials_url,
    "username": url_service_credentials_username,
    "password": url_service_credentials_password
}

headers = urlib3.util.make_headers(basic_auth=(username,password))
form_url = '/api/v1/services/1/models/1/deployments/1/predict
response = requests.get(url, headers=headers, timeout=30)
response_json = json.loads(response.text)
response_json.get('token')
```

DeployIrisTree

Enter input data

- Sepal.Length: 4
- Sepal.Width: 4
- Petal.Length: 4
- Petal.Width: 4

Select Predict
Benefits of PMML

- Allows seamless deployment and model exchange
- Transparency: human and machine-readable
- Fosters best practices in model building and deployment
Agenda

• Why Machine Learning?
• Some History
• PMML Internals
• PMML in Python and R
• PMML in IBM products
• PFA
• Other tools
Portable Format for Analytics - PFA

PMML is great, except when a model or feature is not supported

PFA to overcome this

JSON format, AVRO schemas for data types

A mini functional math language + schema specification

Info: [dmg.org/pfa](http://dmg.org/pfa)

Jim Pivarski
PFA details

• PFA consists of:
  • JSON serialization format
  • AVRO schemas for data types
  • Encodes functions (actions) that are applied to inputs to create outputs with a set of built-in functions and language constructs (e.g. control-flow, conditionals)

• Built-in functions and common models

• Type and function system means PFA can be fully & statically verified on load and run by any compliant execution engine

• Portability across languages, frameworks, run times and versions
A Simple Example of PFA (copied from Nick Pentreath’s presentation)

- Example – multi-class logistic regression
- Specify input and output types using Avro schemas

```json
{
  "name": "logistic-regression-model",
  "input": {
    "type": {
      "type": "array",
      "items": "double"
    }
  },
  "output": {
    "type": "double"
  }
}
```

- Specify the `action` to perform (typically on input)

```json
"action": {
  "a.argmax": {
    "m.link.softmax": {
      "model.reg.linear": {
        "input": {
          "cell": "model"
        }
      }
    }
  }
}
```
Managing State in PFA (copied from Nick Pentreath’s presentation)

- Data storage specified by **cells**
  - A cell is a named value acting as a global variable
  - Typically used to store state (such as model coefficients, vocabulary mappings, etc)
  - Types specified with Avro schemas
  - Cell values are mutable within an action, but immutable between action executions of a given PFA document

- Persistent storage specified by **pools**
  - Closer in concept to a database
  - Pools values are mutable across action executions
Known Support for PFA

**Hadrian** (PFA export and scoring engine)
from Open Data Group (Chicago, IL)

**Aardpfark** (PFA export in SparkML)
by Nick Pentreath, IBM CODAiT, South Africa

**Woken** (PFA export and validation)
by Ludovic Claude, CHUV, Lausanne, Switzerland

There is a lot of interest in PFA!

If you want to help, let me know. Starting OS projects soon.
Other model deployment formats

**Pickle** in Python – binary serialization of Scikit Learn models

**MLeap** - open format, not a standard, uses protobuf. Works for Spark ML models

**Open Neural Network Exchange** (ONNX) from Microsoft and Facebook
- binary (protobuf) format for deep learning models
- describes computation graph (including operators)
- supported by most deep learning frameworks (TF still in progress)
- now adding support for traditional ML

**Neural Network Exchange Format** (NNEF) by Khronos Group
- Dual format (weights in binary file)
- Allows user-defined functions
- Only neural network models
ONNX use pattern

Frontend
Models in different frameworks

Export

ONNX IR Spec .onnx

Import

Backend
Models in different frameworks

Tools
Netron visualizer
Net Drawer visualizer
Checker
Shape Inferencer
Graph Optimizer
Opset Version Converter

Training

Inference
Docker containers: https://docs.docker.com/engine/docker-overview/

Isolated and secure environments

**Docker Engine** is a client-server application with these major components:

- A server which is a type of long-running program called a daemon process (the dockerd command).
- A REST API which specifies interfaces that programs can use to talk to the daemon and instruct it what to do.
- A command line interface (CLI) client (the docker command).
IBM Code Model Asset eXchange (MAX)

Curated repository containing ready-to-use open-source deep-learning models used for image processing, audio processing, and more.

- Pre-trained models include provenance (origin, references, and licensing terms)
- Delivered in Docker containers, exposing
  - Lightweight REST API
  - Swagger API documentation
- ibm.biz/max-models
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

PFA is an emerging standard that needs your help

ONNX and Docker are often used for Deep Learning
Questions?
Connect with me:

LinkedIn: https://www.linkedin.com/in/svetlanalevitan/
Twitter: @SvetaLevitan   Email: slevitan@us.ibm.com

Useful links:

Drone challenge: https://developer.ibm.com/contest
PMML    dmg.org/pmml
PFA      dmg.org/pfa
IBM CODAIT: codait.org
SPSS: https://www.ibm.com/analytics/spss-statistics-software
Watson Studio: https://www.ibm.com/cloud/watson-studio
Thank you.