# Architecture of ML Systems (AMLS)
# 12 Model Deployment and Serving

**Prof. Dr. Matthias Boehm**
Technische Universität Berlin
Berlin Institute for the Foundations of Learning and Data
Big Data Engineering (DAMS Lab)



Last update: Jul 05, 2024

# Announcements / Org

- **#1 Hybrid & Video Recording**
  - Hybrid lectures (in-person, zoom) with optional attendance
    https://tu-berlin.zoom.us/j/9529634787?pwd=R1ZsN1M3SC9BOU1OcFdmem9zT202UT09
  - Zoom **video recordings**, links from website
    https://mboehm7.github.io/**teaching**/ss24_amls/index.htm

- **#2 Course Evaluation**
  - Full lecture/exercise evaluation forms shared on ISIS
  - **Lectures:** 1.9
  - **Exercise:** 1.7

# Course Evaluations (Lectures, 10 Evals)

6. Gesamturteil

6.2) Gibt es etwas, das der*die Lehrende im Hinblick auf die Lehr- und Lernmaterialien verbessern sollte?

- I really love that the videos are uploaded, which allow to manually go through the material and watch the videos that fit our schedule.
- I think it is great that the professor, allows all modalities (presence, zoom, video, pdf) to follow the lecture.
- Sometimes, the lecture slides appear to be overloaded. For a student, it is not fully clear what is essential and what not. For example, many slides have snippets of code (DSL, TensorFlow etc.) and it is not clear if we need to memorize these snippets. Further, the slides are not really self-contained (just by reading the slides one does not get the content, the audio is required).
- vielleicht noch mehr Struktur und Orientierung in den Slides z.B. durch mehr Herleitungen, Kapitelangaben, einheitlichen Überschriften etc? Ohne die Videos ist es manchmal schwierig den Gedanken in den Slides zu folgen. Aber das ist eine Beschwere auf sehr hohem Niveau – im Vergleich zu anderen Veranstaltungen sind die Slides sehr gut!

6.7) Gibt es etwas, das der*die Lehrende von anderen Veranstaltungen lernen könnte?

- I think other classes focus more on the really essential points. When sitting in the lecture, I have the feeling that we touch upon a lot of points superficially making it at times challenging to follow the instructor. At some point one is just lost and after leaving the class I find it hard to tell what the core points were.
- Vorlesungen sind zu dicht vom Stoff. Zeitmanagement (es wird immer überzogen). Es ist nicht klar, was vom Stoff essenziell ist und was nebensächlich ist.

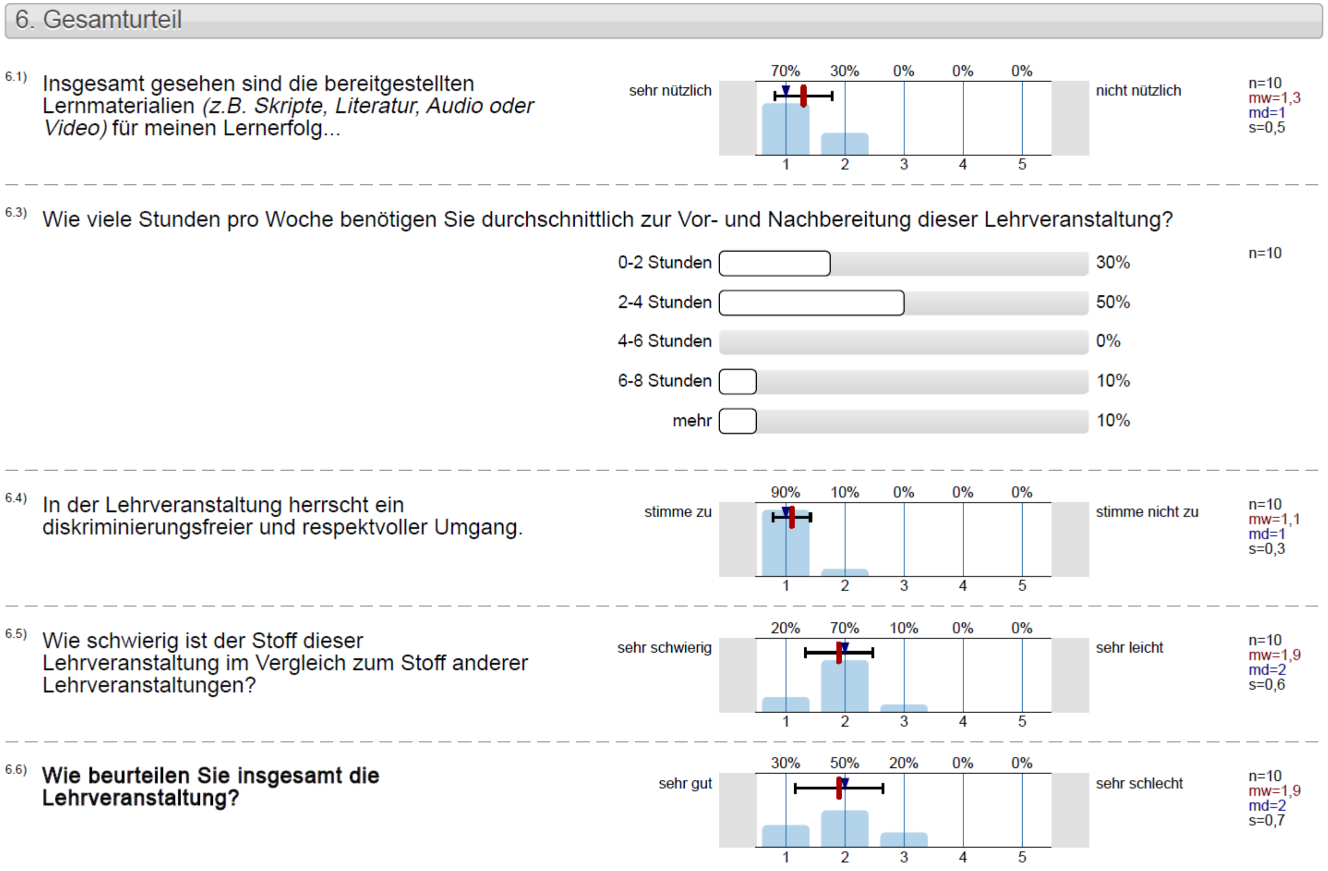
**R1: Keep Multi-modal Channels**

**R2: More Structure**

**R3: Less is More**

# Course Evaluations (Lectures, 10 Evals), cont.

- It's not clear how to prepare for exam, what to expect.
- I think sometimes less content would yield in more learning. Especi
hard to understand new concepts right away.
- I think the way the class is overall organized (relationship of lecture
appears that the lecture and the project (exercise or open source p
of our semester solving the exercise/project for which we actually d
memorizing the slides. This gap is not unusual at TU Berlin, but mo
for the assignments during the semester. While this class touches u
and theoretical. From a student perspective, I think that the ratio be
more balanced. Thank you for taking the time reading this.
- Manche Themen und insbesondere Syntax wird sehr schnell einget
- Thank you for recording and zooming the lecture. This makes it pos
Professors should use this approach as well!
- The lecturer speaks too fast




**R4: Improve Presentation Skills**

**R5: Improve Connection of Lectures and Exercises**

**R6: Exam Prep and Example Exams**

# Course Evaluations (Exercises, 6 evals)

## 6. Gesamturteil

6.2) Gibt es etwas, das der*die Lehrende im Hinblick auf die Lehr- und Lernr

- sentinel dataset is not a nice source for the data acquisition

6.7) Gibt es etwas, das der*die Lehrende von anderen Veranstaltungen lerne

- The first task took the most time but did not yield a lot of points

## 7. Demographische Angaben

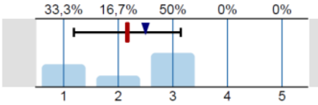
7.2) Wenn "anderer Studiengang" bitte hier angeben welcher:

Es wird keine Auswertung angezeigt, da die Anzahl der Antworten zu gering

7.6) **Wollen Sie uns zu der Lehrveranstaltung noch etwas mitteilen (z.B. (**
**Lehrveranstaltung)?**

## 6. Gesamturteil

6.1) Insgesamt gesehen sind die bereitgestellten Lernmaterialien *(z.B. Skripte, Literatur, Audio oder Video)* für meinen Lernerfolg...

