Fast Machine Learning

+

Takuto KUNIGO

October 24, 2025

Belle II Trigger/DAQ Workshop 2025

KEK

Machine learning in HEP experiments

Could you explain typical applications of machine learning in particle physics experiments in 500 words?

Machine learning (ML) has become an indispensable tool in particle physics, where experiments generate massive, complex datasets that challenge traditional analysis methods. From event reconstruction to anomaly detection, ML enhances nearly every stage of data processing and interpretation in modern high-energy physics (HEP) experiments such as those at CERN's Large Hadron Collider (LHC).

1. Event Reconstruction and Particle Identification
At the heart of collider experiments lies the challenge of reconstructing particle interactions from raw detector signals. Detectors record vast amounts of data from particle collisions—signals from calorimeters, tracking chambers, and muon systems. ML algorithms, particularly deep neural networks (DNNs) and convolutional neural networks (CNNs), are used to map these signals to physical quantities such as particle momenta and energy deposits. For example, ML-based calorimeter reconstruction models outperform classical algorithms by capturing nonlinear detector responses and correlations between channels. Similarly, boosted decision trees (BDTs) and graph neural networks (GNNs) are widely used for track reconstruction—identifying trajectories of charged particles through detector layers—even in the presence of noise and overlapping events ("pile-up").

2. Event Classification and Signal Discrimination
A key goal of particle physics is to identify rare processes (signals) among enormous backgrounds from known physics. Traditional cut-based analyses are often limited by their inability to capture complex, multidimensional correlations between observables. ML classifiers, such as BDTs and DNNs, are now standard for separating signal from background. For instance, the discovery of the Higgs boson involved multivariate classifiers to distinguish Higgs decay signatures from background events. More recent work uses architectures like attention networks and GNNs to represent entire collision events as structured data, enabling more powerful and interpretable discrimination.

3. Anomaly Detection and New Physics Searches
ML also supports *model-independent* searches for new physics. Instead of targeting specific theoretical models, unsupervised and semi-supervised learning methods—autoencoders, variational autoencoders (VAEs), and normalizing flows—are trained on background data to learn "normal" event distributions. Events that deviate significantly from this learned distribution may signal novel phenomena. This approach allows physicists to explore unexpected signatures without bias toward preconceived models.

4. Fast Simulation and Data Generation
Monte Carlo simulations are essential for modeling particle interactions, but they are computationally expensive. Generative ML models such as generative adversarial networks (GANs) and diffusion models are increasingly used to produce high-fidelity simulated data much faster than traditional methods. These models can generate realistic detector responses, accelerating physics analyses and enabling rapid prototyping of detector designs.

5. Trigger Systems and Real-Time Decision Making
Collider experiments record billions of collisions per second, but only a tiny fraction can be stored for analysis. ML algorithms are now being embedded in real-time trigger systems to decide which events to record. Specialized, low-latency ML models—often implemented on FPGAs or GPUs—allow more sophisticated selection criteria, improving the efficiency of capturing rare events while keeping data rates manageable.

6. Calibration, Alignment, and Uncertainty Estimation
ML aids in optimizing detector calibration and alignment, modeling systematic uncertainties, and improving measurement precision. Bayesian ML methods and probabilistic models quantify uncertainties and correlations more accurately than deterministic fits.

