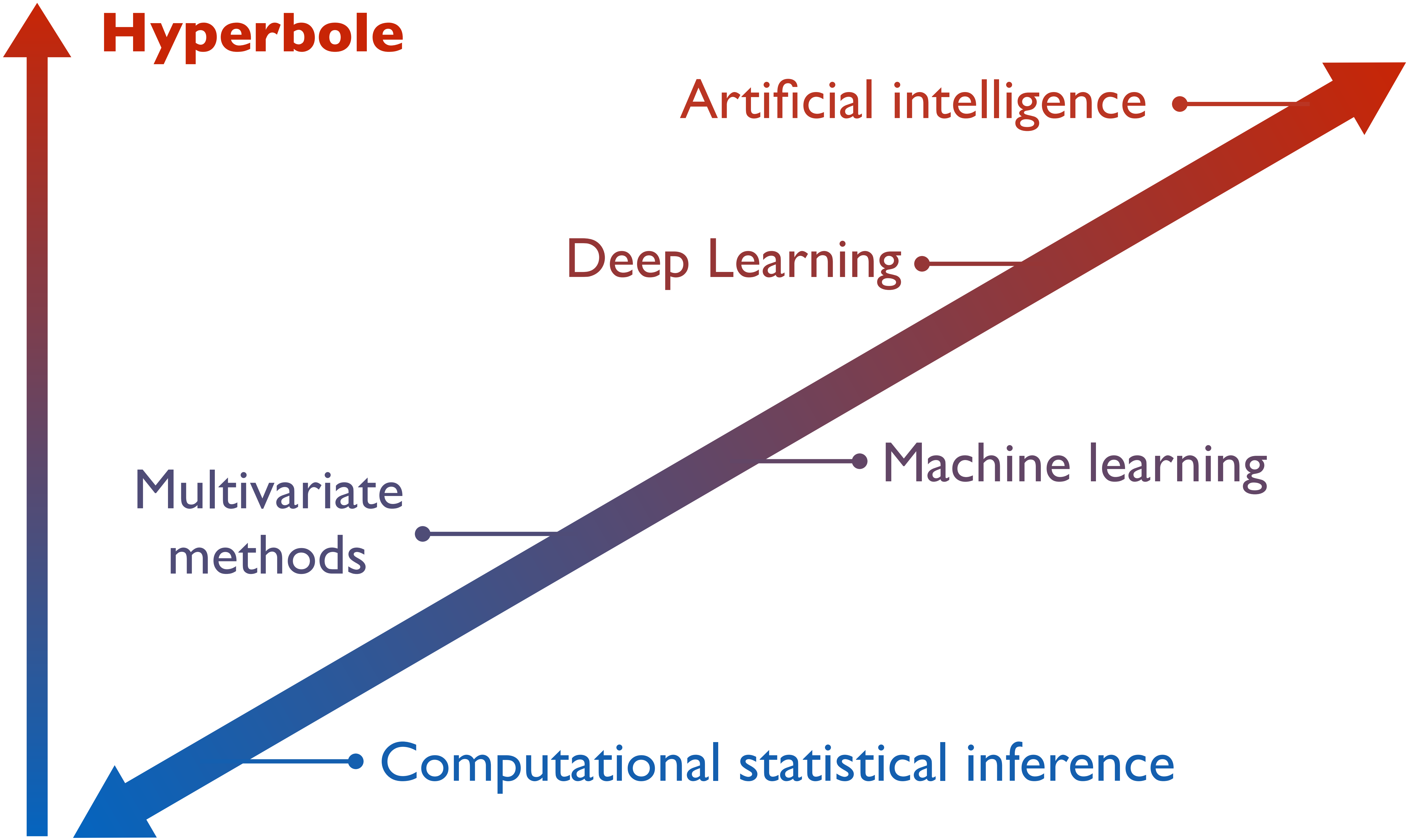
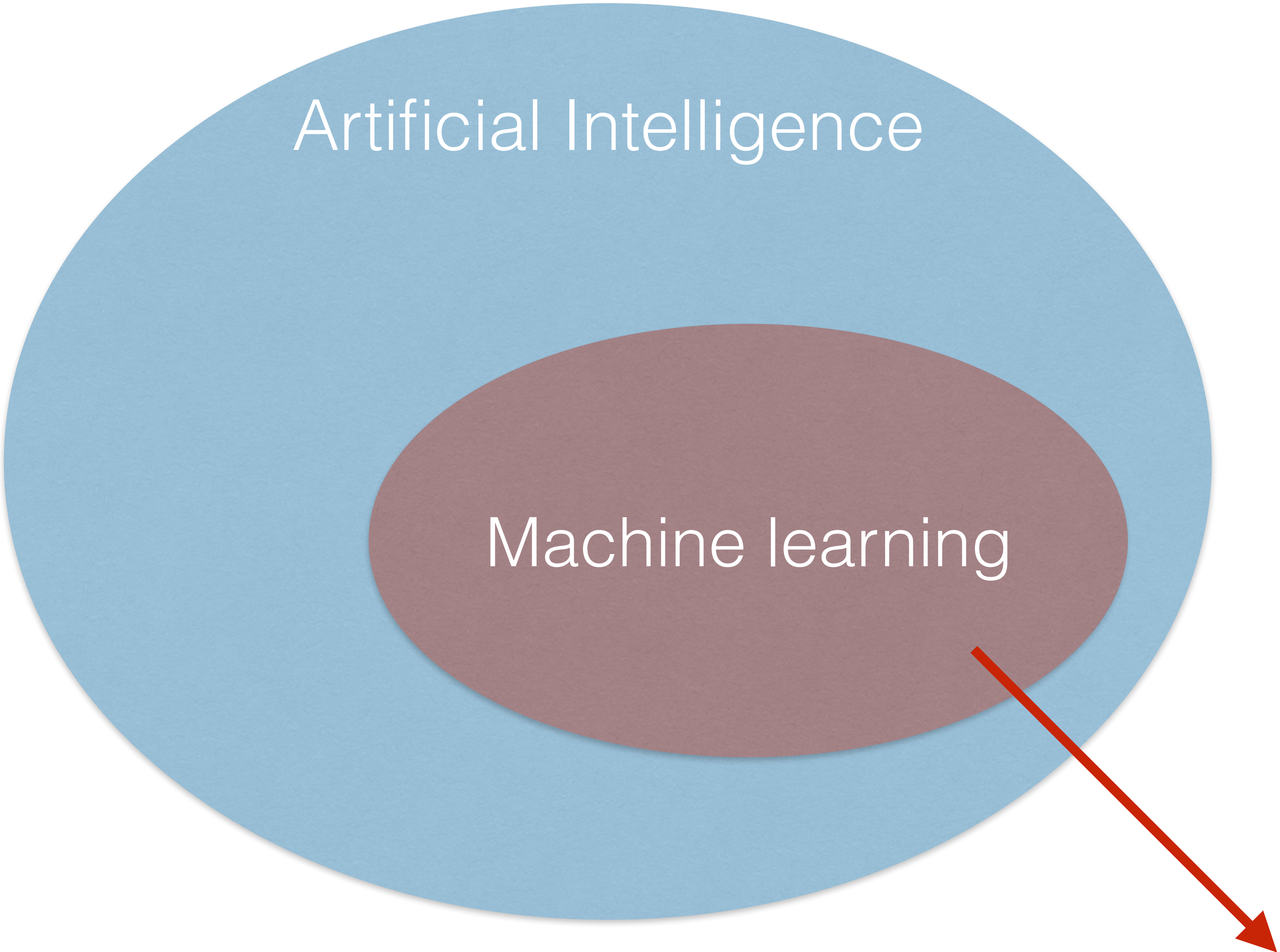
UiO : University of Oslo

Machine learning and anomaly detection: possible applications in distributed computing

James Catmore
University of Oslo

What is machine learning?