sehr nützlich


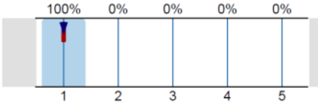
nicht nützlich

n=6
mw=2,2
md=2,5
s=1

6.3) Wie viele Stunden pro Woche benötigen Sie durchschnittlich zur Vor- und Nachbereitung dieser Lehrveranstaltung?

| | | |
|---|---|---|
| 0-2 Stunden | 33,3% | n=6 |
| 2-4 Stunden | 16,7% | |
| 4-6 Stunden | 50% | |
| 6-8 Stunden | 0% | |
| mehr | 0% | |

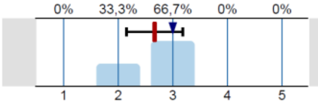
6.4) In der Lehrveranstaltung herrscht ein diskriminierungsfreier und respektvoller Umgang.

stimme zu


stimme nicht zu

n=6
mw=1
md=1
s=0

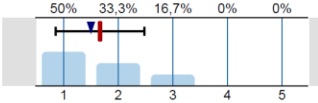
6.5) Wie schwierig ist der Stoff dieser Lehrveranstaltung im Vergleich zum Stoff anderer Lehrveranstaltungen?

sehr schwierig


sehr leicht

n=6
mw=2,7
md=3
s=0,5

6.6) **Wie beurteilen Sie insgesamt die Lehrveranstaltung?**

sehr gut


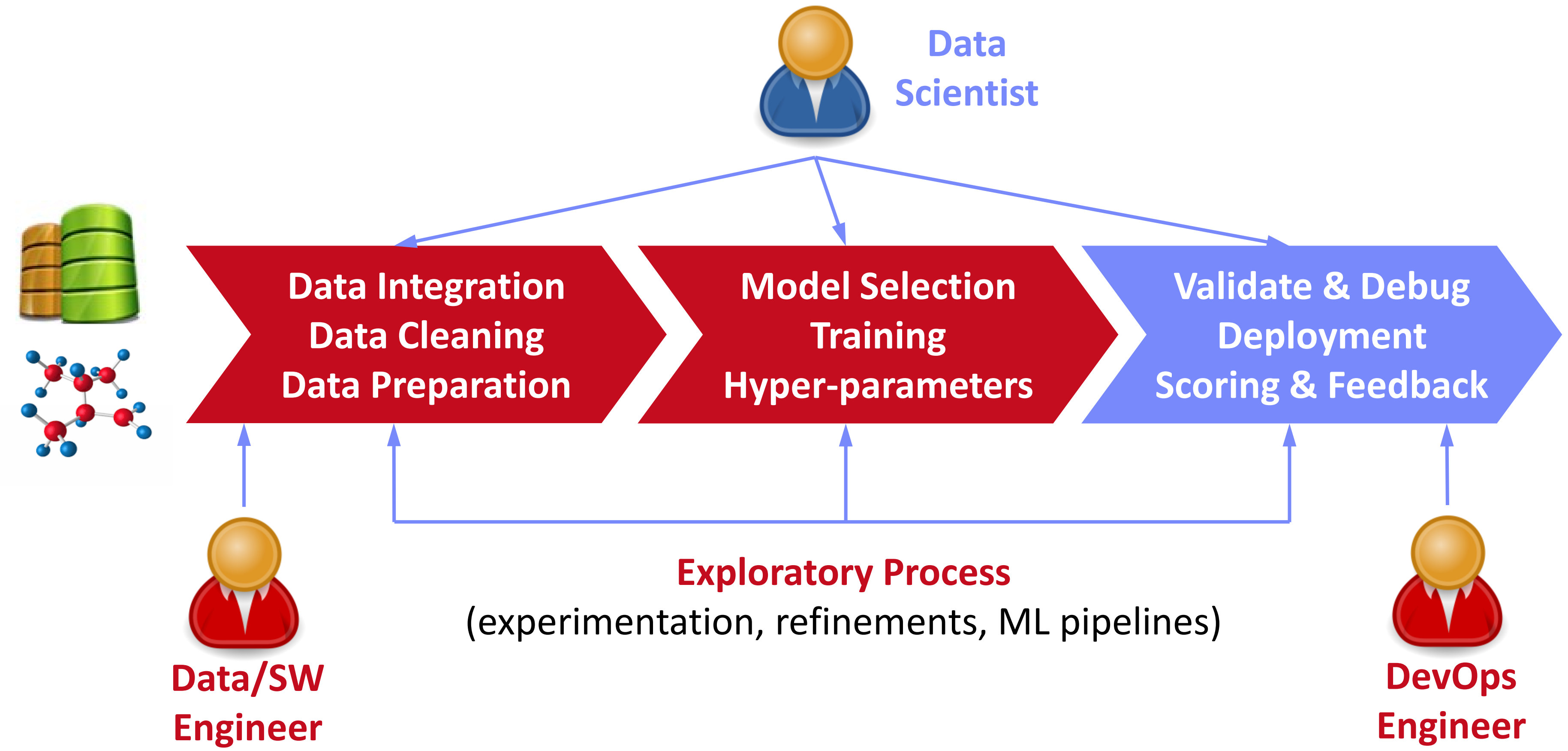
sehr schlecht

n=6
mw=1,7
md=1,5
s=0,8

- I am doing the exercise, and I really like that we have to implement a machine learning pipeline end to end. It is fun and good practice, but it would be nice to get some access to faster hardware especially for training new models.

# Recap: The Data Science Lifecycle (aka KDD Process, aka CRISP-DM)

**Data-centric View:**
Application perspective
Workload perspective
System perspective



# Agenda

- **Model Exchange and Serving**
- **Model Monitoring and Updates**

# Model Exchange and Serving

# Model Exchange Formats

- **Definition Deployed Model**
  - **#1 Trained ML model** (weight/parameter matrix)
  - **#2 Trained weights AND operator graph** / entire ML pipeline
    - ➔ especially for DNN (many weight/bias tensors, hyper parameters, etc)

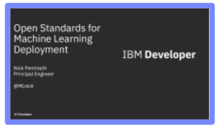
- **Recap: Data Exchange Formats** (model + meta data)
  - General-purpose formats: **CSV**, **JSON**, **XML**, **Protobuf**
  - Sparse matrix formats: **matrix market**, **libsvm**
  - Scientific formats: **NetCDF**, **HDF5**
  - ML-system-specific binary formats (e.g., SystemDS, PyTorch serialized)

```
%%MatrixMarket matrix coordinate real general
% ---------------------------------------------
% 0 or more comment lines
% ---------------------------------------------
5  5  8
1 1 1.000e+00
2 2 1.050e+01
3 3 1.500e-02
1 4 6.000e+00
4 2 2.505e+02
4 4 -2.800e+02
4 5 3.332e+01
5 5 1.200e+01
```

PYTORCH

- **Problem ML System Landscape**
  - Different languages and frameworks, including versions
  - Lack of standardization → **DSLs for ML is wild west**

# Model Exchange Formats, cont.




[Nick Pentreath: Open Standards for Machine Learning Deployment, **bbuzz 2019**]

- **Why Open Standards?**
  - Open source allows inspection but no control
  - Open governance necessary for open standard
  - Cons: needs adoption, moves slowly

- **#1 Predictive Model Markup Language (PMML)**
  - Model exchange format in XML, created by Data Mining Group 1997
  - Package model weights, hyper parameters, and **limited set of algorithms**

- **#2 Portable Format for Analytics (PFA)**
  - Attempt to fix limitations of PMML, created by Data Mining Group
  - JSON and AVRO exchange format
  - **Minimal functional math language** → arbitrary custom models
  - Scoring in JVM, Python, R

# Model Exchange Formats, cont.

- **#3 Open Neural Network Exchange (ONNX)**
  - **Model exchange format** (data and operator graph) via Protobuf
  - First Facebook and Microsoft, then IBM, Amazon → PyTorch, MXNet
  - Focused on **deep learning and tensor operations**
  - ONNX-ML: support for traditional ML algorithms
  - Scoring engine: https://github.com/Microsoft/onnxruntime
  - Cons: **low level** (e.g., fused ops), **DNN-centric** → ONNX-ML