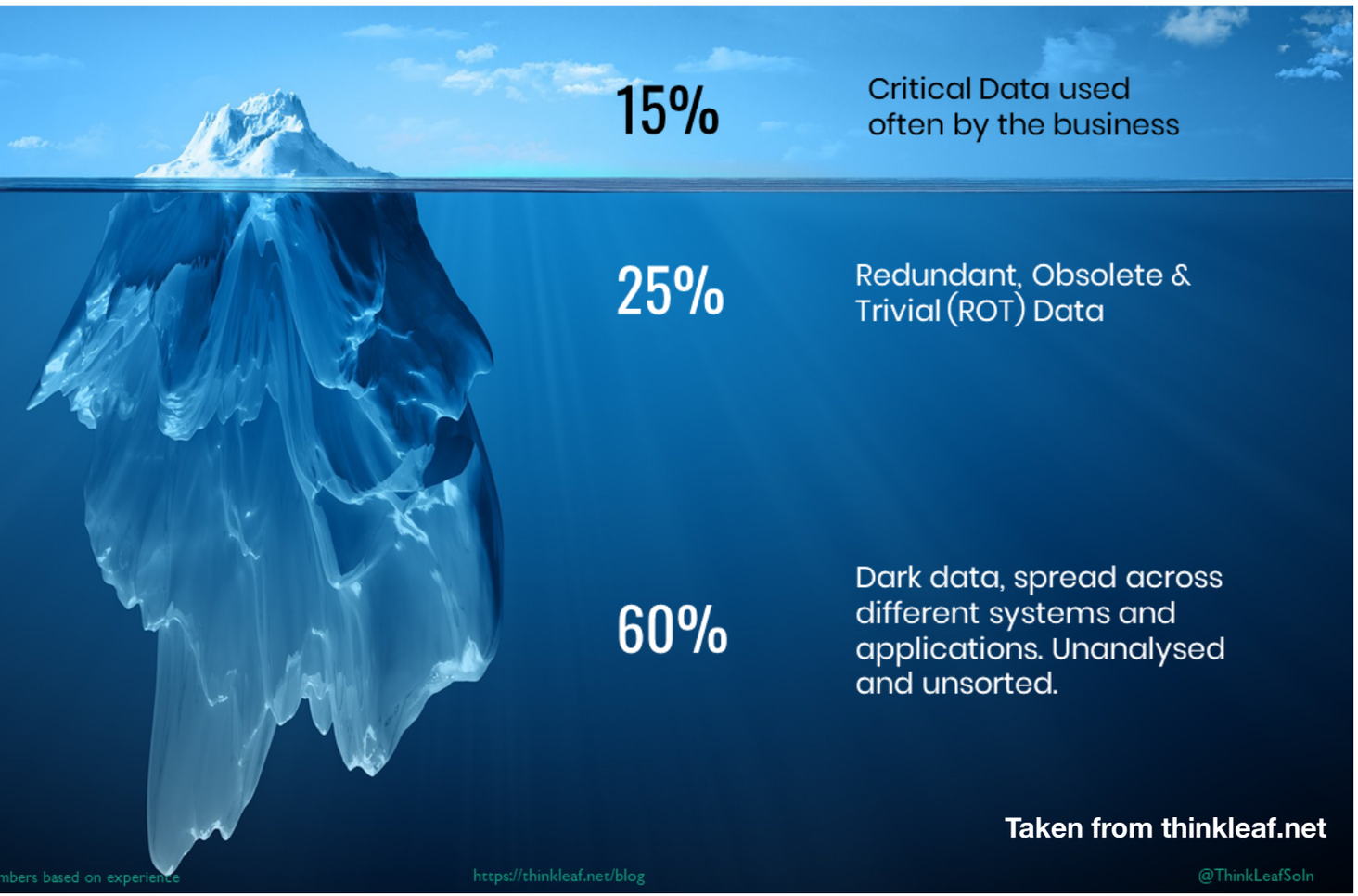
In summary, machine learning has transformed particle physics by enhancing data reconstruction, improving sensitivity to rare signals, accelerating simulations, and enabling new types of discovery. As experiments continue to grow in scale and complexity, ML will remain central to both operational efficiency and scientific insight in high-energy physics.

- ML approaches in the trigger techniques recognised by ChatGPT.
- (Maybe, it is better to add ML-based DQM in the list)
- **No "Fast" or "Realtime" Machine learning item** in the list, unfortunately.