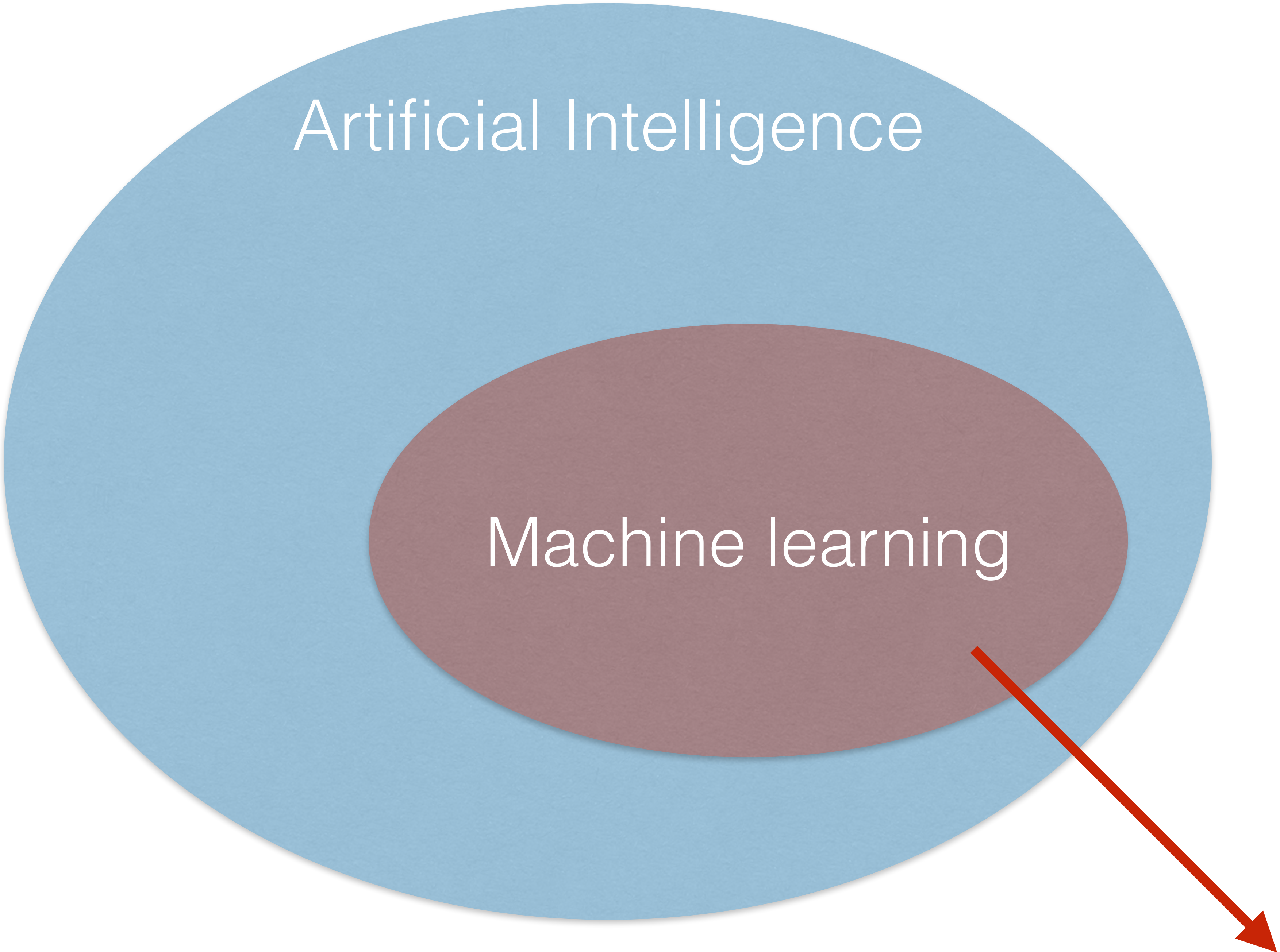




uses statistical inference to extract generalities from "training" data

→ **"learns" from the training data**

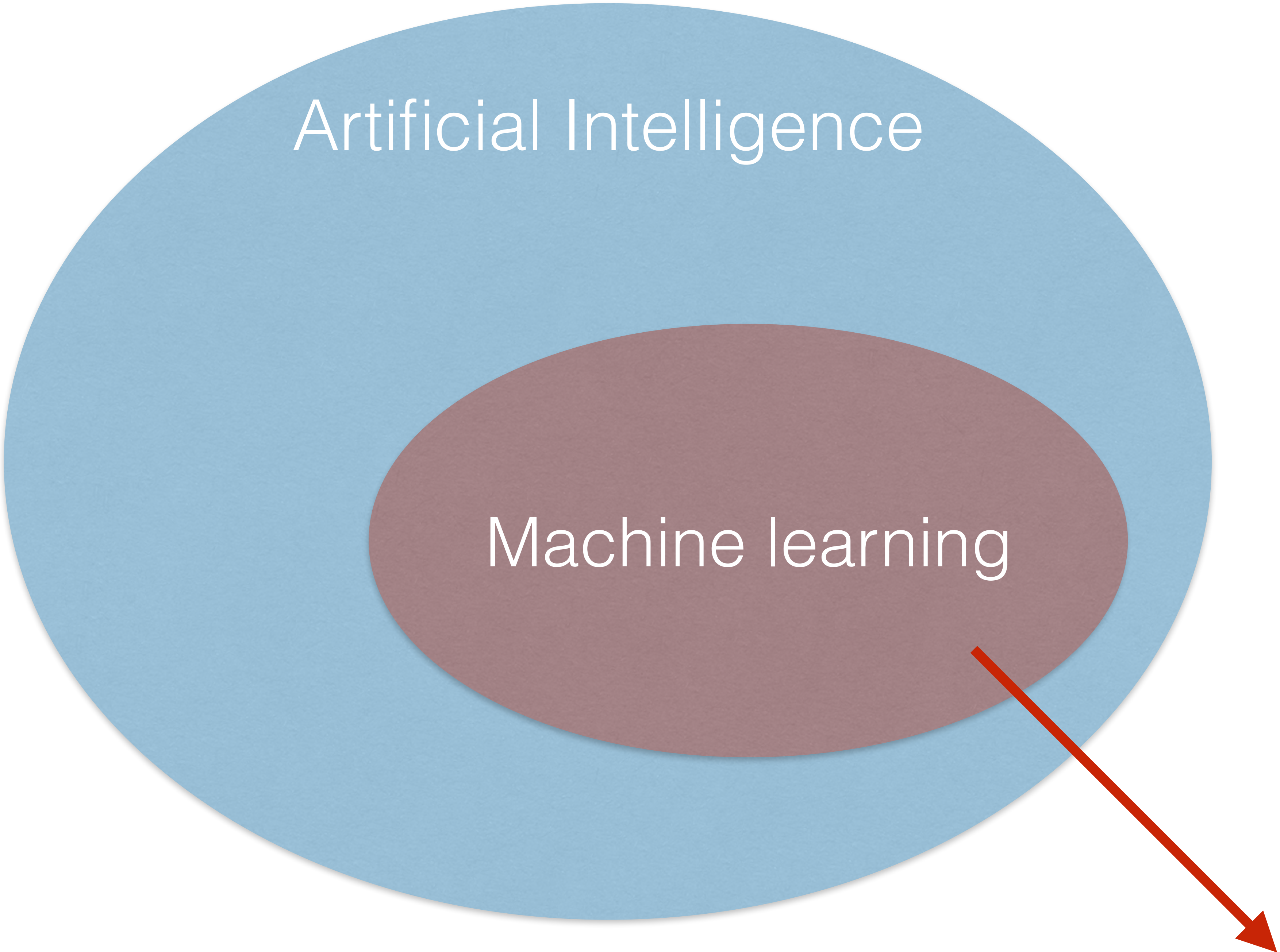
→ when exposed to new data, ***demonstrates behaviours that have not been explicitly programmed***



requires

✓ lots of computing power

✓ lots of training data



requires

✓ lots of computing power

✓ lots of training data

appropriate

This talk

- Where might machine learning have a role in distributed computing?
  - …and where might it not
- Optimising use of resources
- Anomaly detection
- Power efficiency and security

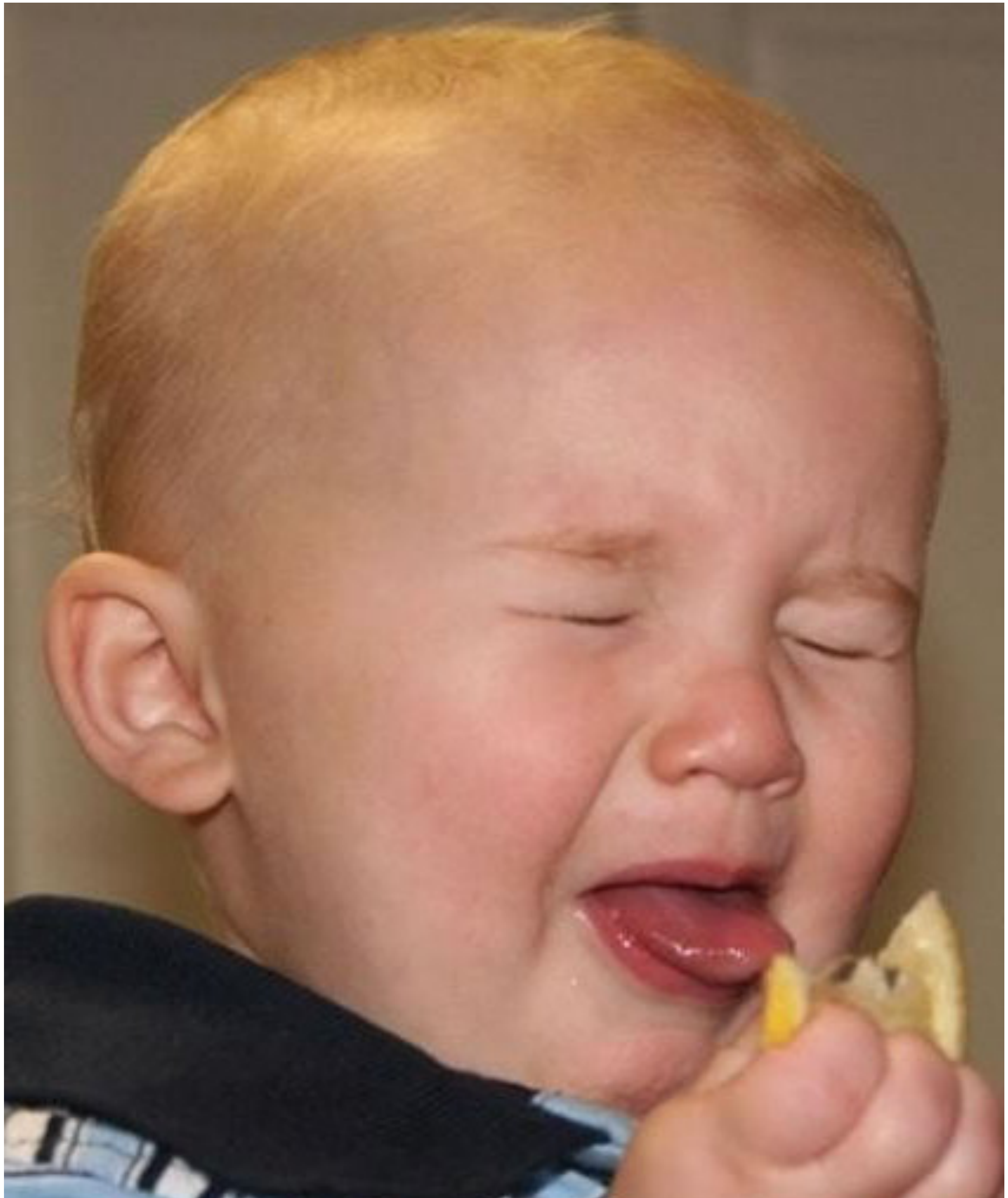
Main messages

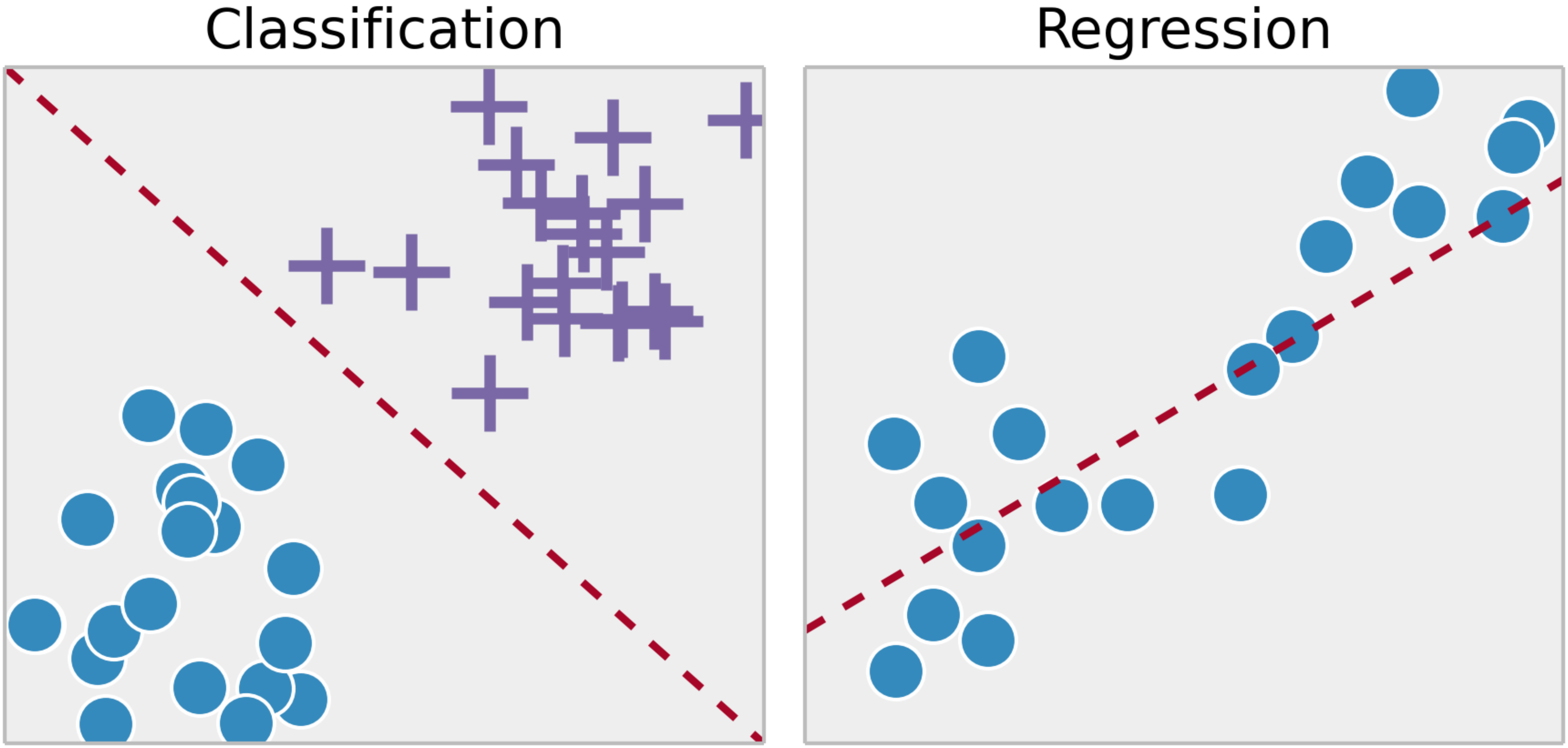
- Machine learning is only useful if there is a sufficient quantity of appropriate training data
  - Rubbish in = rubbish out
- More complicated models do not guarantee better performance
- We should always ask ourselves
  - Can the problem be solved with a simple procedural algorithm rather than a complex ML method?
  - Do we have a large enough set of relevant training data?
  - Is the chosen model best suited to the problem?

Supervised learning

Unsupervised learning

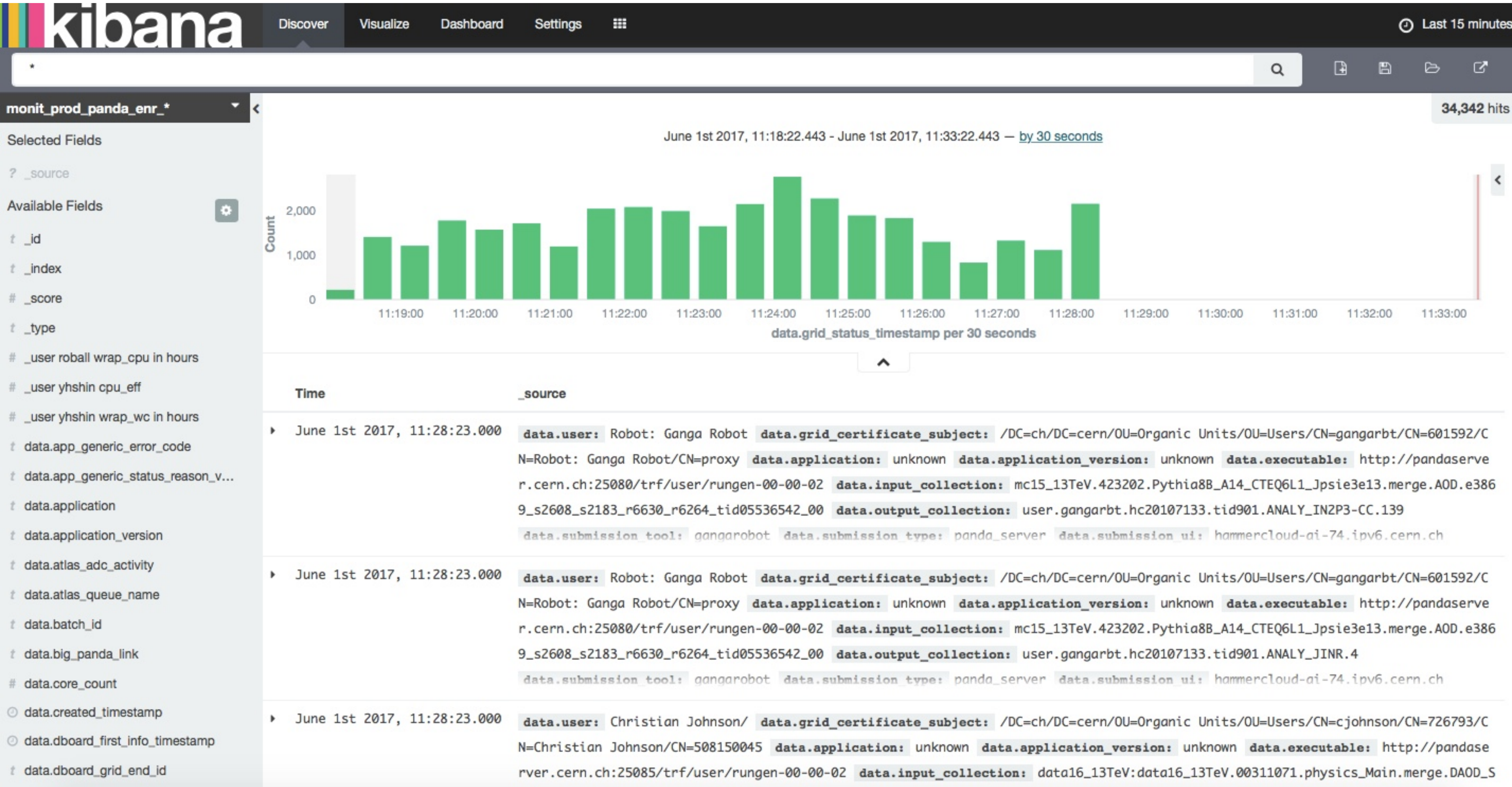
Reinforcement learning




Classification
Regression

Training data

Fortunately we have plenty of historical data from distributed computing operations… jobs, data movement, sites, etc etc