`python/systemds/onnx_systemds`

- **TensorFlow Saved Models**
  - **TensorFlow-specific exchange format** for model and operator graph
  - Freezes input weights and literals, for additional optimizations (e.g., constant folding, quantization, etc)
  - Cloud providers may not be interested in open exchange standards

# ML Systems for Serving

- **#1 Embedded ML Serving**
  - **TensorFlow Lite** and new language bindings (small footprint, dedicated HW acceleration, APIs, and models: MobileNet, SqueezeNet)
  - **TorchScript:** Compile Python functions into ScriptModule/ScriptFunction
  - **SystemML JMLC** (Java ML Connector)
- **#2 ML Serving Services**
  - Motivation: Complex DNN models, ran on dedicated HW
  - RPC/REST interface for applications
  - **TensorFlow Serving:** configurable serving w/ batching
  - **TorchServe:** Specialized model for HW, batching/parallelism
  - **Clipper:** Decoupled multi-framework scoring, w/ batching and result caching
  - **Pretzel:** Batching and multi-model optimizations in ML.NET
  - **Rafiki:** Optimizations for accuracy s.t. latency constraints, batching, multi-model opt

**Google Translate**
140B words/day
→ **82K GPUs** in 2016

**PyTorch TorchServe Config**

```
models={
 "resnet-152": {"1.0": {
  "minWorkers": 1,
  "maxWorkers": 1,
  "batchSize": 8,
  "maxBatchDelay": 50,
  "responseTimeout": 120
}}}
```

[Christopher Olston et al: TensorFlow-Serving: Flexible, High-Performance ML Serving. **ML Systems**@NeurIPS **2017**]

[Daniel Crankshaw et al: Clipper: A Low-Latency Online Prediction Serving System. **NSDI 2017**]

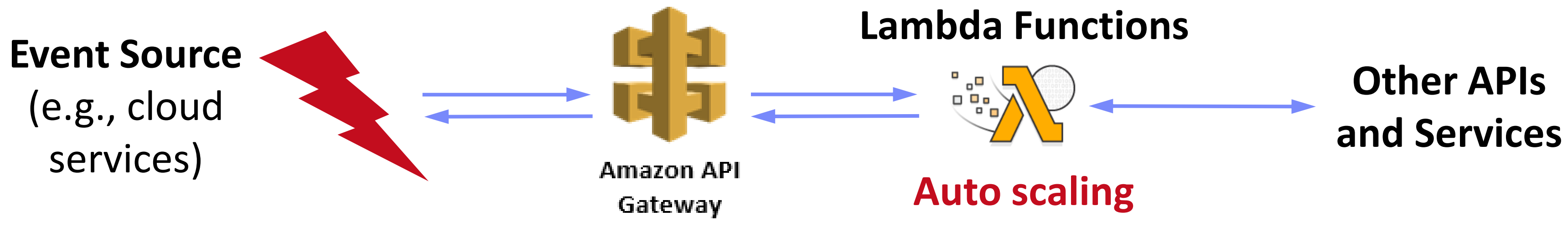
[Yunseong Lee et al.: PRETZEL: Opening the Black Box of Machine Learning Prediction Serving Systems. **OSDI 2018**]

[Wei Wang et al: Rafiki: Machine Learning as an Analytics Service System. **PVLDB 2018**]

# Serverless Computing

[Joseph M. Hellerstein et al: Serverless Computing: One Step Forward, Two Steps Back. **CIDR 2019**]

- **Definition Serverless**
  - **FaaS:** functions-as-a-service (event-driven, stateless input-output mapping)
  - Infrastructure for deployment and auto-scaling of APIs/functions
  - Examples: **Amazon Lambda**, **Microsoft Azure Functions**, etc



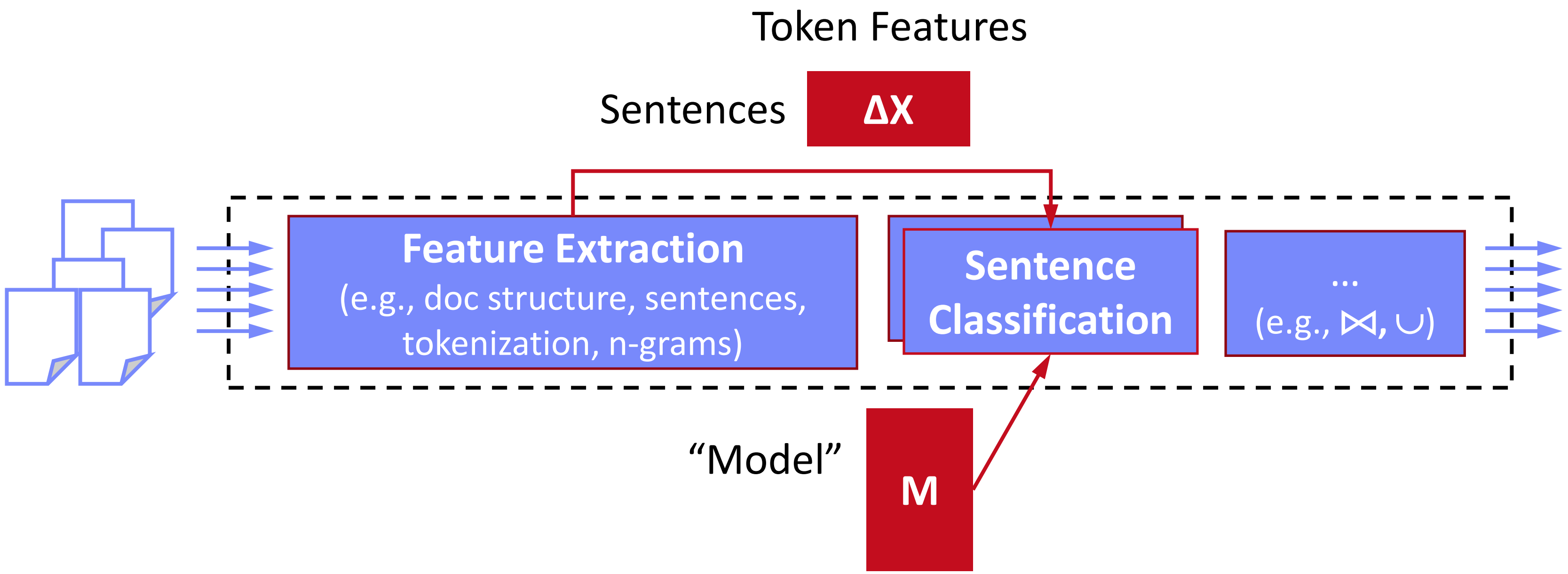

**Auto scaling**
**Pay-per-request**
(1M x 100ms = 0.2$)

- **Example**

```java
import com.amazonaws.services.lambda.runtime.Context;
import com.amazonaws.services.lambda.runtime.RequestHandler;

public class MyHandler implements RequestHandler<Tuple, MyResponse> {
    @Override
    public MyResponse handleRequest(Tuple input, Context context) {
       return expensiveModelScoring(input); // with read-only model
    }
}
```

# Example SystemDS JMLC

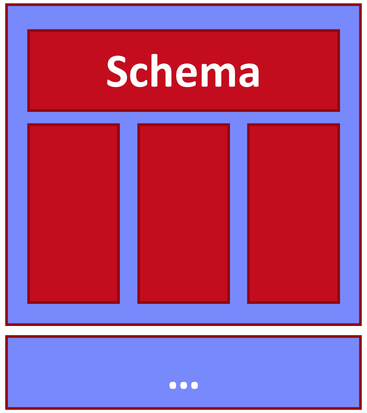
- **Example Scenario**




- **Challenges**
  - Scoring part of larger **end-to-end pipeline** → **Embedded scoring**
  - External parallelization w/o materialization
  - Simple **synchronous scoring** → **Latency ⇒ Throughput**
  - **Data size** (tiny ΔX, huge model M) → **Minimize overhead per ΔX**
  - **Seamless integration** & model consistency → **Token inputs & outputs**