Motivation

Fundamental question

1. How much “dark data” we have?
2. If we use it effectively, we may predict near future errors?

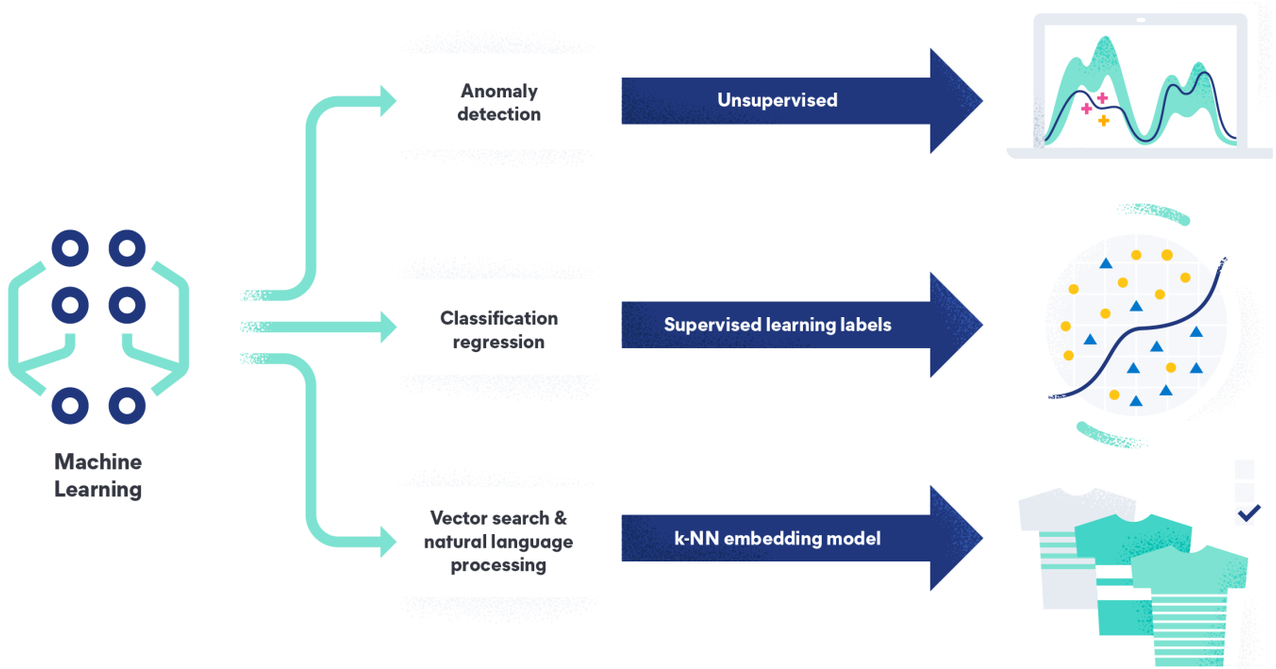


Dark data is data which is acquired through various computer network operations but not used in any manner to derive insights or for decision making. (by Wikipedia)