Optimising use of computing resources

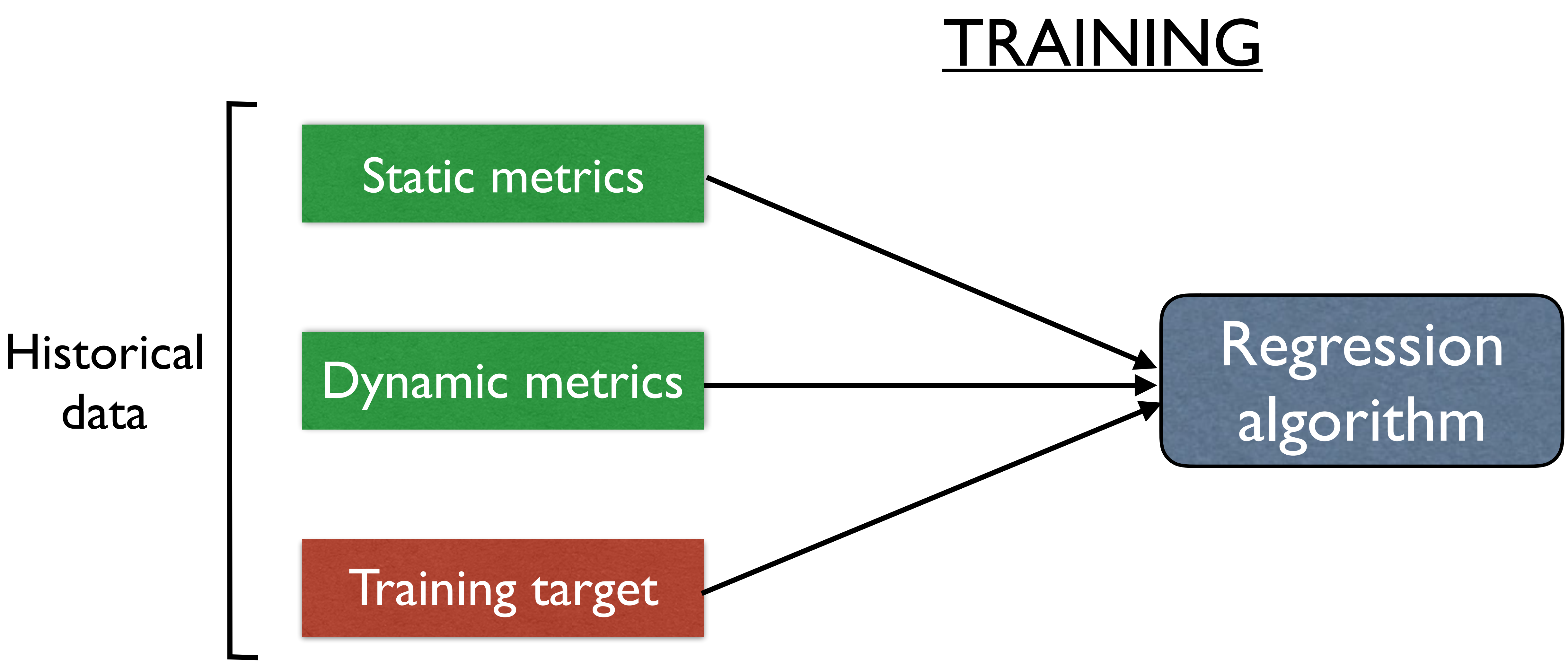
Optimising power utilisation efficiency

Anomaly detection

Security

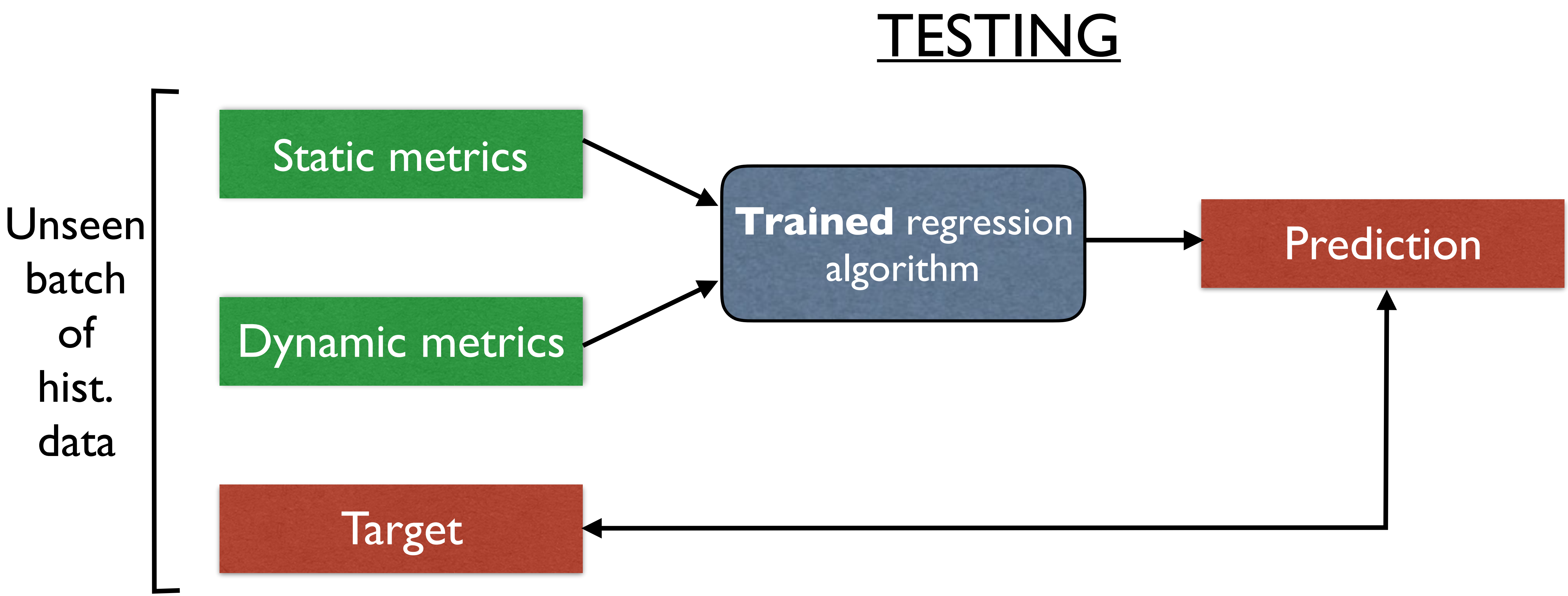
Optimising use of resources

- M. Lassnig et al @ CHEP2016, contribution 131
  - *Using machine learning algorithms to forecast network and system load metrics for ATLAS Distributed Computing*
  - https://indico.cern.ch/event/505613/contributions/2227924/attachments/1346952/2031409/Oral-131.pdf



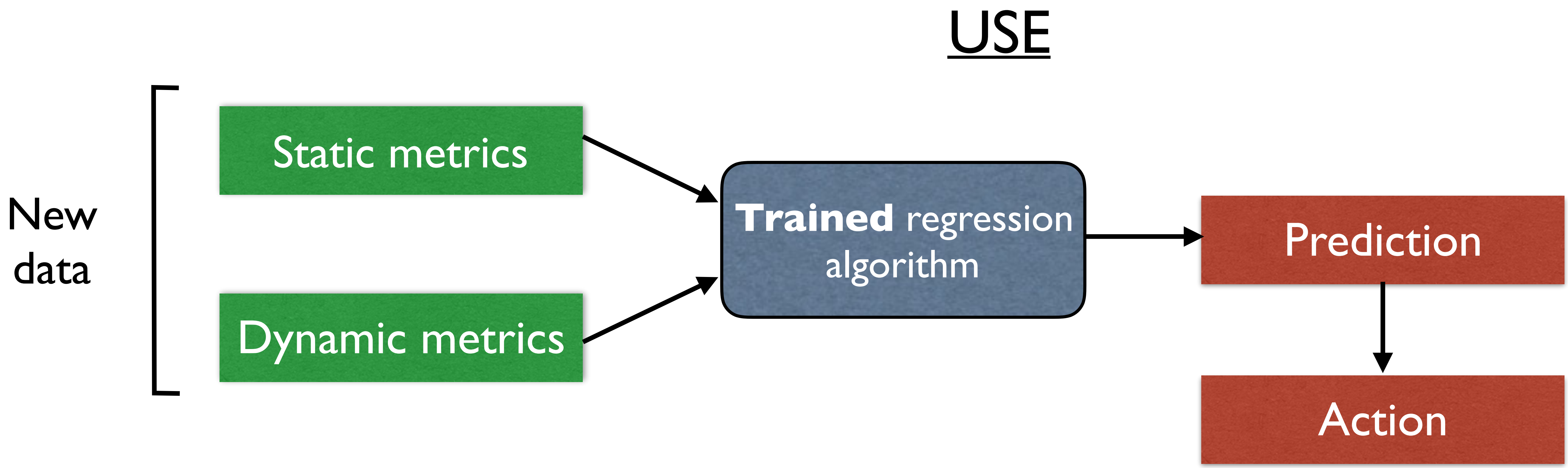
Optimising use of resources

- M. Lassnig et al @ CHEP2016, contribution 131
  - *Using machine learning algorithms to forecast network and system load metrics for ATLAS Distributed Computing*
  - https://indico.cern.ch/event/505613/contributions/2227924/attachments/1346952/2031409/Oral-131.pdf



Optimising use of resources

- M. Lassnig et al @ CHEP2016, contribution 131
  - *Using machine learning algorithms to forecast network and system load metrics for ATLAS Distributed Computing*
  - https://indico.cern.ch/event/505613/contributions/2227924/attachments/1346952/2031409/Oral-131.pdf



Optimising use of resources: DDM example

- M. Lassnig et al @ CHEP2016, contribution 131
- Basic idea:
  - ATLAS distributed data management system involves a heterogeneous infrastructure with a highly dynamic state
  - Human interaction is important - "signing off" decisions and tasks; algorithms and their parameters tuned based on experience
  - Potential for improvement
    - Data rebalancing: disk space doesn't match CPU
    - Placement selection: where to put data?
    - Source selection: where to run jobs if multiple input copies available?
    - Robustness: automatically reschedule tasks/transfers

DDM Network Metrics

Centrally collect and make available DDM metrics to help with those problems

Static link metrics

Static metrics

- ↪ **Source** and **destination** site
- ↪ **Closeness** as defined by ATLAS Distributed Computing Operations, updated monthly

Dynamic link metrics

Dynamic metrics

- ↪ **Packetloss** as a percentage [perfSONAR]
- ↪ **Latency** as median one-way link delay [perfSONAR]
- ↪ **Percentile File Throughput** in mbps [FTS, Dashboard, FAX]
- ↪ **Link Throughput** in mbps [perfSONAR]
- ↪ **Queued files** per activity [Rucio]
- ↪ **Done files** per activity in the last *n* hours [Rucio]

First objective: Heavy Ion placement

Training target

minimise	**job waiting time**	t[activated - defined]
subject	**limited number of potential sites**	with himem queues
	existing data across	all sites
	available free space at	potential destination sites
	DDM network metrics	latency, packetloss, throughput, closeness
	all involved queue lengths	prodsys, panda, rucio
learn	**for all heavy ion data subject to given constraints → classify destination sites**	

