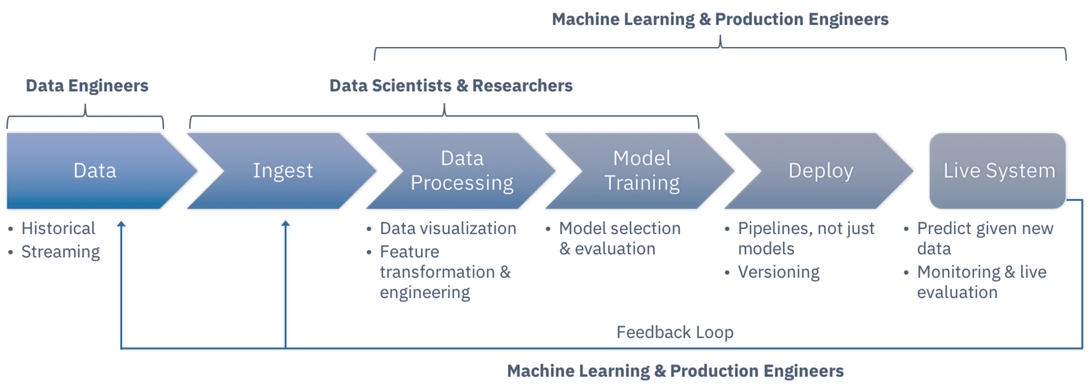
Open standards for deployment, storage and sharing of predictive models

PMML / PFA / ONNX in action

Svetlana Levitan, IBM CODAIT and DMG, Chicago, IL, USA; Nick Pentreath, IBM CODAIT, South Africa
Ludovic Claude, CHUV, Lausanne, Switzerland (@SvetaLevitan, @mlnick, @ludovicc)

The Machine Learning Pipeline



Machine learning piplelines span organizational teams and tools. Challenges include:
- Bridge various languages, frameworks, runtimes, versions
- Friction between teams - data science vs production vs business
- Proliferation of formats - lack of standardization leads to custom solutions

Predictive Model Markup Language (PMML)

- An Open Standard for XML Representation of models
- Developed by DMG since about 1997
- Supported by over 30 vendors and organizations

PMML Models: 16 models + ensembles/compositions
Supporting open source: JPMML, Augustus, R pmml
Supporting IBM products: IBM SPSS Statistics, IBM SPSS Modeler, Watson Studio
Uses of PMML: model and/or data preparation deployment and exchange, model evaluation, visualization, simulation, explanation.

```
<PMML xmlns="http://www.dmg.org/PMML-4_3" version="4.3"> <Header copyright="DMG.org"/>
  <DataDictionary> <DataField name="lefthippocampus" optype="continuous" dataType="double"/>
    <DataField name="contTarget" optype="continuous" dataType="double"/>
  </DataDictionary>
  <RegressionModel modelName="Alzheimer and hippocampus" functionName="regression">
    <MiningSchema> <MiningField name="lefthippocampus"/>
      <MiningField name="contTarget" usageType="target"/>  </MiningSchema>
    <Output>
      <OutputField name="RawResult" optype="continuous" dataType="double" feature="predictedValue"/>
      <OutputField name="Final" optype="categorical" dataType="string" feature="transformedValue">
        <Discretize field="RawResult" dataType="string" >
          <DiscretizeBin binValue="AD"> <Interval closure="openOpen" rightMargin="1.001"/>
          </DiscretizeBin> <DiscretizeBin binValue="Other">
            <Interval closure="closedOpen" leftMargin="1.001" rightMargin="1.203"/>
          </DiscretizeBin> <DiscretizeBin binValue="CN">
            <Interval closure="closedOpen" leftMargin="1.203"/> </DiscretizeBin>
        </Discretize>  </OutputField>
    </Output>
    <RegressionTable intercept="0.997">
      <NumericPredictor name="lefthippocampus" exponent="1" coefficient="0.1"/>
    </RegressionTable> </RegressionModel> </PMML>
```

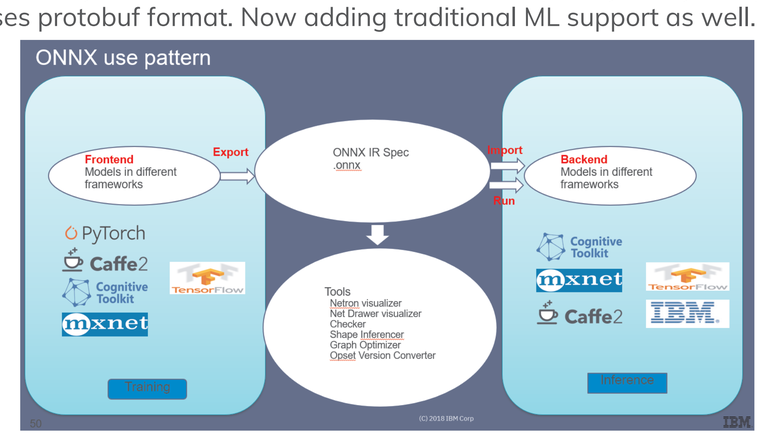
Portable Format for Analytics (PFA)