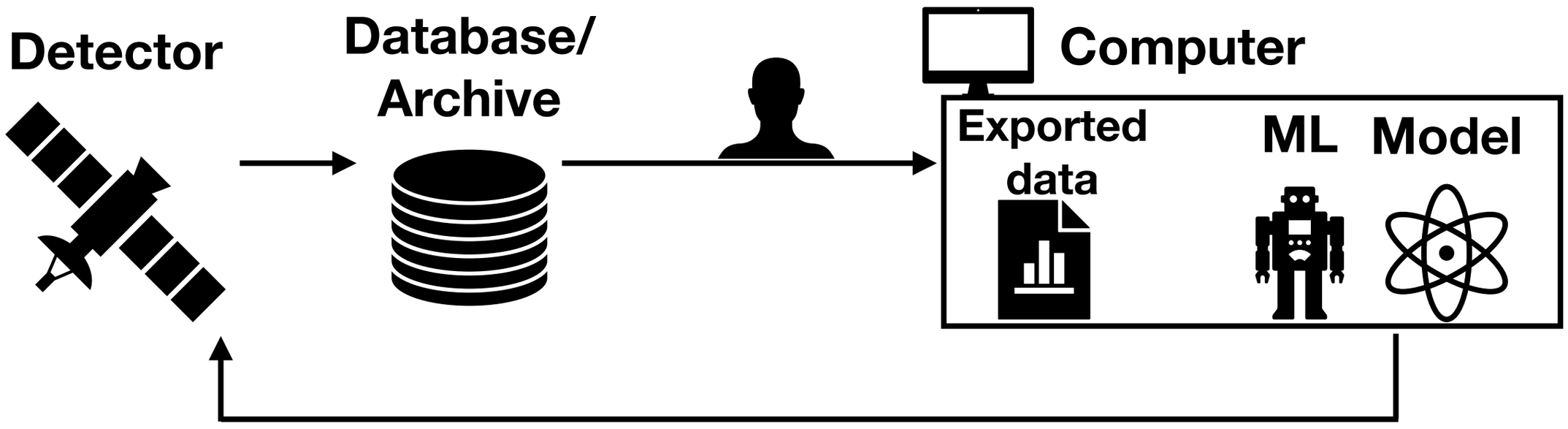
FastML: Introduction

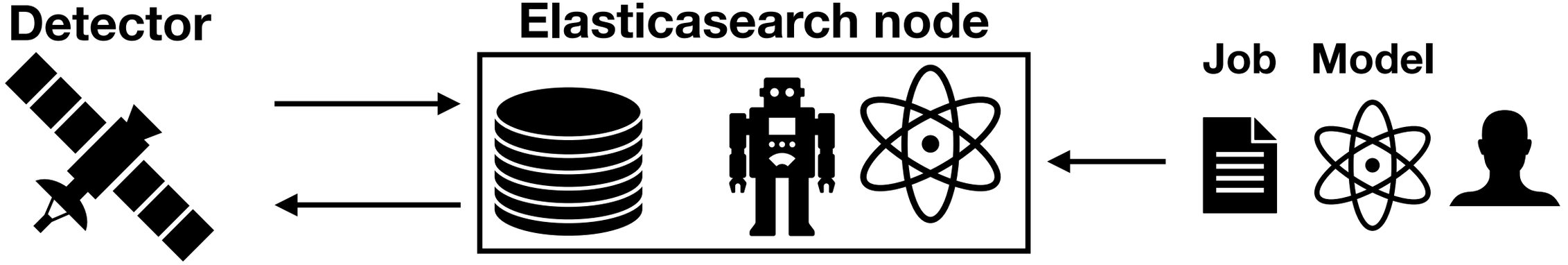
Motivation

- ML approaches used in HEP community for a long time.
- However, use cases are limited; reconstruction, tracking, PID, trigger, event categorisation, etc.
- Is it effective to feed our operational data into ML to train a model in realtime?
  - Fast/Realtime ML is widely used in the world (e.g. network security)
  - We (probably anyone) don't know if it is effective in HEP operation
- To use the ML feature, we need a paid license. (or a trial license)
- I'm evaluating the ML performance requesting a trial license time to time. I'm contacting the company to activate it in November/December.
- ML modelling can be done outside the Elasticsearch cluster using a Python package.



Benefit of FastML





- No need 1) to prepare a computer, 2) to export data, 3) to worry about interrupting the other processes due to limited I/O, network bandwidth
- We can concentrate on what we really would like to achieve using ML
  - Even without expertise, it is possible to define a new ML job (For example, Anomaly detection using Instantaneous luminosity + Trigger rates + Run Control state)
  - With expertise, you can train a model by yourself and upload it to Elasticsearch node

Ideas of FastML

- **Categorisation of the run-stop reasons (DAQ)**
  - **Description:** CR shifters describe the run-stop reason in e-log entries; sometimes the reason is not correct or sometimes CR shifters forget writing it
  - **Input:** RCPanel + HVPanel + TTDInfo + BeamInfo
  - **ML:** Clustering (Image similarity)
  - **Goal:** Train a model to automatically diagnose the run-stop reasons and perform an appropriate recovery action (The trained model is automatically updated)

- **Categorising the reasons of beam aborts (MDI? whole Belle II)**
  - **Description:** Sudden Beam Loss (SBL) is the biggest issue in Belle II/SuperKEKB There are many studies on-going incl. ML-based ones. If we implement them into Elasticsearch, training/analyses become automatic, and automatic action can be implemented.
  - **Input:** As many variables/images as possible.
  - **ML:** Clustering
  - **Goal:** Understand the beam condition; this knowledge should be used to stabilise the beams

(continued)

- **Anomaly detection in (L1+HLT) trigger rates(TRG+DAQ)**
  - **Description:** Trigger rates can be a good barometer to understand the detector configuration. If rates of a specific trigger is too low or too high, it can be a key of mis-configuration.
  - **Input:** Luminosity, BG related parameters, detector config, various trigger rates
  - **ML:** (Isolation forest? Regression?), sorry, I'm not sure
  - **Goal:** Automatically detect a mis-configuration.

Our Elasticsearch cluster has enough data for machine learning
We can try many ML applications in Elasticsearch