Place or rebalance heavy ion data as close as possible to potential scheduling targets
Constrained learning function with all input and output metrics available

Regression algorithm

Time to complete transfer estimator

Close in the geographical sense is misleading, instead train an estimator

- ↪ **Learn input** DDM network metrics, including categorized variates
- ↪ **Model input** (*bytes, source, destination, activity*)
- ↪ **Model output** *file transfer duration*

Data Consolidation, T0 Export, Production Input, etc...

full workflow, including queues, not only time on the network

Method uses decision trees

- ↪ Effective and efficient tool for classification and regression of large datasets
- ↪ Finds nonlinear relationships between variates

Cross-validation against overfitting

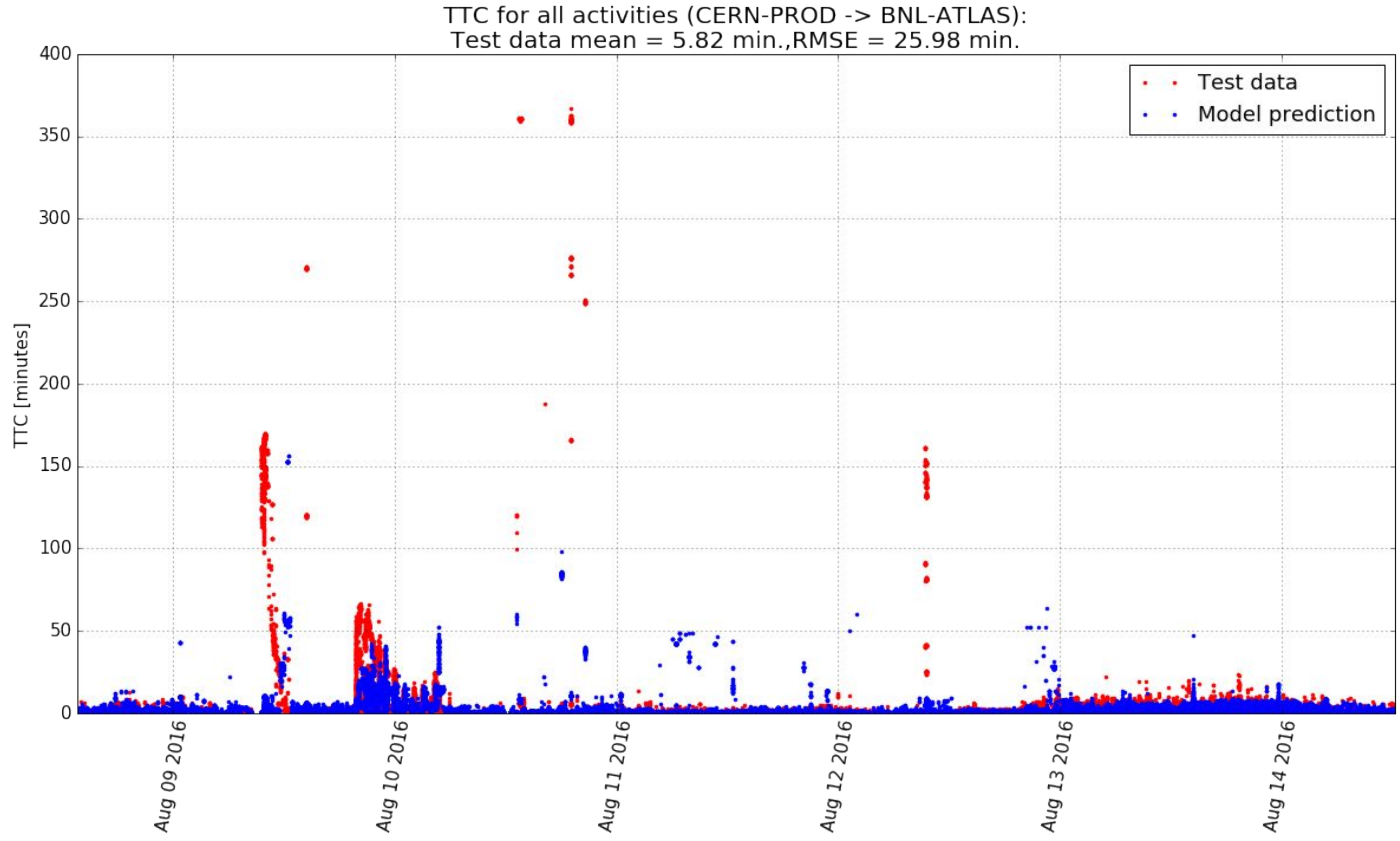
- ↪ Many random samples generated, each with 80% training, 20% test split
- ↪ Each sample fitted with separate tree, in our first evaluation 1 month of data used

Random forest regressor of many trees

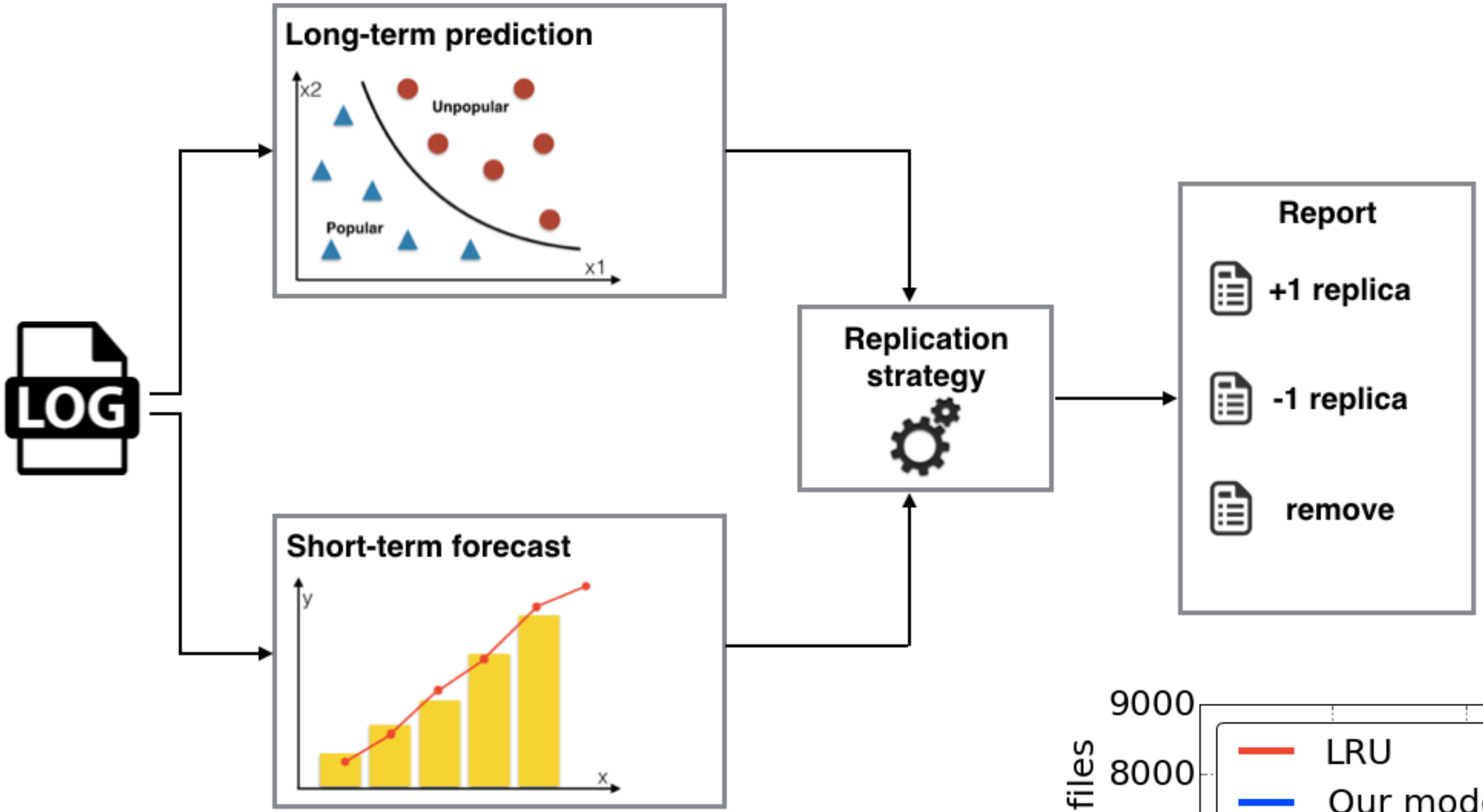
- ↪ Final prediction which is robust to outliers and noise (Breiman, 2001)

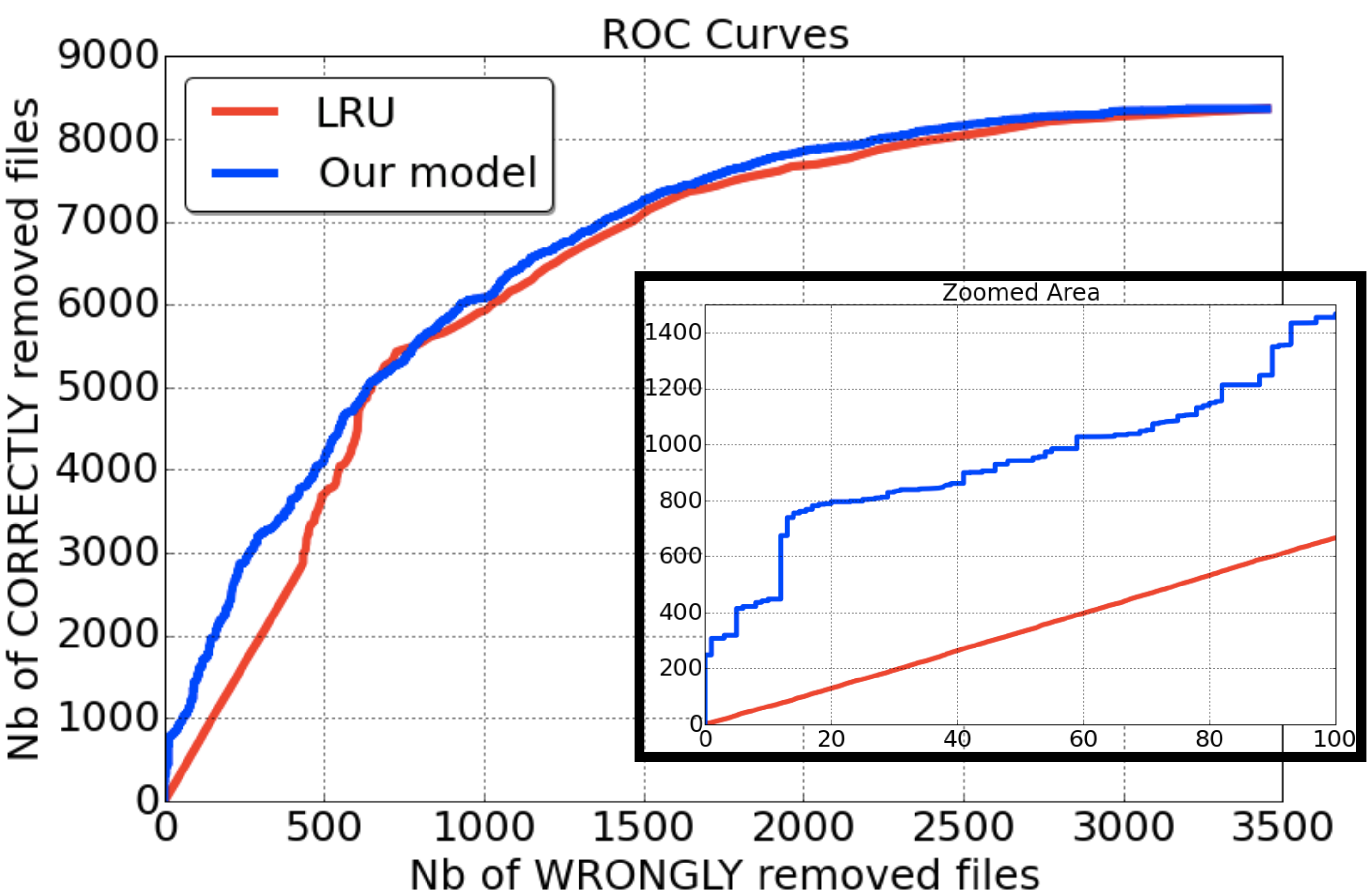
Prediction

Time to complete transfer estimator



Storage optimisation for LHCb





Mikhail Hushchyn, CHEP 2016, https://indico.cern.ch/event/505613/contributions/2230916/attachments/1347081/2044978/Oral-295-v2.pdf

Optimising use of resources: next steps

- Use predictions of regression algorithms to optimise data placement?
  - incrementally, adding as a weight to existing placement algorithms?
- Another idea: can we extend to job placement?
  - Are there sufficient metrics available to be able to make useful predictions?
  - More important in a cloud computing environment?
    - Costs of CPU/network/storage become metrics…
- ATLAS qualification task for Simen Hellesund (UiO)
- Key component of the "Archestrate" application