# Example SystemDS JMLC, cont.

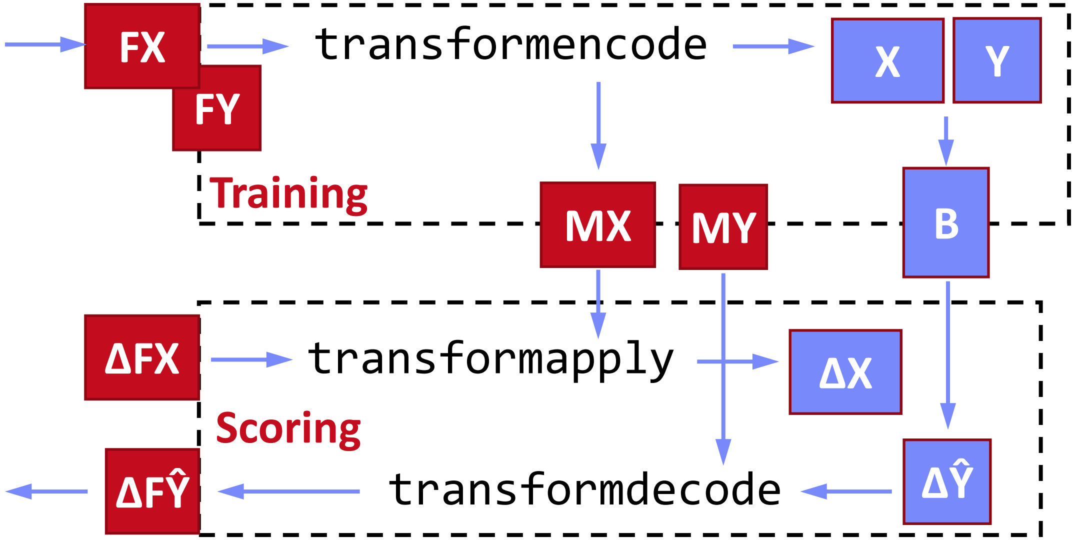
- **Background: Frame**
  - **Abstract data type with schema** (BIN, INT64, FP64, STR)
  - Column-wise block layout, with ragged arrays
  - Local and distributed operations




**Distributed representation:**
? x ncol(F) blocks

**(shuffle-free conversion of csv / datasets)**

- **Data Preparation via Transform**



# Example SystemML JMLC, cont.

- **Motivation**
  - ➔ Embedded scoring
  - ➔ Latency ⇒ Throughput
  - ➔ Minimize overhead per ΔX

➔

**Typical compiler/runtime overheads:**

| | |
|---|---|
| Script parsing and config: | ~100ms |
| Validation, compile, IPA: | ~10ms |
| HOP DAG (re-)compile: | ~1ms |
| Instruction execute: | <0.1µs |

- **Example**

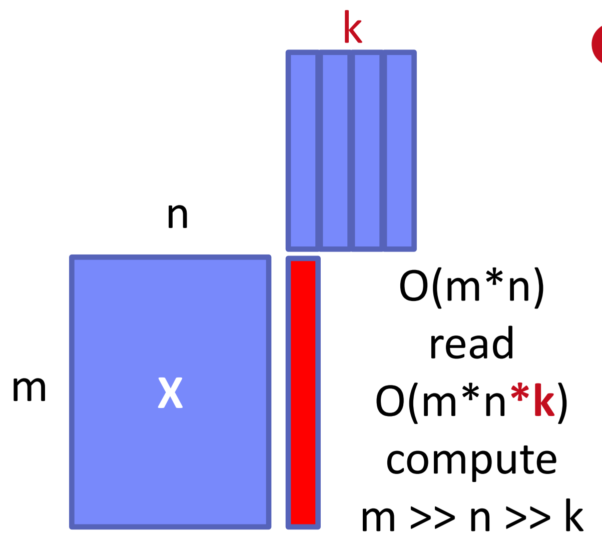
```
Connection conn = new Connection();
PreparedScript pscript = conn.prepareScript(
    getScriptAsString("glm-predict-extended.dml"),
    new String[]{"FX","MX","MY","B"}, new String[]{"FY"});
// ... Setup constant inputs
for( Document d : documents ) {
    FrameBlock FX = ...; //Input pipeline
    pscript.setFrame("FX", FX);
    FrameBlock FY = pscript.executeScript().getFrame("FY");
    // ... Remaining pipeline
}
```

// single-node, no evictions,
// no recompile, no multithread.

// execute precompiled script
// many times

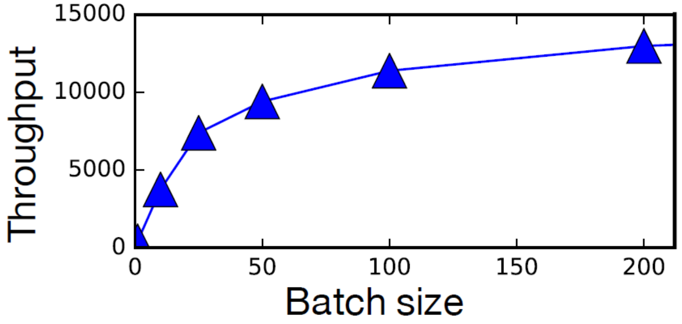
# Serving Optimizations – Batching

- **Recap: Model Batching** (see 08 Data Access)
  - One-pass evaluation of multiple configurations
  - EL, CV, feature selection, hyper parameter tuning
  - E.g.: TUPAQ [SoCC'16], Columbus [SIGMOD'14]

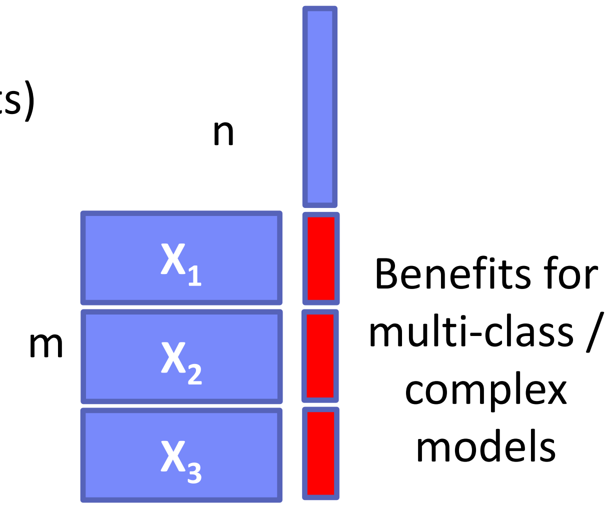



- **Data Batching**
  - Batching to utilize the HW more efficiently under SLA
  - **Use case:** multiple users use the same model (wait and collect requests)
  - **Adaptive:** additive increase, multiplicative decrease

[Clipper @ NSDI'17]




Fewer kernel launches, Parallelization



# Serving Optimizations – Quantization

08 Data Access Methods

- **Quantization**
  - **Lossy compression via ultra-low precision / fixed-point**
  - Ex.: **62.7% energy** spent on data movement

[Amirali Boroumand et al.: Google Workloads for Consumer Devices: Mitigating Data Movement Bottlenecks. **ASPLOS 2018**]

- **Quantization for Model Scoring**
  - Usually **much smaller data types** (e.g., **UINT8**)
  - Quantization of model weights, and sometimes also activations
    → reduced memory requirements and better latency / throughput (SIMD)

```
import tensorflow as tf
converter = tf.lite.TFLiteConverter.from_saved_model(saved_model_dir)
converter.optimizations = [tf.lite.Optimize.OPTIMIZE_FOR_SIZE]
tflite_quant_model = converter.convert()
```

[**Credit:** https://www.tensorflow.org/lite/performance/post_training_quantization ]