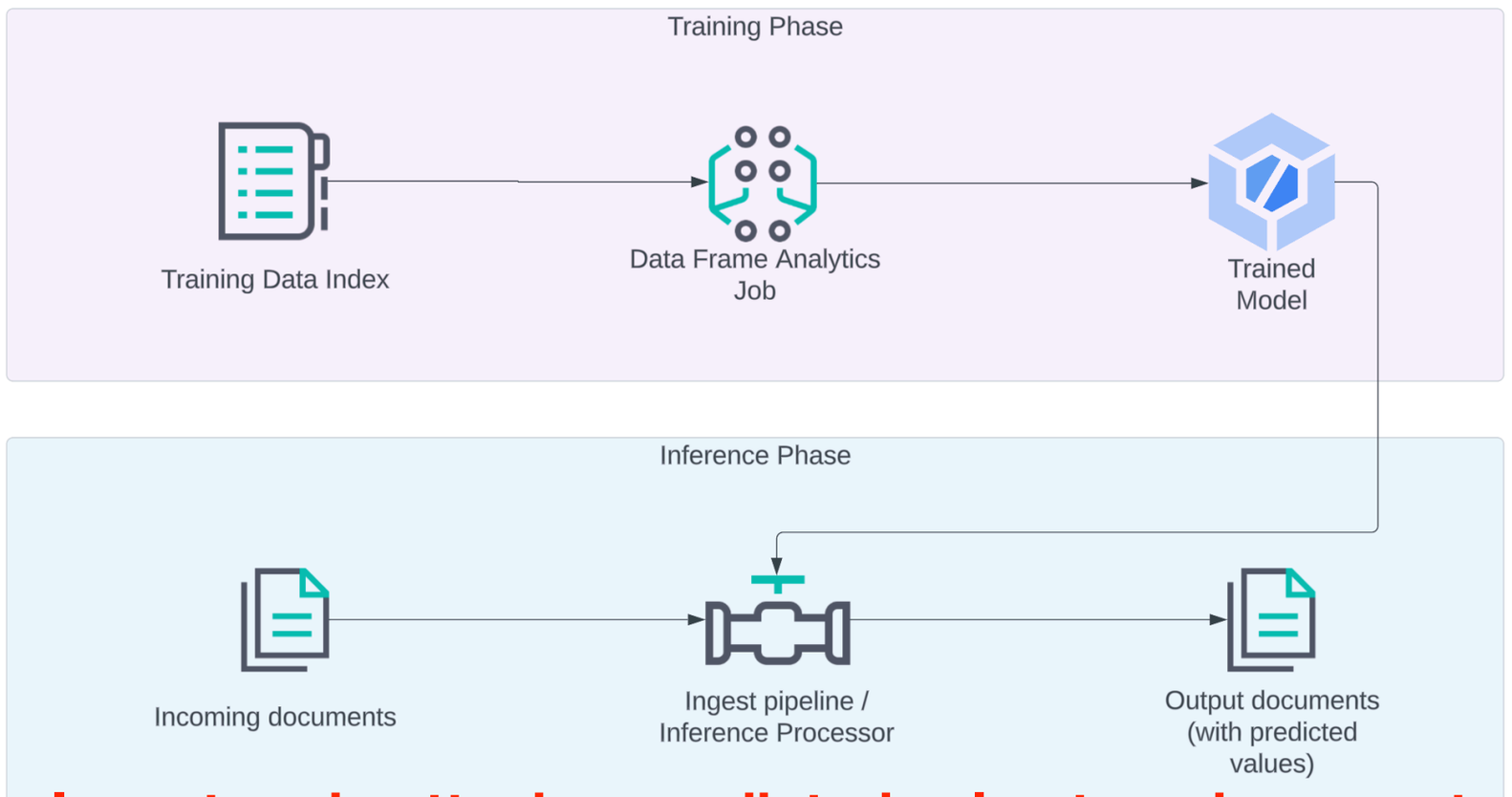
- **Note**
  - (Not all but) some models from scikit-learn, XGBoost, and LightGBM can be imported using a Python module
  - It is rather difficult to (for me) to evaluate its performance without a license
  - I cannot guarantee that I can get a license always you want it

ML: Training and Ingest pipeline

Elasticsearch provides machine learning algorithms;

1. Anomaly detection
2. Data frame analysis
   A. Outlier detection
   B. Regression
   C. Classification
3. Natural language processing: k-NN embedding model
4. Image similarity search

Training in a ML node



Ingest node attach a predicted value to a document

Correlation: Luminosity–Trigger rate

Delivered luminosity - Recorded luminosity - Input L1 rate

Too large discrepancy need to be ignored

May 22nd 2024, 14:00	< 1	mean("DAQ.TTDS.TrigIn.Rate") partitionfield="DAQ.RC.RunType.keyword"	physics	DAQ.RC.RunType.keyword: physics	3,910.467	4,421.48	1.1x lower	
May 21st 2024, 12:00	< 1	mean("DAQ.TTDS.TrigIn.Rate") partitionfield="DAQ.RC.RunType.keyword"	physics	DAQ.RC.RunType.keyword: physics	267.341	4,158.555	16x lower	

Log-messages Trained model by me

Elasticsearch can handles non-numeric format by converting them to embeddings.