Optimising use of computing resources

Power utilisation efficiency

Anomaly detection

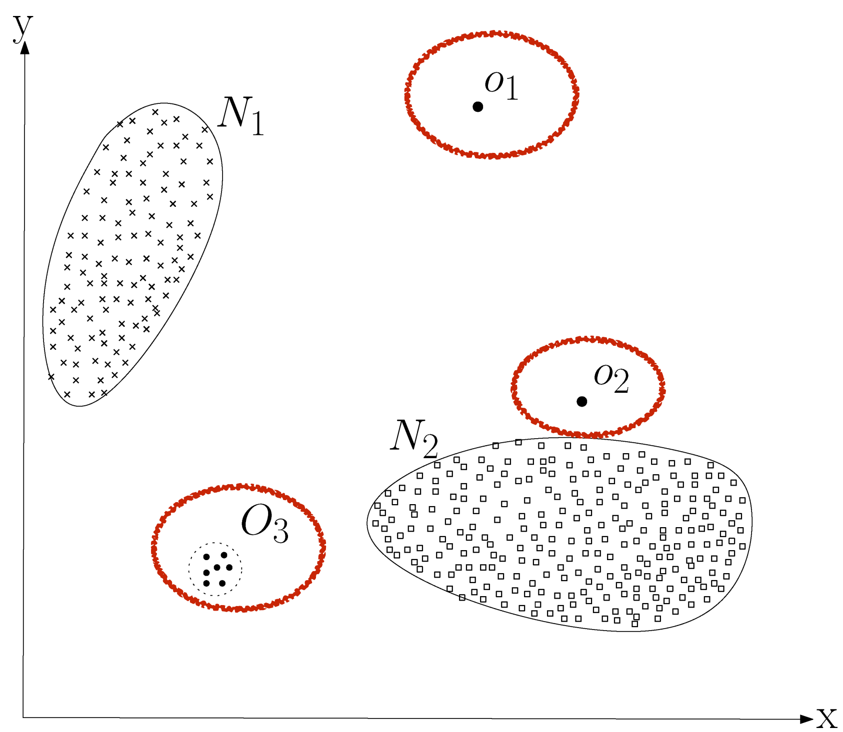
Security

What ~~is~~ I think anomaly detection~~?~~ is

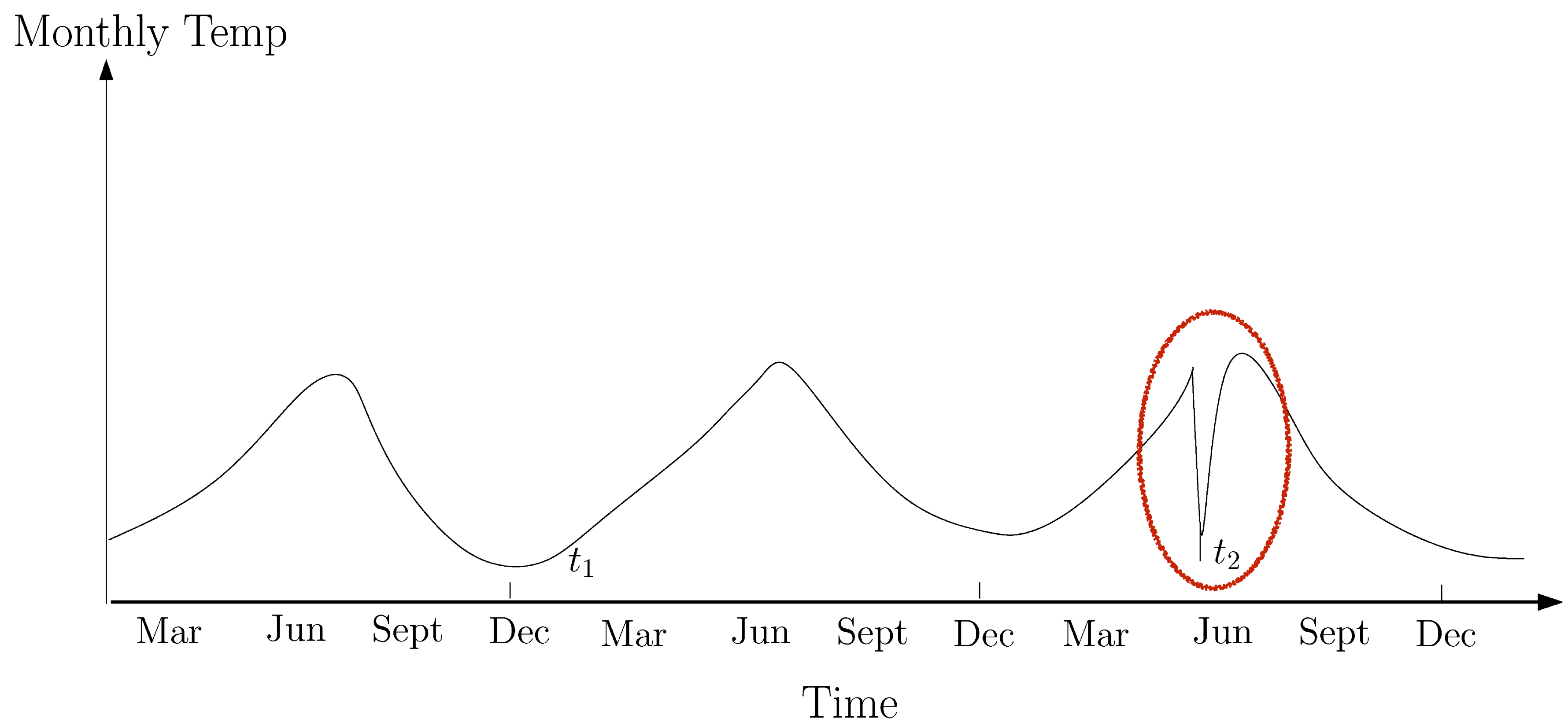
- Automatic identification of data instances (events) that are in some way different from the bulk of the data and which need detailed scrutiny by experts. Usually implied:
  - produced by a different mechanism than the bulk of the events
  - small number of anomalies w.r.t. the main part of the data
- Can be
  - supervised: train to recognise specific anomalous cases
  - semi-supervised: train only on the bulk of the data without anomalies → strong relation to one-class classification
  - unsupervised: algorithm automatically identifies the bulk by some means and thence the anomalies
- Difficult problem because in general we don't know what the anomalies look like, and there may be very few of them
  - Testing is particularly challenging: how do we evaluate the performance of an algorithm on a type of event that we have never seen before?

Types of anomaly

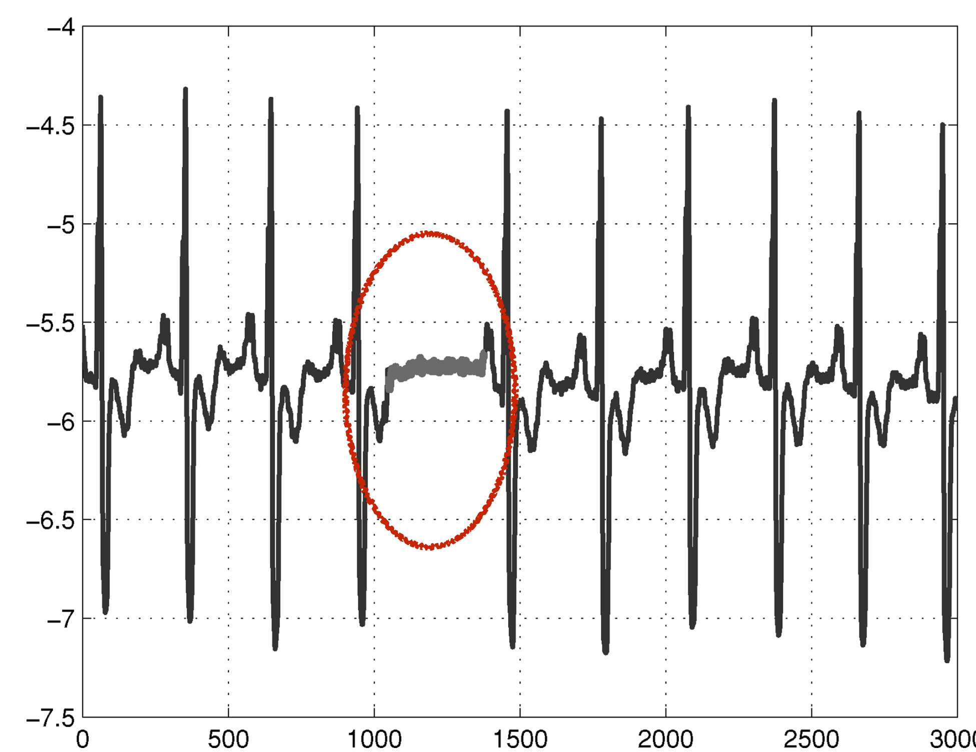
POINT



CONTEXTUAL



COLLECTIVE (population-level)

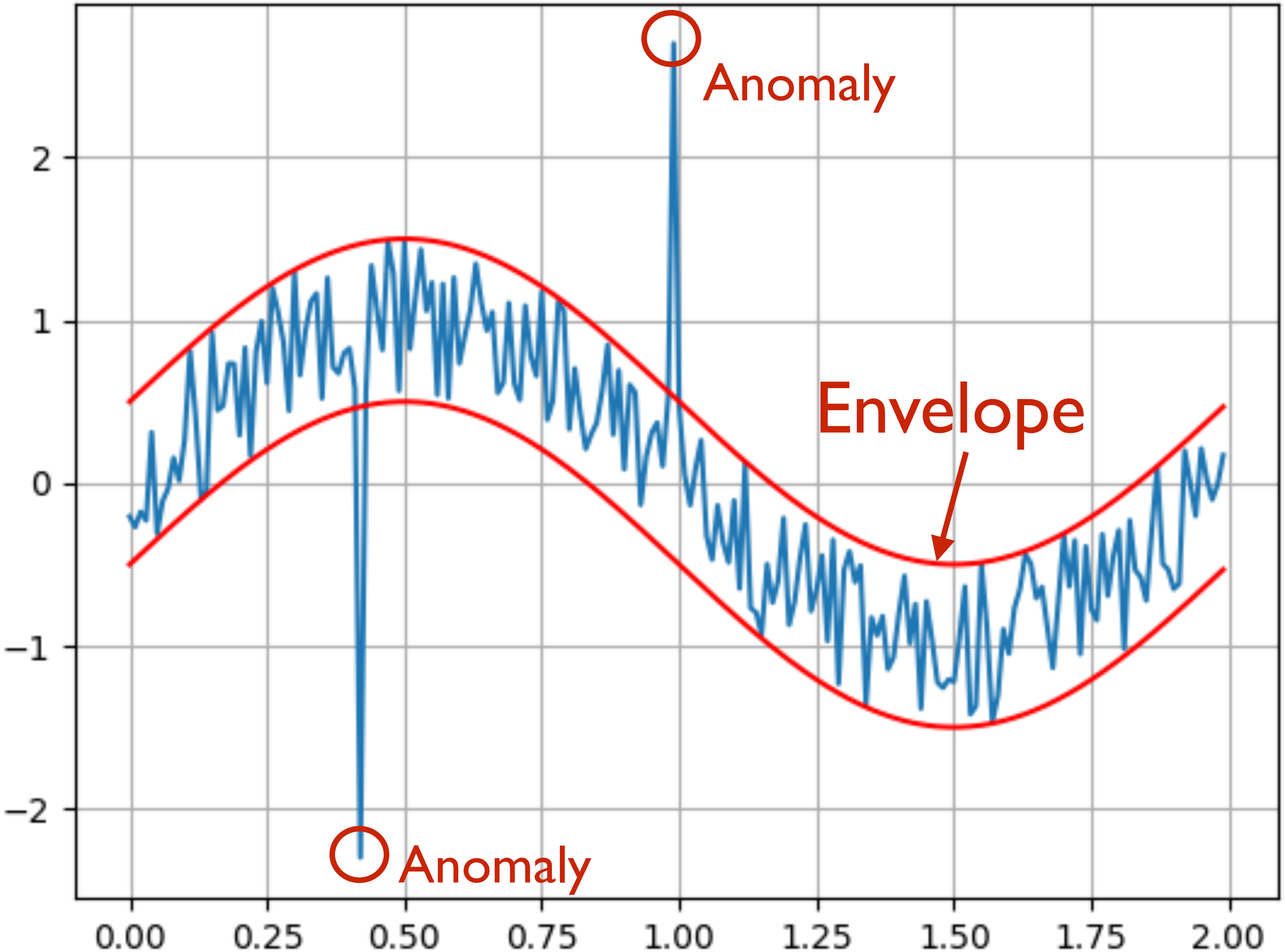


Images:
ACM Computing Surveys,
Vol 41, No. 3, Article 15

Anomaly detection applications in Grid computing

- Detect abnormal performance and alert shifters
  - Network
  - Disk/tape activity
  - Time to complete jobs
  - Memory consumption, etc
  - Intrusion detection
- Conversely, identify jobs/transfers/tasks which are causing error messages but are **not anomalous** and will probably go away
  - Saves time of shifters