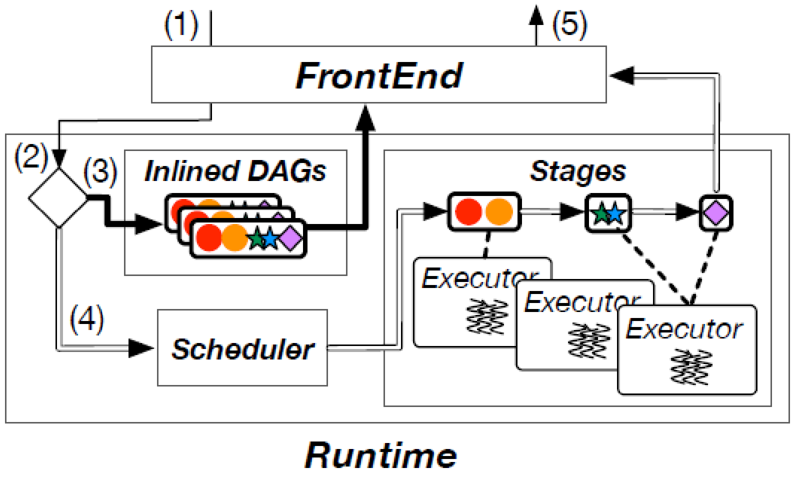
# Serving Optimizations – MQO

- **Result Caching**
  - Establish a function cache for X → Y (memoization of deterministic function evaluation)
  - E.g., translation use case

```
Predict(m: ModelId, x: X) -> y: Y
```

- **Multi Model Optimizations**
  - Same input fed into multiple partially redundant model evaluations
  - Common subexpression elimination between prediction programs
  - In PRETZEL, programs compiled into physical stages and registered with the runtime + caching for stages (decided based on hashing the inputs)

[Yunseong Lee et al.: PRETZEL: Opening the Black Box of Machine Learning Prediction Serving Systems. **OSDI 2018**]



# Serving Optimizations – Compilation

- **TensorFlow `tf.compile`**
  - Compile entire TF graph into binary function w/ low footprint
  - **Input:** Graph, config (feeds+fetches w/ fixes shape sizes)
  - **Output:** x86 binary and C++ header (e.g., inference)
  - Specialization for frozen model and sizes

[Chris Leary, Todd Wang: XLA – TensorFlow, Compiled!, **TF Dev Summit 2017**]

- **PyTorch Compile**
  - Compile Python functions into ScriptModule/ScriptFunction
  - Lazily collect operations, optimize, and JIT compile
  - Explicit `jit.script` call or `@torch.jit.script`

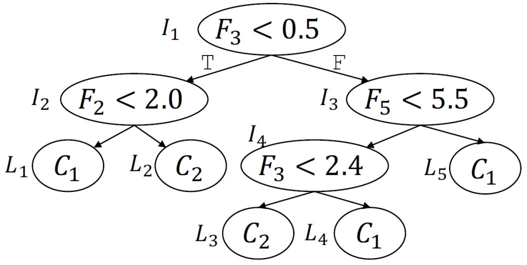
```
a = torch.rand(5)
def func(x):
  for i in range(10):
    x = x * x # unrolled into graph
  return x

jitfunc = torch.jit.script(func) # JIT
jitfunc.save("func.pt")
```

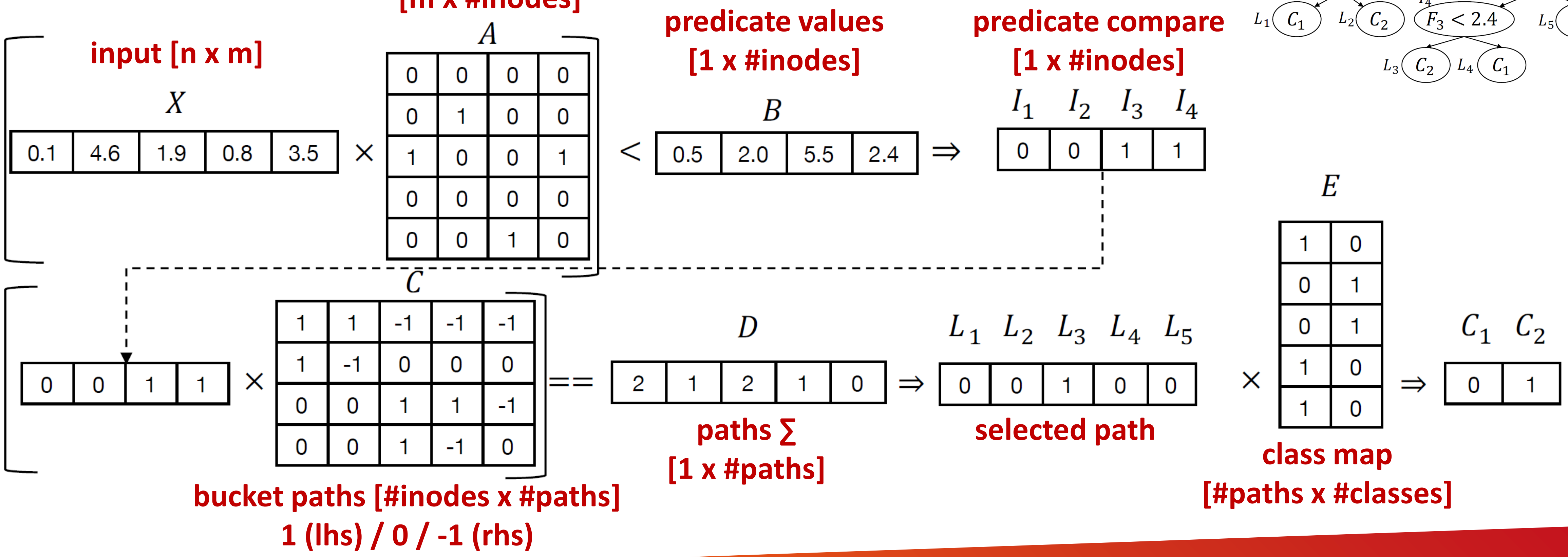

[Vincent Quenneville-Bélair: How PyTorch Optimizes Deep Learning Computations, **Guest Lecture Stanford 2020**]

# Serving Optimizations – Model Vectorization

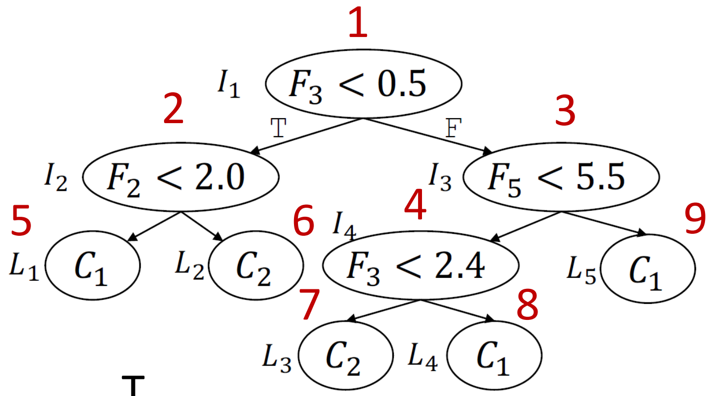
[Supun Nakandala et al: A Tensor Compiler for Unified Machine Learning Prediction Serving. **OSDI 2020,** https://github.com/microsoft/hummingbird]

- **Compile ML scoring pipelines into tensor ops** (3 strategies w/ different redundancy)
- **#1 Matmult (GEMM)**



# Serving Optimizations – Model Vectorization, cont.




- **#2 Tree Traversal (TT)**
  - Traversal for batch of records via value indexing / table() and `ifelse(Tv<Tt, Tl, Tr)`

**Algorithm 2** TreeTraversal Strategy (Notation in Tables 5)

**Input** : $X \in \mathbb{R}^{n \times |F|}$, Input records
**Output** : $R \in \{0,1\}^{n \times |C|}$, Predicted class labels

/* Initialize all records to point to $k$, with $k$ the index of Root node. */
$T_I \leftarrow \{k\}^n$ // $T_I \in \mathbb{Z}^n$
**for** $i \leftarrow 1$ **to** TREE_DEPTH **do**
  /* Find the index of the feature evaluated by the current node. Then find its value. */
  $T_F \leftarrow$Gather$(N_F, T_I)$ // $T_F \in \mathbb{Z}^n$
  $T_V \leftarrow$Gather$(X, T_f)$ $T_F$ // $T_V \in \mathbb{R}^n$
  /* Find the threshold, left child and right child */
  $T_T \leftarrow$Gather$(N_T, T_I)$ // $T_T \in \mathbb{R}^n$
  $T_L \leftarrow$Gather$(N_L, T_I)$ // $T_L \in \mathbb{Z}^n$
  $T_R \leftarrow$Gather$(N_R, T_I)$ // $T_R \in \mathbb{Z}^n$
  /* Perform logical evaluation. If true pick from $T_L$; else from $T_R$. */
  $T_I \leftarrow$Where$(T_V < T_T, T_L, T_R)$ // $I \in \mathbb{Z}^n$
**end**
/* Find label for each leaf node */
$R \leftarrow$Gather$(N_C, T_I)$ // $R \in \mathbb{Z}^n$