K-means Clustering by scikit-learn

logcollectord Edit job selection

Filter by influencer fields... (mlcategory : 2)

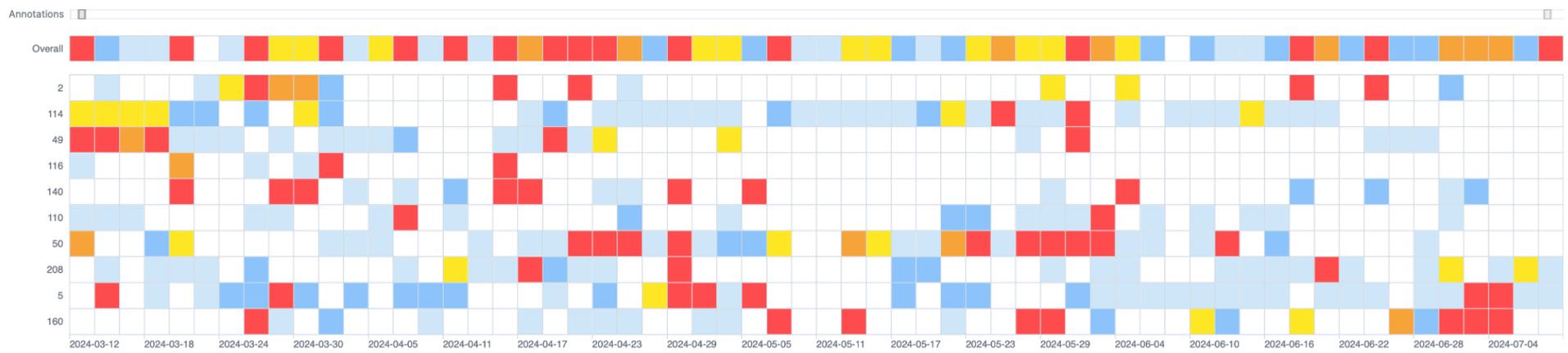
Top influencers

mlcategory		
mlcategory 2	99	18k
mlcategory 114	99	1530
mlcategory 49	99	1141
mlcategory 116	99	274
mlcategory 140	99	1615
mlcategory 110	99	225
mlcategory 50	99	1994
mlcategory 208	99	682
mlcategory 5	99	25k
mlcategory 160	99	26k

Anomaly timeline

View by mlcategory Seve... 0 0 25 50 75 100

(Sorted by max anomaly score)



Rows per page: 10 1 2 3 4 5 ... 33

Always “anomaly”

Outlier detection

We can find some unusual messages

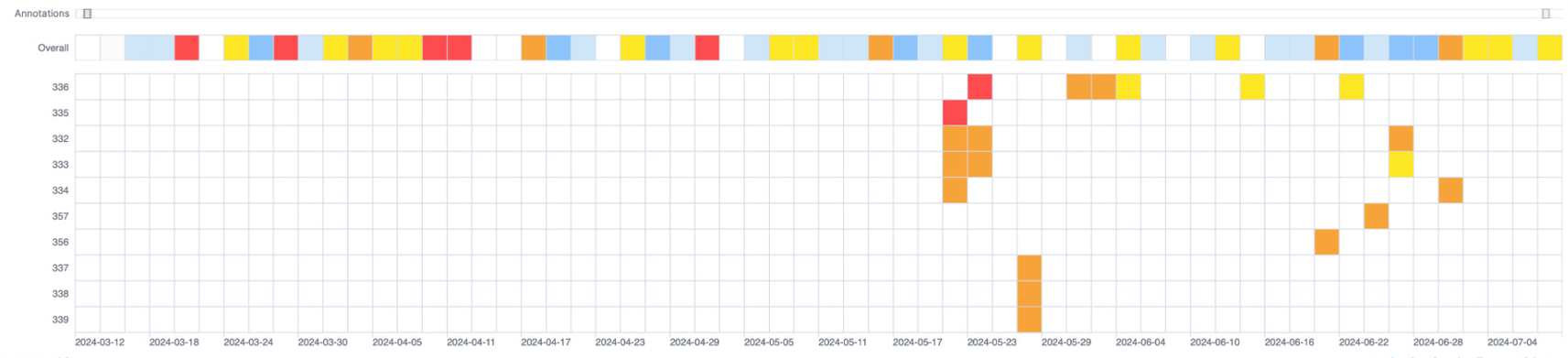
Top influencers

mlcategory		
mlcategory 336	76	491
mlcategory 335	75	75
mlcategory 332	74	232
mlcategory 333	74	230
mlcategory 334	74	137
mlcategory 357	73	73
mlcategory 356	73	73
mlcategory 337	73	290
mlcategory 338	73	131
mlcategory 339	73	73