- An Open Standard for flexible JSON representation of models
- JSON and AVRO based, a mini programming language
- Developed by DMG since 2015, currently at version 0.8.1

Supporting open source:
Hadrian, Titus, Aurelius (PFA export and scoring engine) Open Data Group (Chicago)
Aardpfark (PFA export in SparkML) by Nick Pentreath,
Woken (PFA export and validation) by Ludovic Claude
sklearn_to_pfa (PFA export from Python) by Mojmir Vinkler

Open Neural Network Exchange (ONNX)

De-facto standard for deep learning models, supported by most DL frameworks.
Started by Microsoft and Facebook, now developed by many companies.
Uses protobuf format. Now adding traditional ML support as well.



Open Standards for Model Serialization

Open standards for model serialization provide ways to easily deploy or exchange models between different products or systems regardless of programming languages, operating systems, file systems.

The Data Mining Group (DMG) was created in 1990's to work on such open standards. See dmg.org

Mainstream open standards for model serialization and deployment include:
- Predictive Model Markup Language (PMML)
- Portable Format for Analytics (PFA)
- Open Neural Network Exchange (ONNX)

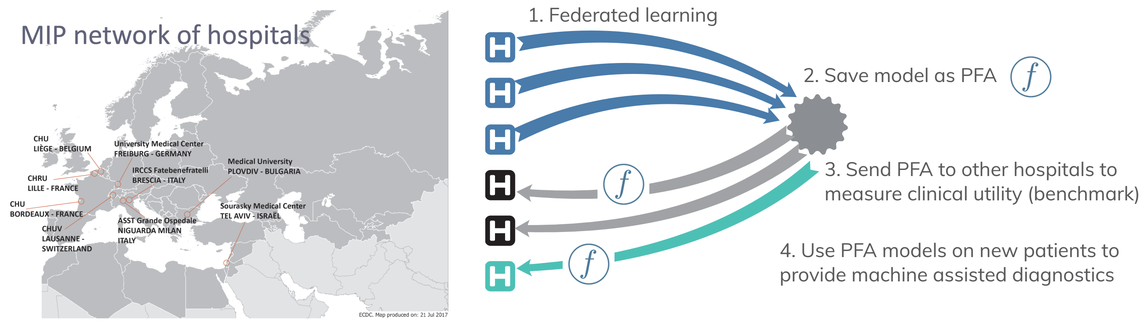
Application: PFA Models in Brain Research

Co-funded by the European Union

@HBPmedical https://humanbrainproject.eu/Medical

The Medical Informatics platform of the Human Brain Project is deployed in 10+ hospitals in EU and provides privacy-preserving machine learning and analytics services.

Aims: Discover better signatures of brain disease, assist clinicians in their diagnostic of brain disease, while protecting patient privacy and consent.



Usage and Benefits of PFA for the platform

- Scientist write Machine learning algorithms in Python, R, Matlab, Java. Algorithms are packaged in Docker, adapted to export models as PFA, and deployed in hospitals
- On each node, we can verify quality of machine learning using the PFA model and a cross-validation algorithm
- PFA models are collected in the central node, then stored in a database for sharing and reuse.
- PFA models can be shared between hospitals, a hospital can build models from its data then test the model onto another hospital without revealing or exchanging any of its patient data.

Sample PFA model generated in MIP from clinical data

```
name: linear_model
description: Naive predictor of Alzheimer disease from volume of hippocampus
input:
 fields: [{name: lefthippocampus, type: double}]
output: string # Alzheimer's disease diagnostic
metadata:
 coef_: "[[0.1],[0.3],[0.2]]"
 intercept_: "[-1.001,-1.002,0.997]"
 score: '0.6670547148'
action:
- let:
  x: {a.flatten:
       {new: { u.arr: { cast.double: {attr: input, path: {string: "lefthippocampus"}}},
       type: { items: {items: double, type: array}, type: array}}
  scores: {a.map: [{cell: model},
            do: {model.reg.linear: [x,r], params: [...], ret: double}}
- {cell: classes, path: [a.argmax: [scores]]}
cells:
 classes: {init: [AD, CN, Other],  type: {items: string, type: array}}
 model:
   init: [{coeff:[0.1],const:-1.001},{coeff:[0.3],const:-1.002},
          {coeff:[0.2],const:0.997}]
   type: {items: Regression, type: array}
fcns: {C: [...],  arr: [...],  standardize: [...]}
```

Towards FAIR machine learning models

Findability: A DOI should be generated for each model. PFA model describes input and output of predictive algorithms. It needs to be extended with additional annotations (ontologies), then registered in a search engine that can leverage its metadata, e.g. Blue Brain Nexus.
Accessibility: PFA models are standard JSON documents.
Interoperability: The algorithm described in PFA can be interpreted in any programming language. Currently, Python, Scala and R interpreters exist
Reusability: Add license and provenance information to PFA document