Input data

| | | | | |
|---|---|---|---|---|
| F1 | F2 | F3 | F4 | F5 |
| F1 | F2 | F3 | F4 | F5 |
| F1 | F2 | F3 | F4 | F5 |

$T_I$

| |
|---|
| 1 |
| 1 |
| 1 |

Nodes position of individual tuples

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| $N_L$ | 2 | 5 | 4 | 7 | 5 | 6 | 7 | 8 | 9 |
| $N_R$ | 3 | 6 | 9 | 8 | 5 | 6 | 7 | 8 | 9 |
| $N_F$ | 3 | 2 | 5 | 3 | 1 | 1 | 1 | 1 | 1 |
| $N_T$ | 0.5 | 2.0 | 5.5 | 2.4 | 0 | 0 | 0 | 0 | 0 |
| $t(N_C)$ | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 |
| | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 |

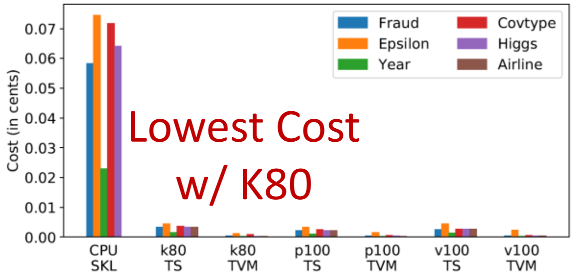
# Serving Optimizations – Model Vectorization, cont.
## Batch Scoring Experiments

Forest Inference Library (FIL)

| Algorithm | Dataset | Baselines (CPU) | | HB CPU | | | Baselines (GPU) | HB GPU | |
|---|---|---|---|---|---|---|---|---|---|
| | | Sklearn | ONNX-ML | PyTorch | TorchScript | TVM | RAPIDS FIL | TorchScript | TVM |
| Rand. Forest | Fraud | **2.5** | 7.1 | 8.0 | 7.8 | 3.0 | not supported | 0.044 | **0.015** |
| | Epsilon | 9.8 | 18.7 | 14.7 | 13.9 | **6.6** | not supported | **0.13** | **0.13** |
| | Year | 1.9 | 6.6 | 7.8 | 7.7 | **1.4** | not supported | 0.045 | **0.026** |
| | Covtype | **5.9** | 18.1 | 17.22 | 16.5 | 6.8 | not supported | 0.11 | **0.047** |
| | Higgs | **102.4** | 257.6 | 314.4 | 314.5 | 118.0 | not supported | 1.84 | **0.55** |
| | Airline | 1320.1 | timeout | timeout | timeout | **1216.7** | not supported | 18.83 | **5.23** |
| LightGBM | Fraud | 3.4 | 5.9 | 7.9 | 7.6 | **1.7** | **0.014** | 0.044 | **0.014** |
| | Epsilon | 10.5 | 18.9 | 14.9 | 14.5 | **4.0** | 0.15 | 0.13 | **0.12** |
| | Year | 5.0 | 7.4 | 7.7 | 7.6 | **1.6** | **0.023** | 0.045 | 0.025 |
| | Covtype | 51.06 | 126.6 | 79.5 | 79.5 | **27.2** | not supported | 0.62 | **0.25** |
| | Higgs | 198.2 | 271.2 | 304.0 | 292.2 | **69.3** | 0.59 | 1.72 | **0.52** |
| | Airline | 1696.0 | timeout | timeout | timeout | **702.4** | 5.55 | 17.65 | **4.83** |
| XGBoost | Fraud | 1.9 | 5.5 | 7.7 | 7.6 | **1.6** | **0.013** | 0.44 | 0.015 |
| | Epsilon | 7.6 | 18.9 | 14.8 | 14.8 | **4.2** | 0.15 | 0.13 | **0.12** |
| | Year | 3.1 | 8.6 | 7.6 | 7.6 | **1.6** | **0.022** | 0.045 | 0.026 |
| | Covtype | 42.3 | 121.7 | 79.2 | 79.0 | **26.4** | not supported | 0.62 | **0.25** |
| | Higgs | 126.4 | 309.7 | 301.0 | 301.7 | **66.0** | 0.59 | 1.73 | **0.53** |
| | Airline | 1316.0 | timeout | timeout | timeout | **663.3** | 5.43 | 17.16 | **4.83** |

Azure NC6 v2
(6 vcores, 112GB, P1 GPU)

Batch of 10K records
[seconds]



# Serving Optimizations – Model Distillation

- **Model Distillation**
  - Ensembles of models → **single NN model**
  - Specialized models for different classes (found via differences to generalist model)
  - Trained on soft targets (softmax w/ temperature T)

[Geoffrey E. Hinton, Oriol Vinyals, Jeffrey Dean: Distilling the Knowledge in a Neural Network. **CoRR 2015**]

$$q_i = \frac{exp(z_i/T)}{\sum_j exp(z_j/T)}$$

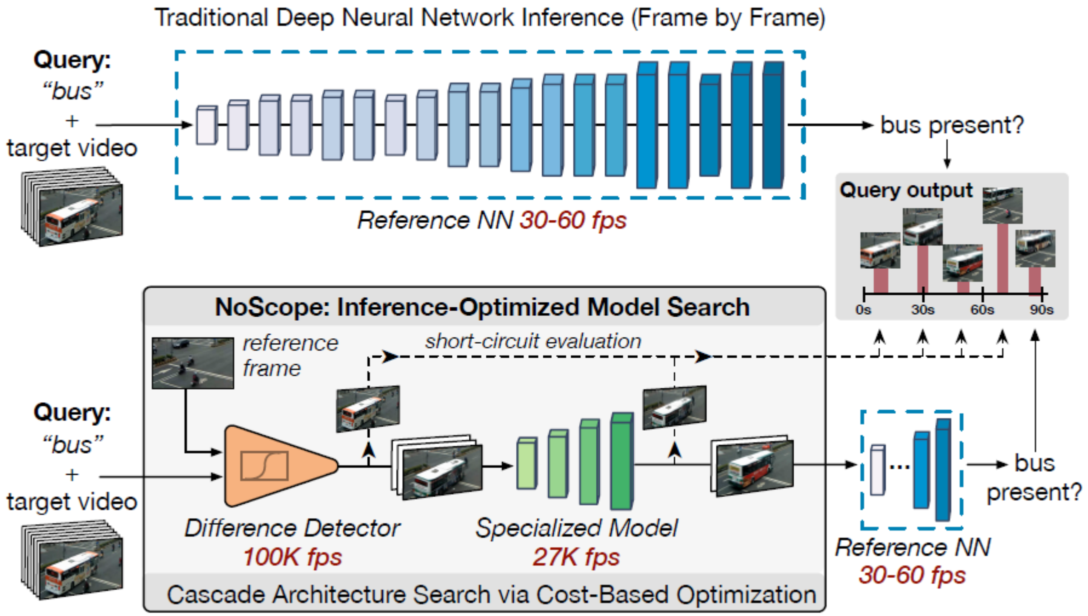
- **Example Experiments**
  - Automatic Speech Recognition
  - Frame classification accuracy, and word error rate

| System | Test Frame Accuracy | Word Error Rate |
|---|---|---|
| **Baseline** | 58.9% | 10.9% |
| **10x Ensemble** | 61.1% | 10.7% |
| **Distilled 1x Model** | 60.8% | 10.7% |

# Serving Optimizations – Specialization

- **NoScope Architecture**
  - Baseline: YOLOv2 on 1 GPU per video camera @30fps
  - **Optimizer to find filters**

[Daniel Kang et al: NoScope: Optimizing Deep CNN-Based Queries over Video Streams at Scale. **PVLDB 2017**]




- **#1 Model Specialization**
  - Given query and baseline model
  - Trained shallow NN (based on AlexNet) on output of baseline model
  - Short-circuit if prediction with high confidence

- **#2 Difference Detection**
  - Compute difference to ref-image/earlier-frame
  - Short-circuit w/ ref label if no significant difference