Anomaly timeline

View by mlcategory Seve... 0 0 25 50 75 100

(Sorted by max anomaly score)



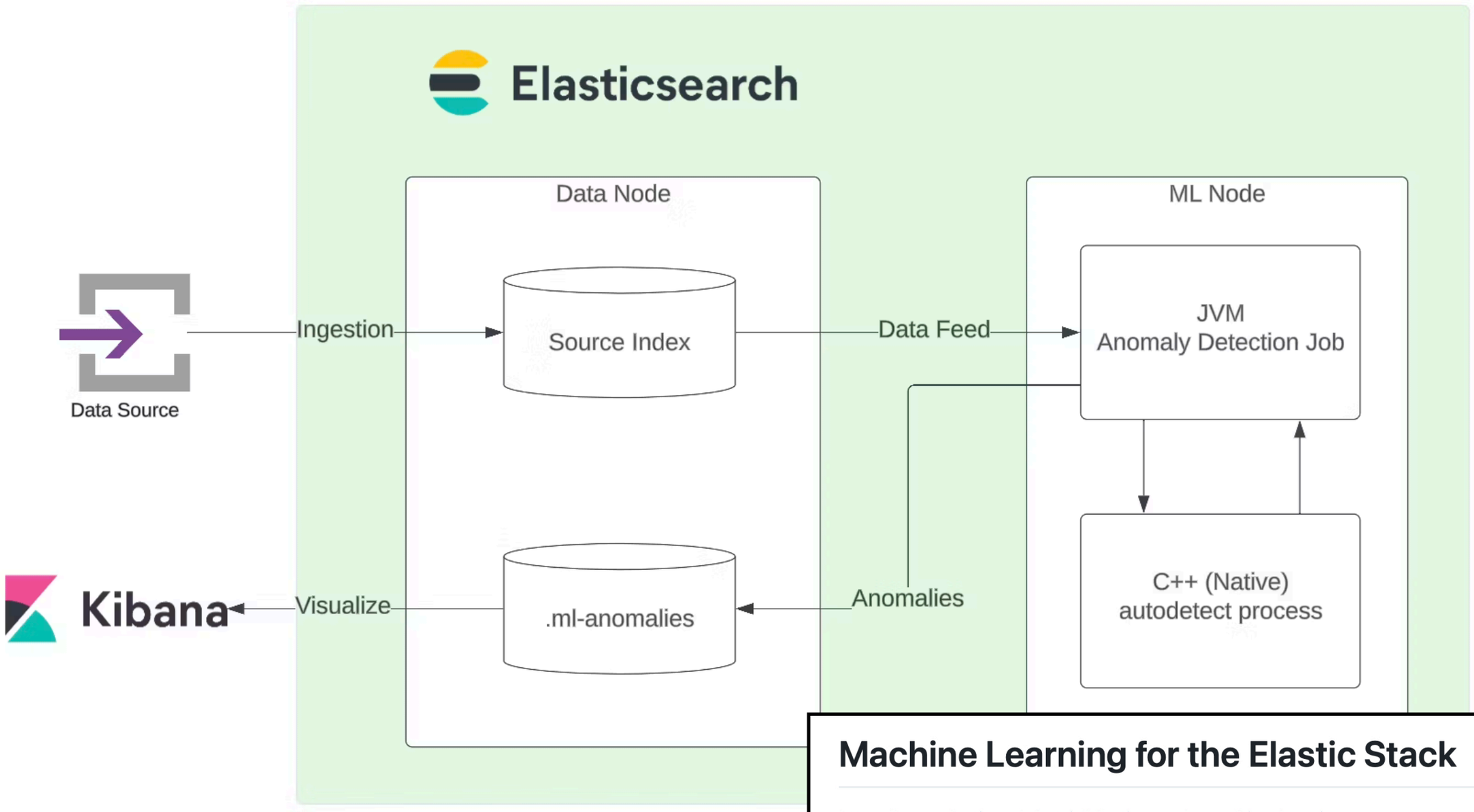
Summary

- FastML maybe effective for HEP experiments (but may not be)
- Software license to utilise the ML feature is expensive, thus little people performed a realistic survey
- (As far as I know) Belle II is the best experiment for the first FastML trial on operational data
- Some early studies on-going now
- I need some team members;
  - FastML is much more efficient than a hand-maid ML app
  - Many possible applications; DAQ, Trigger, MDI, sub-detectors, etc.
  - however, it is difficult to welcome many “guest” users
  - also, I cannot guarantee that ML license is always available nor we can reach the goal
  - Team members who can make the story real(istic) are more than welcome!