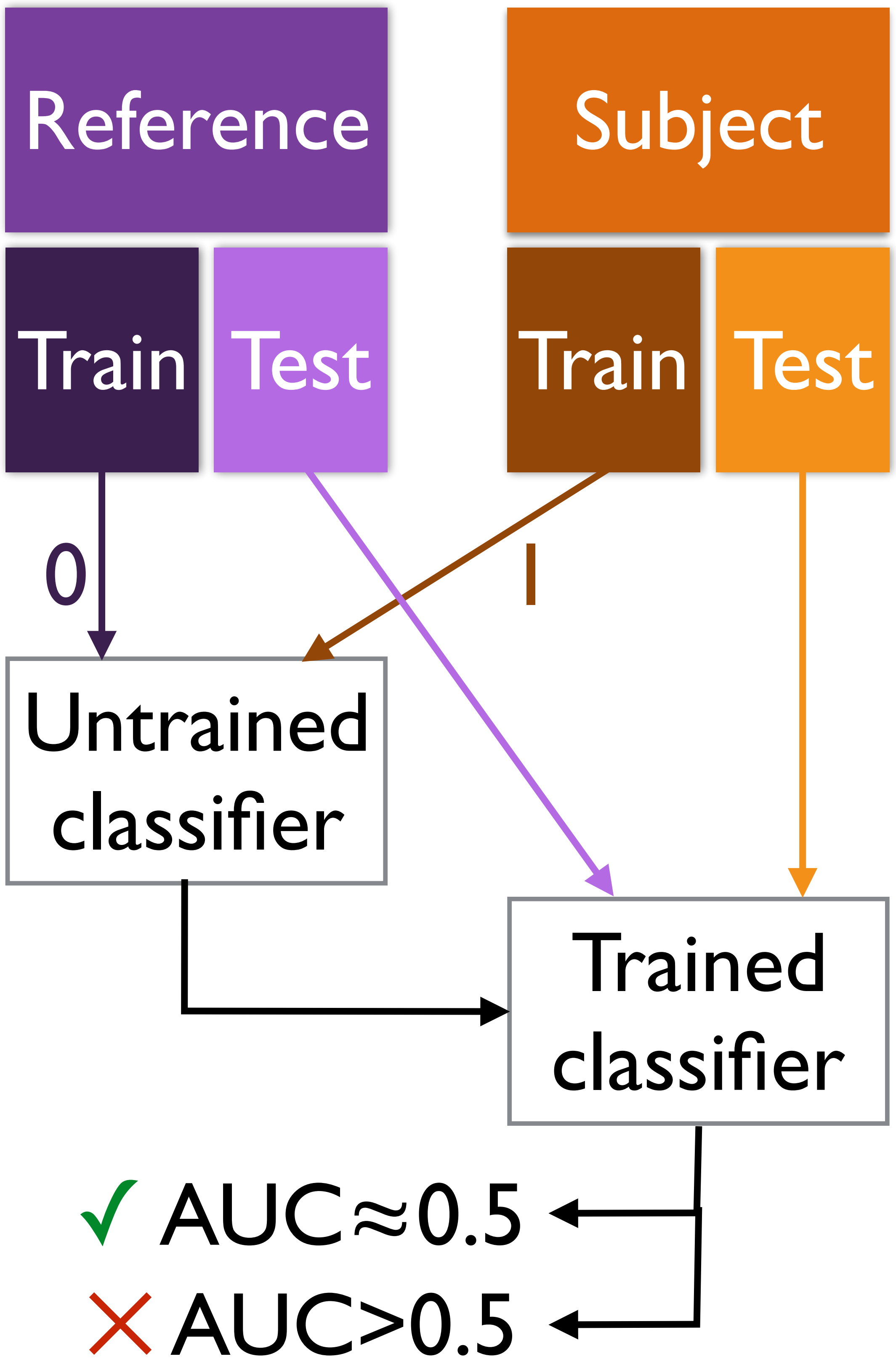
Anomaly detection: time series analysis



- Wide variety of techniques including multi-variate methods
- Range from very simple (moving averages, fits) to highly sophisticated (recurrent NNs)
- Determining the tolerance is a big part of this

Anomaly detection: split sample classifier

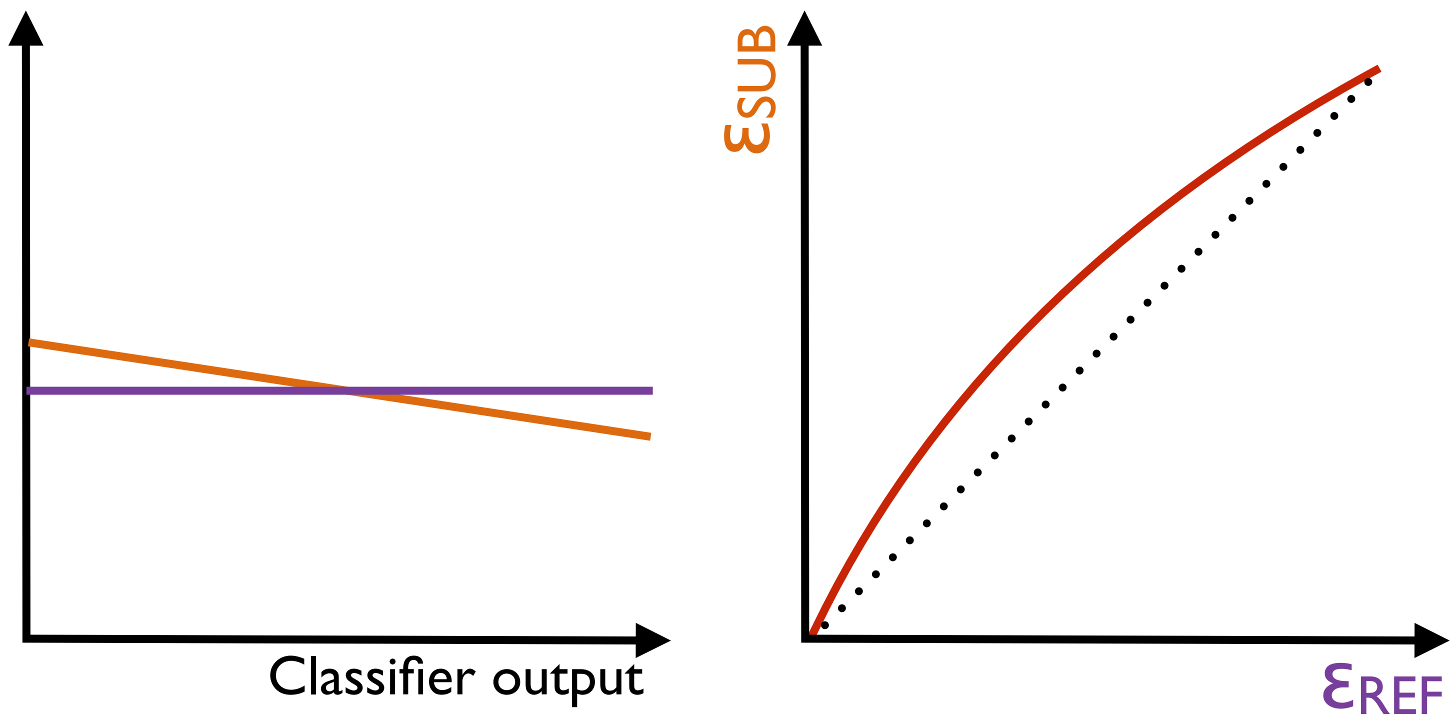
- Suitable for collective anomalies only
- Two samples: reference and subject
  - We want to see if the subject is consistent with the reference
- Split both into two parts - training and testing
- Train a classifier (BDT, NN etc) to distinguish between the test and reference (e.g. as if they were "signal" and "background")
- In the testing phase, see if the classifier makes any progress in separating the reference and subject
  - If it does: there is something about the subject sample to distinguish it from the reference
  - If it is supposed to be the same as the reference, clear evidence that something is wrong



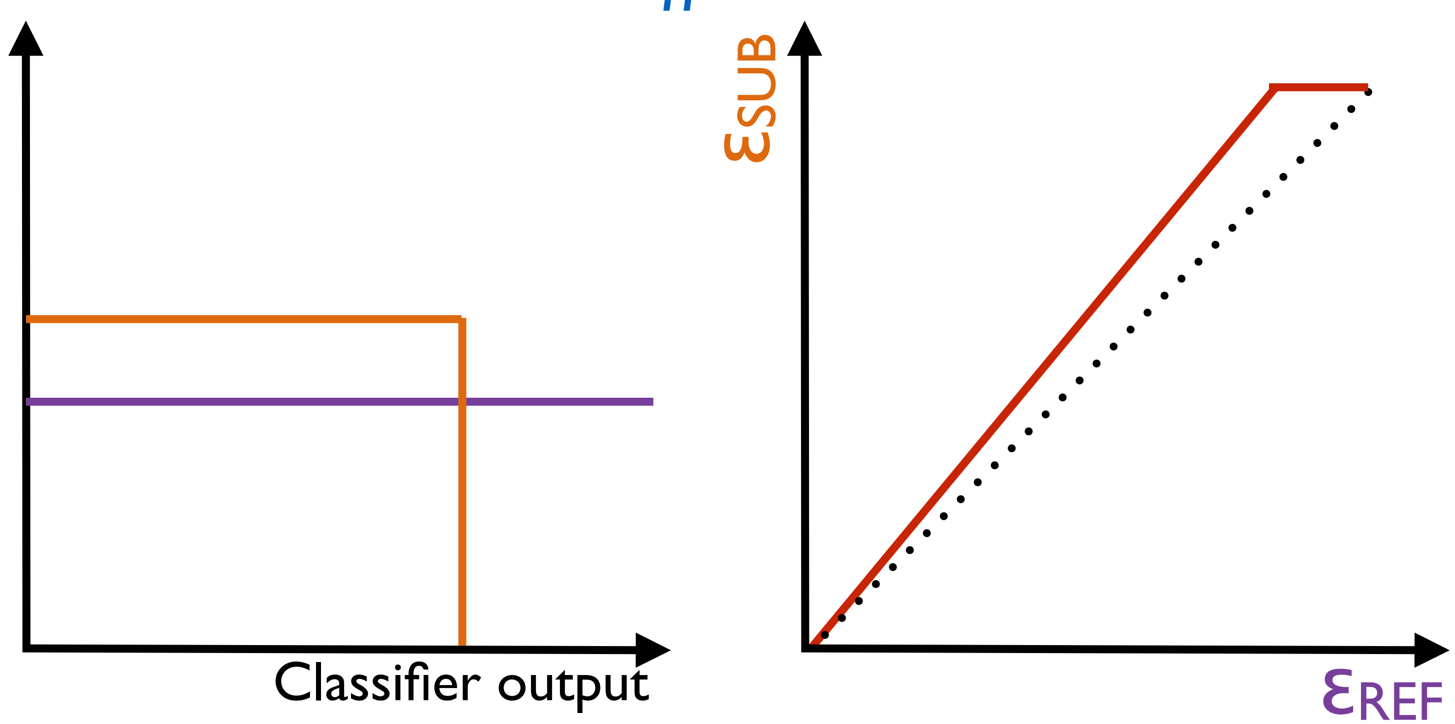
Anomaly detection: split sample classifier

- Shape of the ROC curve may help to understand whether the problem is local or global
- Inspection of the BDT/NN weights may allow us to work out which combination of variables are leading to the separation