# Model Monitoring and Updates

**Part of Model Management and MLOps**
**(see 10 Model Selection & Management)**

# Model Deployment Workflow

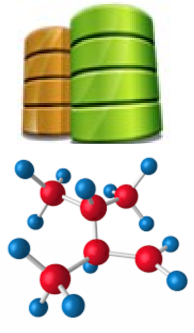
**Data Integration**
**Data Cleaning**
**Data Preparation**

**Model Selection**
**Training**
**Hyper-parameters**

**#1 Model Deployment**

**MX** **MY** **B**

Prediction Requests



**#2 Continuous Data Validation / Concept Drift Detection**

**Model Serving**

**#3 Model Monitoring**

**#4 Periodic / Event-based Re-Training & Updates**
(automatic / semi-manual)

**DevOps Engineer**

# Monitoring Deployed Models


Google
Data Management Challenges in Production Machine Learning
Neoklis Polyzotis, Sudip Roy, Steven Whang, Martin Zinkevich

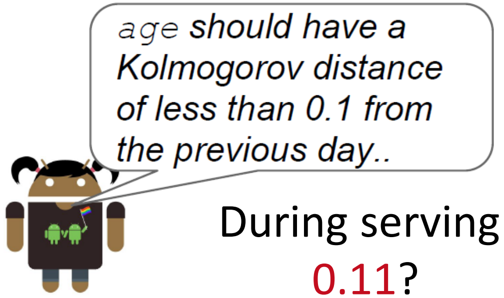
[Neoklis Polyzotis, Sudip Roy, Steven Whang, Martin Zinkevich: Data Management Challenges in Production Machine Learning, **SIGMOD 2017**]

- **Goals:** Robustness (e.g., data, latency) and **model accuracy**

- **#1 Check Deviations Training/Serving Data**
  - Different data distributions, distinct items → impact on model accuracy?
  - → See **09 Data Acquisition and Preparation** (Data Validation)

- **#2 Definition of Alerts**
  - Understandable and actionable
  - Sensitivity for alerts (**ignored if too frequent**)

- **#3 Data Fixes**
  - Identify problematic parts
  - Impact of fix on accuracy
  - How to backfill into training data




During serving: 0.11?

**"The question is not whether something is 'wrong'. The question is whether it gets fixed"**

# Monitoring Deployed Models, cont.


[Neoklis Polyzotis, Sudip Roy, Steven Whang, Martin Zinkevich: Data Management Challenges in Production Machine Learning, **SIGMOD 2017**]

- **Alert Guidelines**
  - **Make them actionable**
    missing field,
    field has new values,
    distribution changes

    less actionable
  - **Question** data AND constraints
  - Combining repairs:
    **principle of minimality**

[George Beskales et al: On the relative trust between inconsistent data and inaccurate constraints. **ICDE 2013**]

[Xu Chu, Ihab F. Ilyas: Qualitative Data Cleaning. Tutorial, **PVLDB 2016**]

- **Complex Data Lifecycle**
  - Adding new features to production ML pipelines is a **complex process**
  - Data does not live in a DBMS; data often resides in **multiple storage systems** that have **different characteristics**
  - Collecting data for training can be **hard and expensive**

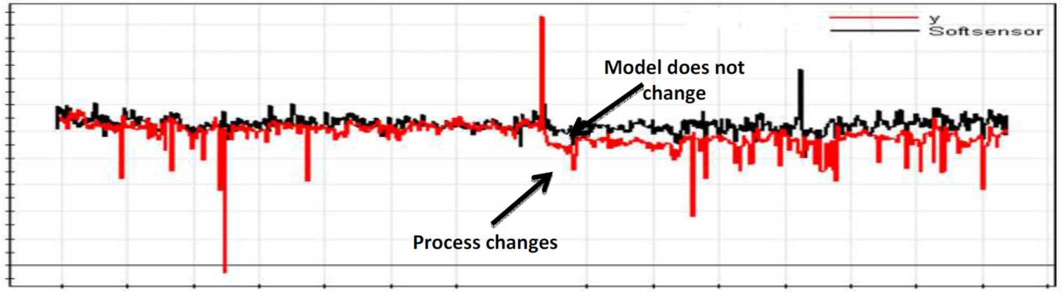
# Concept Drift

[A. Bifet, J. Gama, M. Pechenizkiy, I. Žliobaitė: Handling Concept Drift: Importance, Challenges & Solutions, **PAKDD 2011**]

- **Recap Concept Drift** (features → labels)
  - **Change of statistical properties** / dependencies (features-labels)
  - Requires re-training, parametric approaches for deciding when to retrain

- **#1 Input Data Changes**
  - Population change (gradual/sudden), but also new categories, data errors
  - **Covariance shift** p(x) with constant p(y|x)

- **#2 Output Data Changes**
  - **Label shift** p(y)
  - Constant conditional feature distributed p(x|y)



*source: Evonik Industries*

- **Goals:** Fast adaptation; noise vs change, recurring contexts, small overhead

# Concept Drift, cont.

[A. Bifet, J. Gama, M. Pechenizkiy, I. Žliobaitė: Handling Concept Drift: Importance, Challenges & Solutions, **PAKDD 2011**]


Handling Concept Drift:
Importance, Challenges & Solutions

- Approach 1: **Periodic Re-Training**
  - Training: **window of latest data** + data selection/weighting
  - Alternatives: incremental maintenance, warm starting, online learning
- Approach 2: **Event-based Re-Training**
  - **Change detection** (supervised, unsupervised)
  - Often model-dependent, specific techniques for time series
  - **Drift Detection Method:** binomial distribution, if error outside scaled standard-deviation → raise warnings and alerts
  - **Adaptive Windowing (ADWIN):** window W, append data to W, drop old values until avg windows W=W1-W2 similar (below epsilon), raise alerts
  - **Kolmogorov-Smirnov distance / Chi-Squared:** univariate statistical tests training/serving

[Albert Bifet, Ricard Gavaldà: Learning from Time-Changing Data with Adaptive Windowing. **SDM 2007**]

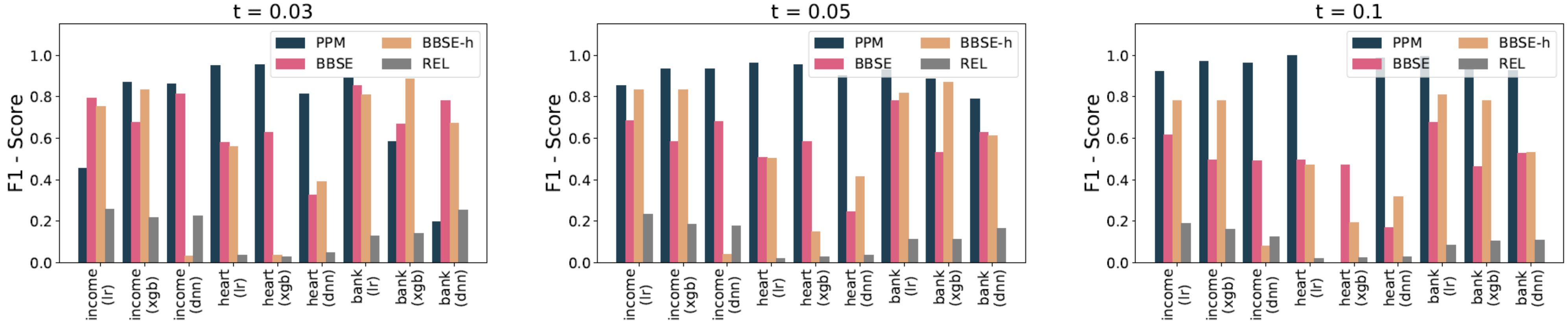
[https://scikitmultiflow.readthedocs.io/en/stable/api/generated/skmultiflow.drift_detection.ADWIN.html]

# Concept Drift, cont.

[Sebastian Schelter, Tammo Rukat, Felix Bießmann: Learning to Validate the Predictions of Black Box Classifiers on Unseen Data. **SIGMOD 2020**]