Backup

ML: Anomaly detection

In a ML node, anomaly detection job controls a sub-job, in which core analysis of ML is performed.
Source codes are available: but it will take long time to understand them…
(It's a commercial-level product)



Machine Learning for the Elastic Stack

https://www.elastic.co/what-is/elasticsearch-machine-learning

ML: Home made

Eland, Elasticsearch Python client and toolkit, provides a pandas-combative dataframe

```
#!/usr/bin/env python3
import eland as ed

df = ed.DataFrame(
    es_client="http://daqslc-elk.daqnet.kek.jp:9200",
    es_index_pattern="b2daq-efficiency*",
)
```

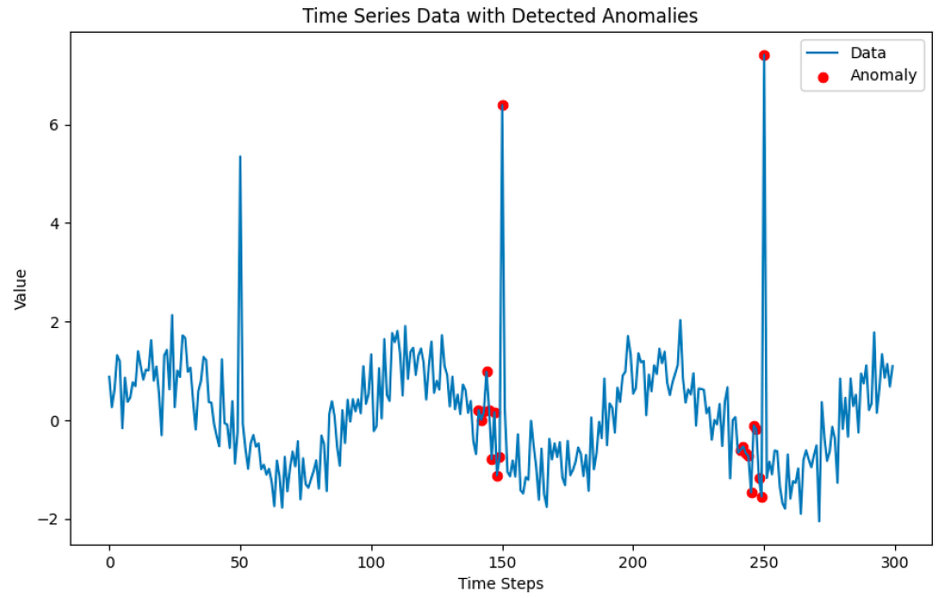
We can transform trained models from scikit-learn, XGBoost, and LightGBM libraries to be serialised and import to Elasticsearch (Regression, classification)

```
from xgboost import XGBClassifier
from eland.ml import MLModel

# Train and exercise an XGBoost ML model locally
xgb_model = XGBClassifier(booster="gbtree")
xgb_model.fit(training_data[0], training_data[1])

# Import the model into Elasticsearch
es_model = MLModel.import_model(
    es_client="http://daqslc-elk.daqnet.kek.jp:9200",
    model_id="xgb-classifier",
    model=xgb_model,
    feature_names=["f0", "f1", "f2", "f3", "f4"],
)

# Exercise the ML model in Elasticsearch with the training data
es_model.predict(training_data[0])
[0 1 1 0 1 0 0 0 1 0]
```

ML: Home made anomaly detection

Anomaly detection

- I'm trying to mimic the anomaly calculation implemented in Elasticsearch
- Whatever algorithms I should try; even if it is not supported by Elasticsearch
- I need to (re)learn the techniques of anomaly detection (especially multivariate anomaly detection on time-series data)



Huggingface hug models:

- Eland can import PyTorch models
- can import from huggingface.co (local import is also possible)

```
git clone https://huggingface.co/dslim/bert-base-NER
eland_import_hub_model \
      --url 'http://daqslc-elk.daqnet.kek.jp:9200' \
      --hub-model-id /PATH/TO/MODEL \
      --task-type ner \
      --es-model-id bert-base-ner
```