Global difference

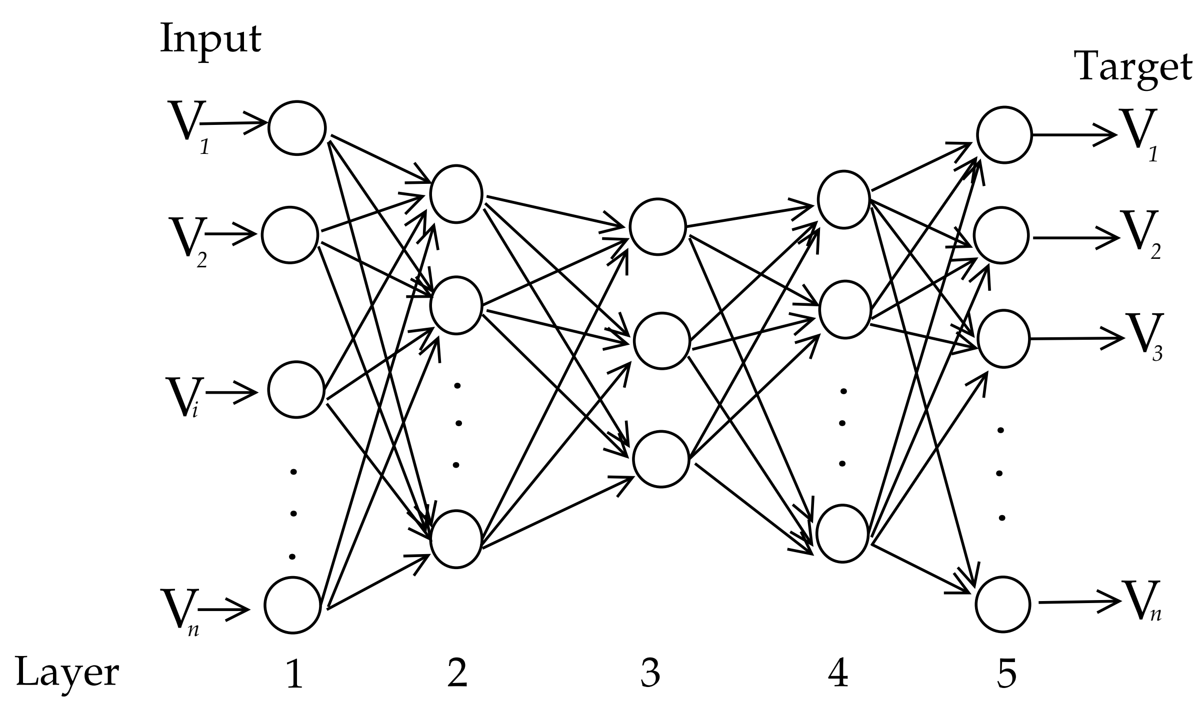


Local difference



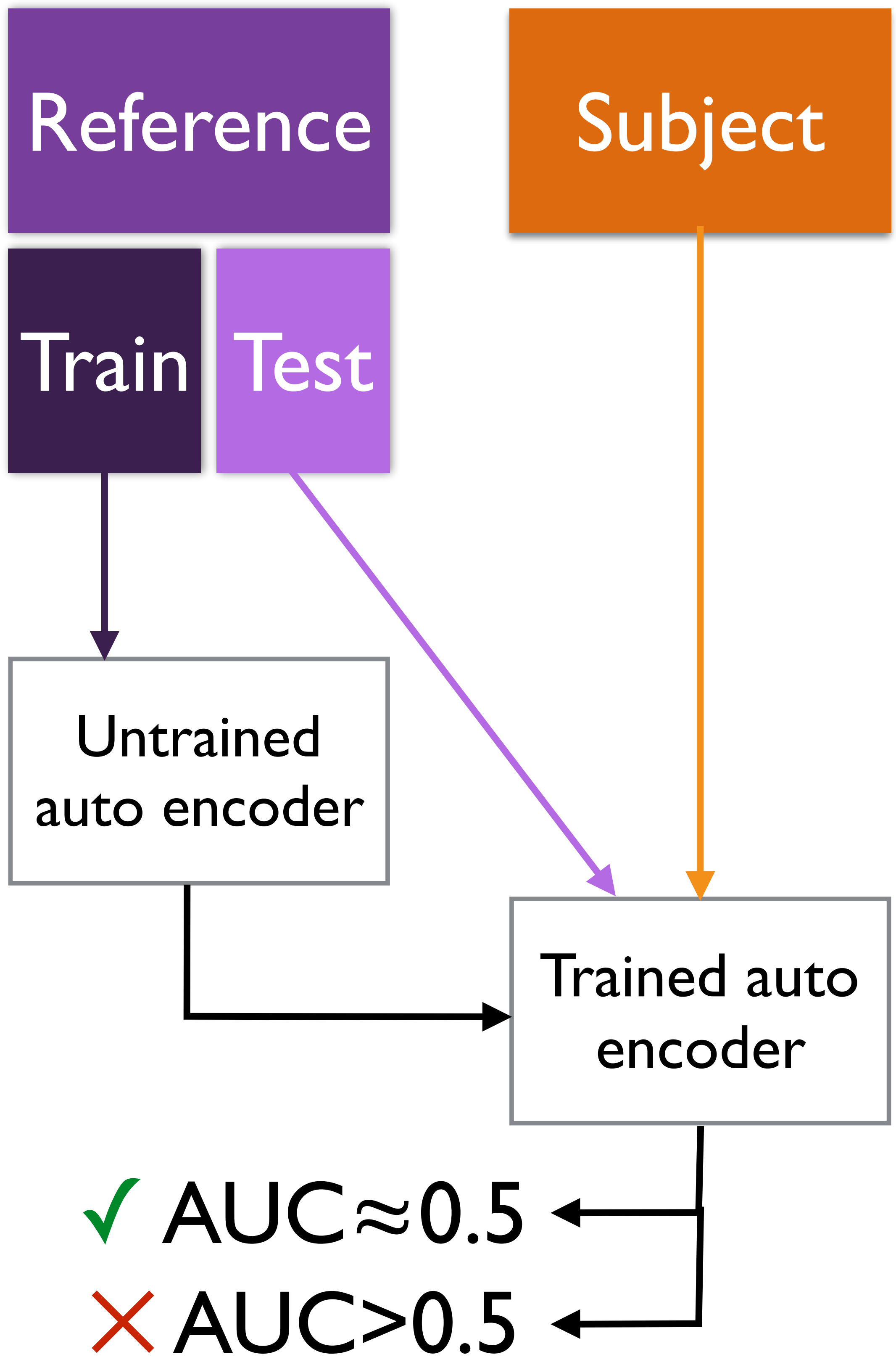
Anomaly detection: auto-encoder

- Auto-encoder: NN trained on its own input - suitable for collective or point anomalies
  - Usually includes a bottle-neck to compress the features of the data (e.g. PCA)



- Normally used for dimension-reduction but proposed as a means of anomaly detection
- Idea: a trained replicator neural network should reconstruct new examples taken from the bulk (normal) data with low error, but when presented with an anomalous example, will reconstruct it with a high error since it contains qualities that have not previously been encoded
  - Provides a natural measure of abnormality: the reconstruction error (difference between the input and the output)
  - Reconstruction error per event = $\sum_{i=1}^{N}(x_i^{in} - x_i^{out})^2$ N is the number of features

Anomaly detection: auto-encoder

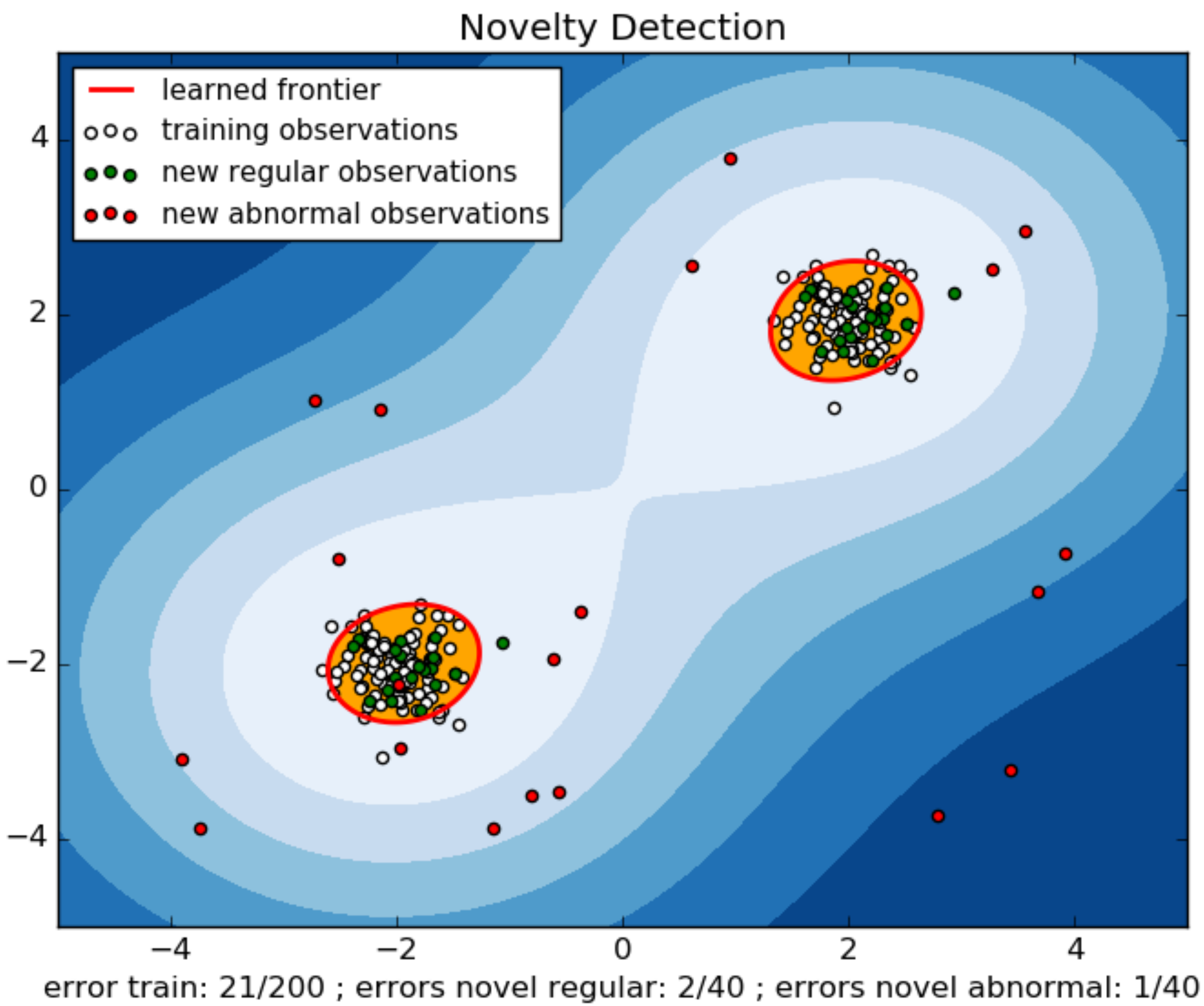


<u>"One class" classifier</u>

Drawback: less discrimination power

Advantage: no need to train on subject data, fast evaluation

Anomaly detection: surface/density-based



- Abnormal cases likely to be separated in variable space from normal cases
- Form a boundary around the normal cases (one class SVM), abnormal cases beyond the boundary
- Need to worry about tolerances (and evaluation time)

Optimising use of computing resources

Power utilisation efficiency

Anomaly detection

Security

Power utilisation efficiency (PUE)

- Example: study by Google/DeepMind
- PUE = ratio of the total building energy usage to the IT energy usage
    - Not easy to model due to high complexity of data centre cooling equipment and vast number of potential configurations, non-linear relations between equipment and environmental conditions

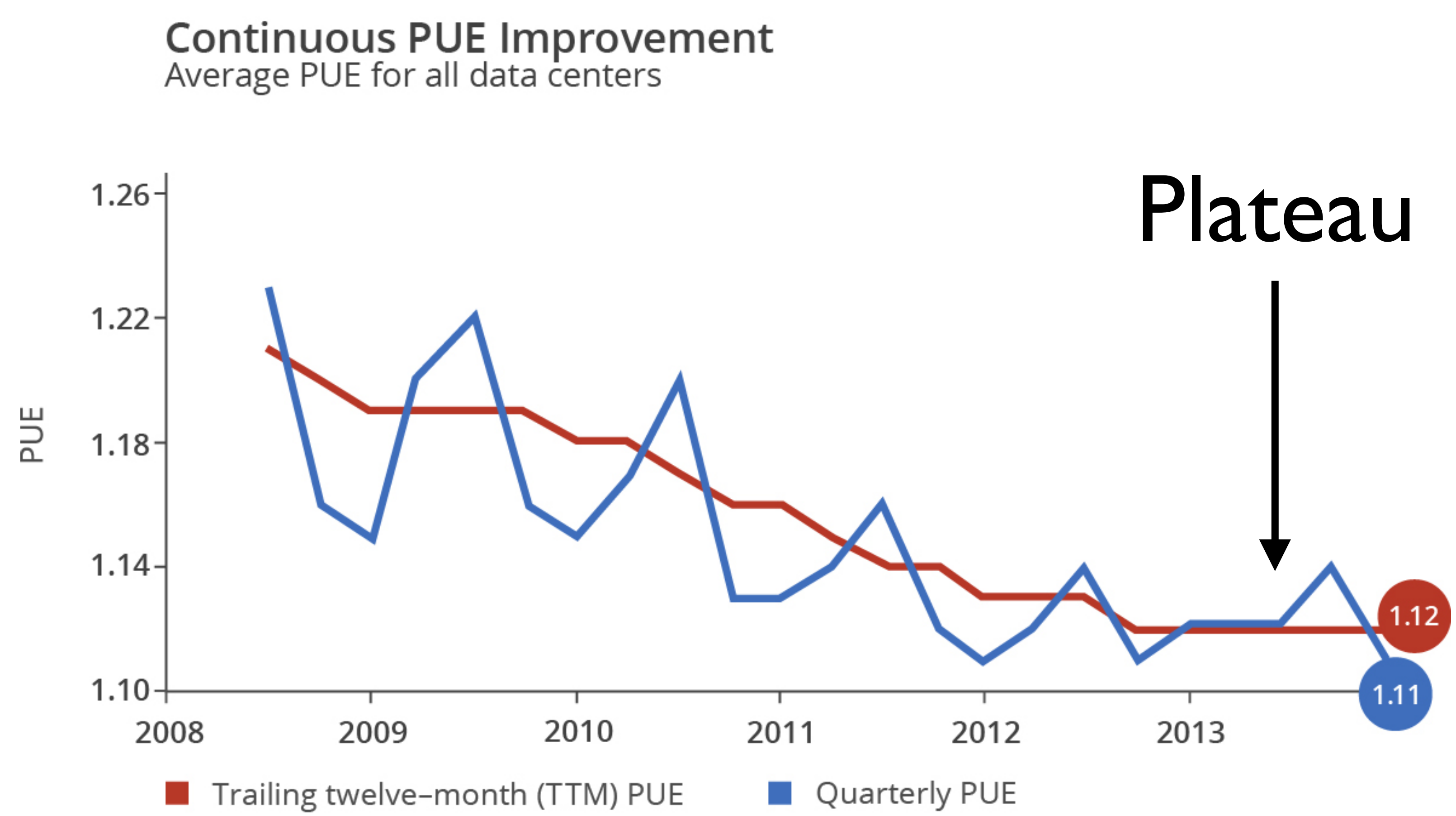


Fig 1. Historical PUE values at Google.