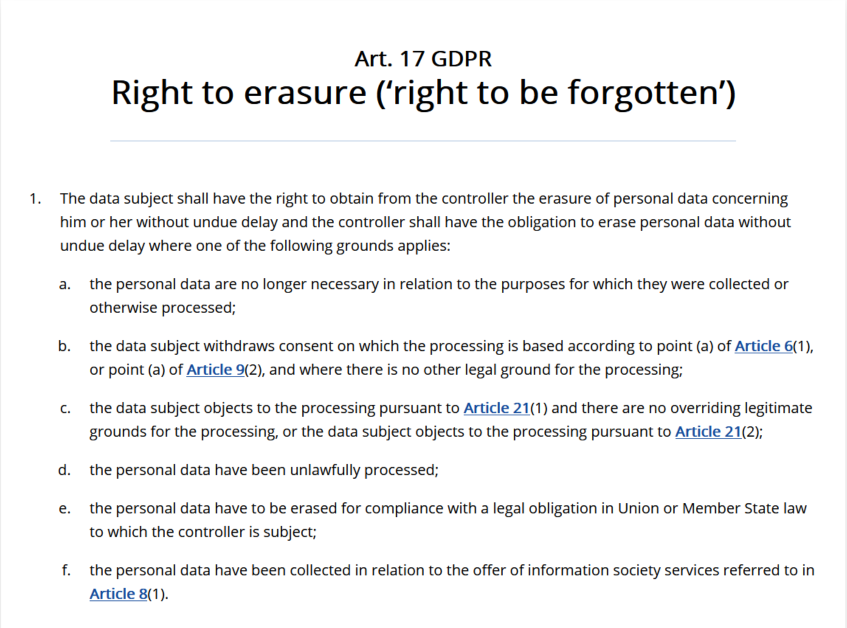
- **Model-agnostic Performance Predictor**
  - Approach 2: Event-based Re-Training
  - User-defined error generators
  - Synthetic data corruption → impact on black-box model
  - Train performance predictor (regression/classification at threshold t) for expected prediction quality on **percentiles of target variable ŷ**

- **Results PPM**



# GDPR (General Data Protection Regulation)

[https://gdpr.eu/article-17-right-to-be-forgotten/]



Art. 17 GDPR

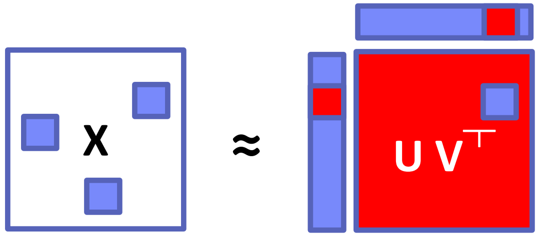
Right to erasure ('right to be forgotten')

1. The data subject shall have the right to obtain from the controller the erasure of personal data concerning him or her without undue delay and the controller shall have the obligation to erase personal data without undue delay where one of the following grounds applies:
   a. the personal data are no longer necessary in relation to the purposes for which they were collected or otherwise processed;
   b. the data subject withdraws consent on which the processing is based according to point (a) of Article 6(1), or point (a) of Article 9(2), and where there is no other legal ground for the processing;
   c. the data subject objects to the processing pursuant to Article 21(1) and there are no overriding legitimate grounds for the processing, or the data subject objects to the processing pursuant to Article 21(2);
   d. the personal data have been unlawfully processed;
   e. the personal data have to be erased for compliance with a legal obligation in Union or Member State law to which the controller is subject;
   f. the personal data have been collected in relation to the offer of information society services referred to in Article 8(1).

- **GDPR "Right to be Forgotten"**
  - Recent laws such as GDPR require companies and institutions to **delete user data upon request**
  - Personal data must not only be deleted from primary data stores but also from **ML models** trained on it (Recital 75)

- **Example Deanonymization**
  - Recommender systems: models **retain user similarly**
  - Social network data / clustering / KNN
  - Large language models (e.g., GPT-3)

[Sebastian Schelter: "Amnesia" - Machine Learning Models That Can Forget User Data Very Fast. **CIDR 2020**]

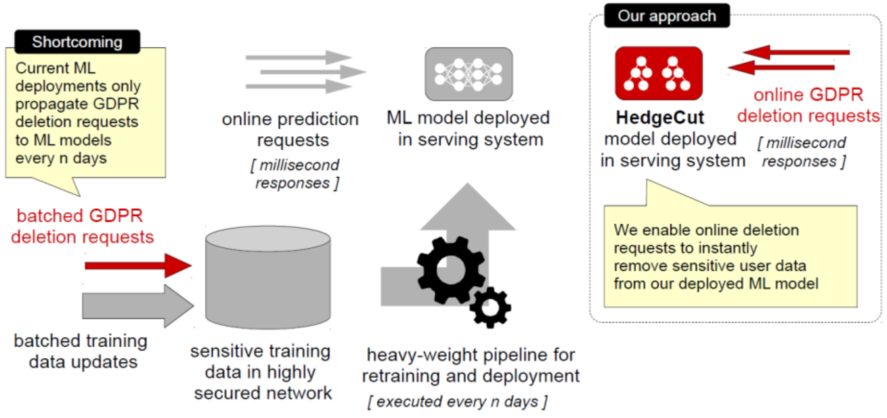



See incremental computations in **03 Sizes Inferences and Rewrites**

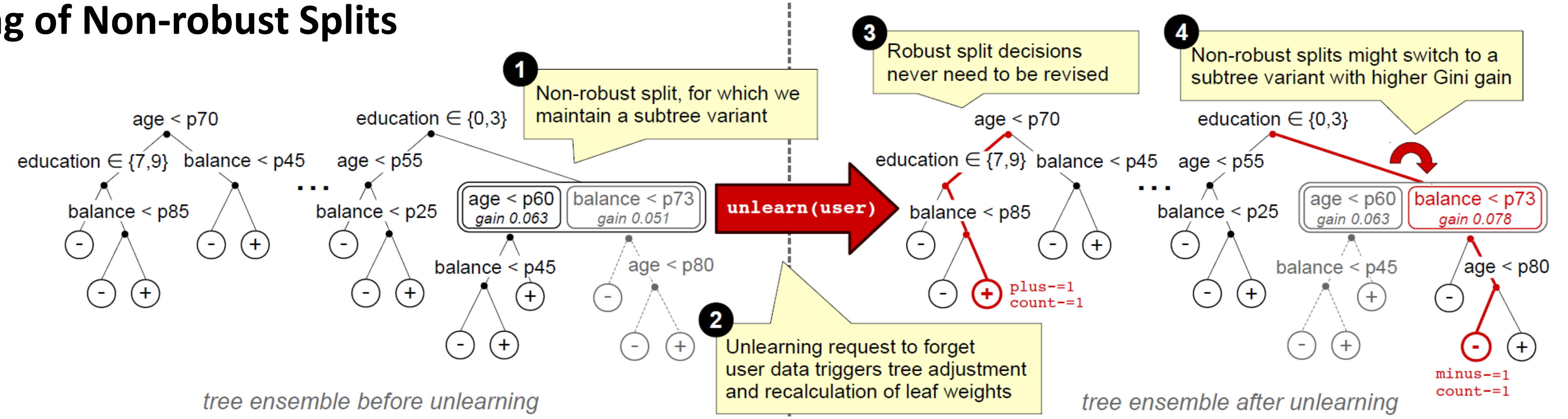
# GDPR (General Data Protection Regulation), cont.

[Sebastian Schelter, Stefan Grafberger, Ted Dunning: HedgeCut: Maintaining Randomised Trees for Low-Latency **Machine Unlearning**, **SIGMOD 2021**]

- **HedgeCut Overview**
  - Extremely Randomized Trees (ERT): ensemble of DTs w/ randomized attributes and cut-off points
  - **Online unlearning requests** < 1ms w/o retraining for few points




- **Handling of Non-robust Splits**



# Summary & QA

- **Model Exchange and Serving**
- **Model Monitoring and Updates**

# Thanks

- **#1 Exam Preparation – Ask Questions in the Forum**
- **#2 Written Exams**
  - Register for an exam slot **in MOSES**
  - Thu, **Jul 18, 4.15pm** in H0107 (**24**/144 seats) → **21** registration(s)
  - Sa, **Aug 10, 2.15pm** in A053 and EB 301 (**106**/639 seats) → **11** registration(s)
  - Sa, **Oct 12, 13.15pm**