Power utilisation efficiency (PUE)

- Use an ensemble of deep neural networks to learn the PUE w.r.t. historically measured data, for a large number of parameters
- Use the trained networks to predict PUE under a range of conditions
  - enabling optimisation of the data centre to minimise PUE
- Remark: similar task to the Grid performance optimisation work…?

1. Total server IT load [kW]
2. Total Campus Core Network Room (CCNR) IT load [kW]
3. Total number of process water pumps (PWP) running
4. Mean PWP variable frequency drive (VFD) speed [%]
5. Total number of condenser water pumps (CWP) running
6. Mean CWP variable frequency drive (VFD) speed [%]
7. Total number of cooling towers running
8. Mean cooling tower leaving water temperature (LWT) setpoint [F]
9. Total number of chillers running
10. Total number of drycoolers running
11. Total number of chilled water injection pumps running
12. Mean chilled water injection pump setpoint temperature [F]
13. Mean heat exchanger approach temperature [F]
14. Outside air wet bulb (WB) temperature [F]
15. Outside air dry bulb (DB) temperature [F]
16. Outside air enthalpy [kJ/kg]
17. Outside air relative humidity (RH) [%]
18. Outdoor wind speed [mph]
19. Outdoor wind direction [deg]

Power utilisation efficiency (PUE)

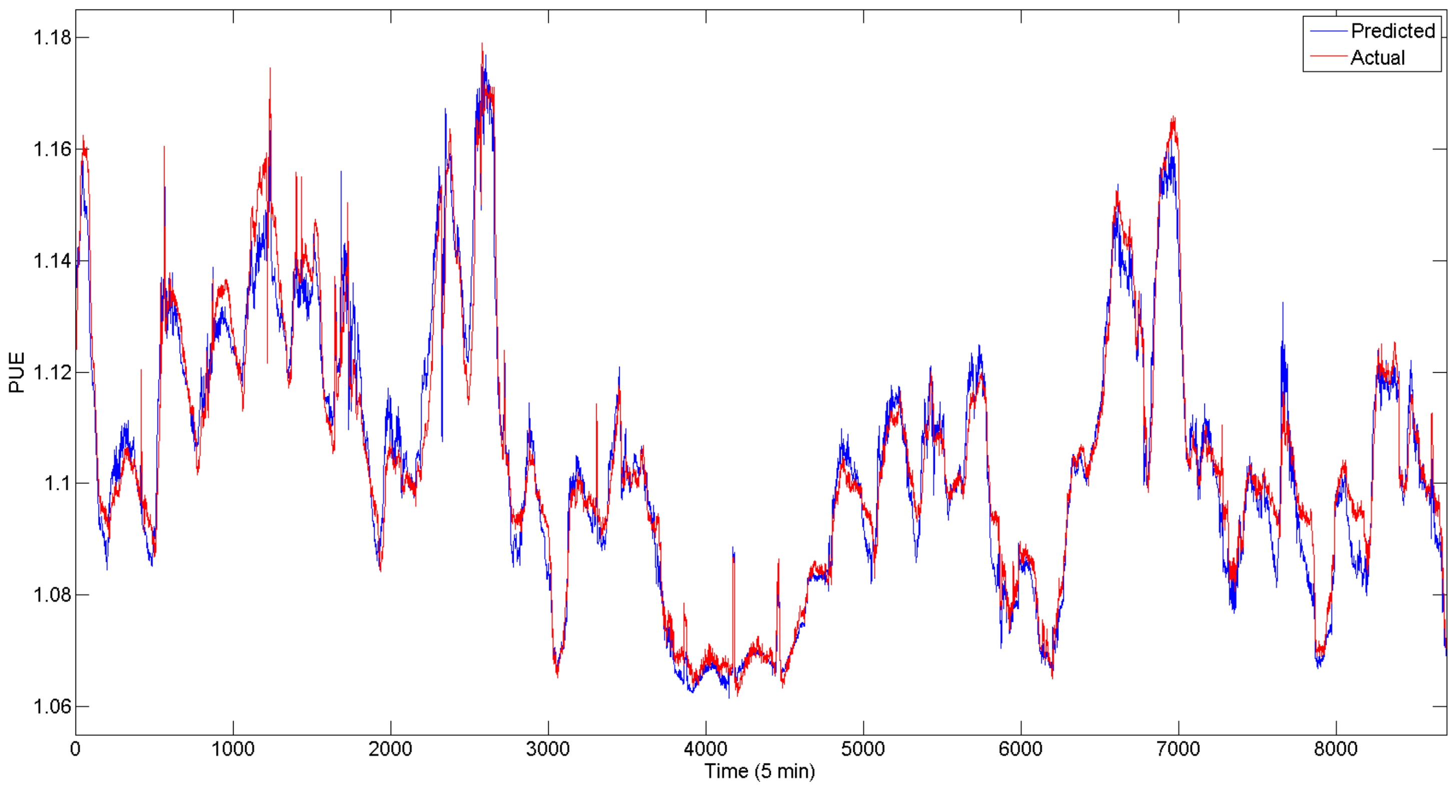
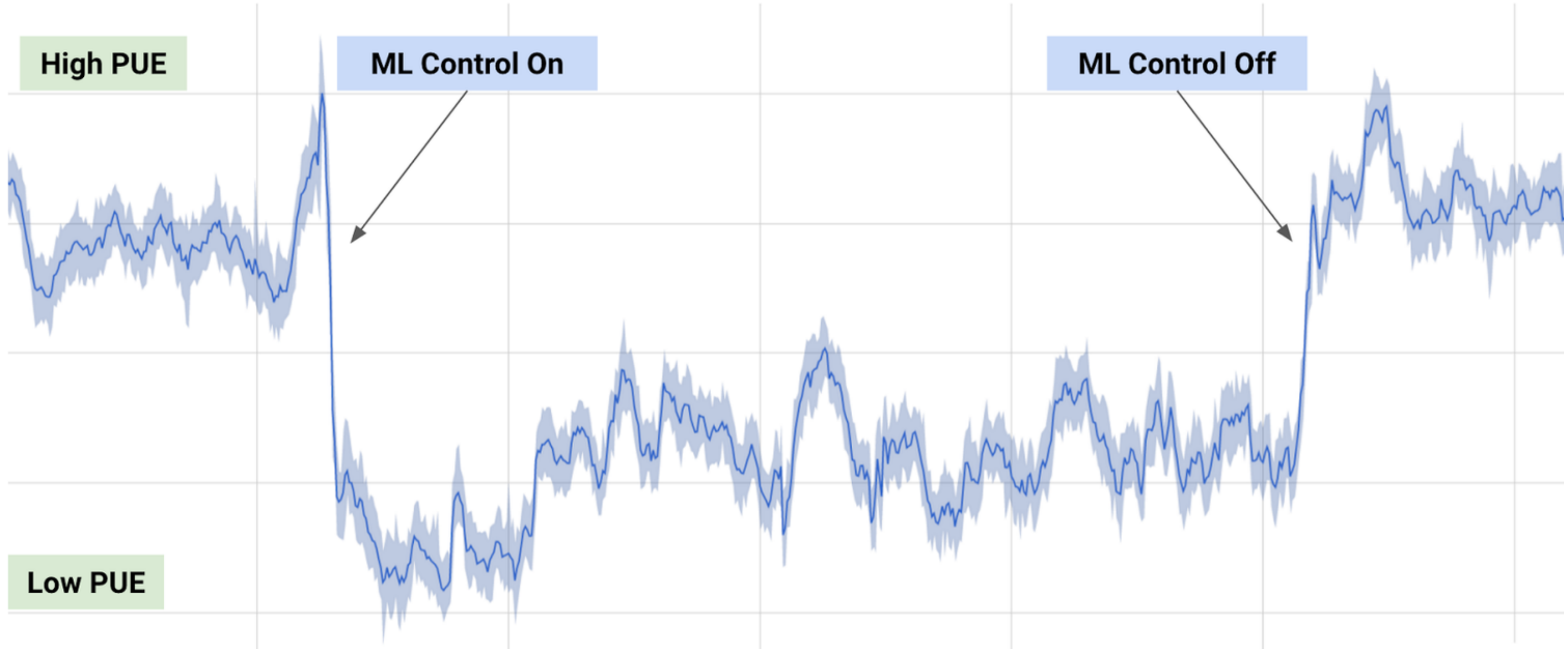


Fig. 3 Predicted vs actual PUE values at a major DC.



Optimising use of computing resources

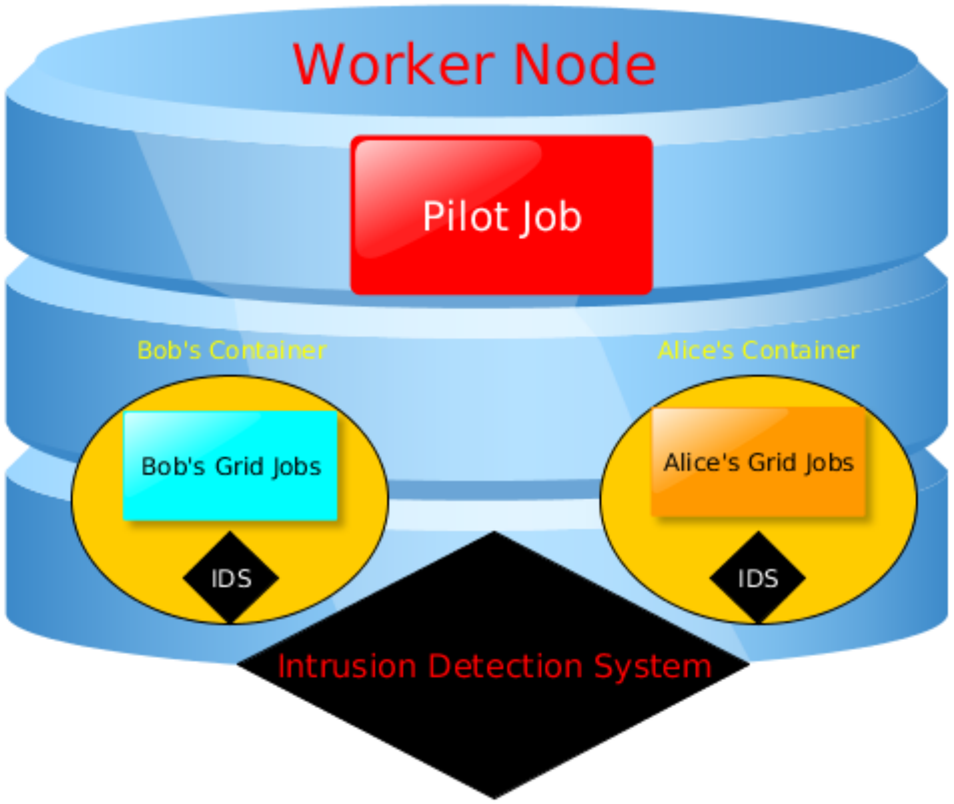
Power utilisation efficiency

Anomaly detection

Security

Intrusion detection

- ALICE example: https://arxiv.org/pdf/1704.06193.pdf
  - Grids face complex security challenges
  - Interesting targets for attackers seeking for huge computational resources, since users can execute arbitrary code in the worker nodes on the Grid sites
  - Even with unbreakable isolation (VMs, containers) the jobs themselves may still do considerable harm
    - Benign users can often break things unintentionally
  - Proposal from ALICE to monitor the jobs themselves using ML techniques
  - Use job and system logs, system call sequence, other common monitoring data.
  - SVMs suggested as a reasonable algorithm choice

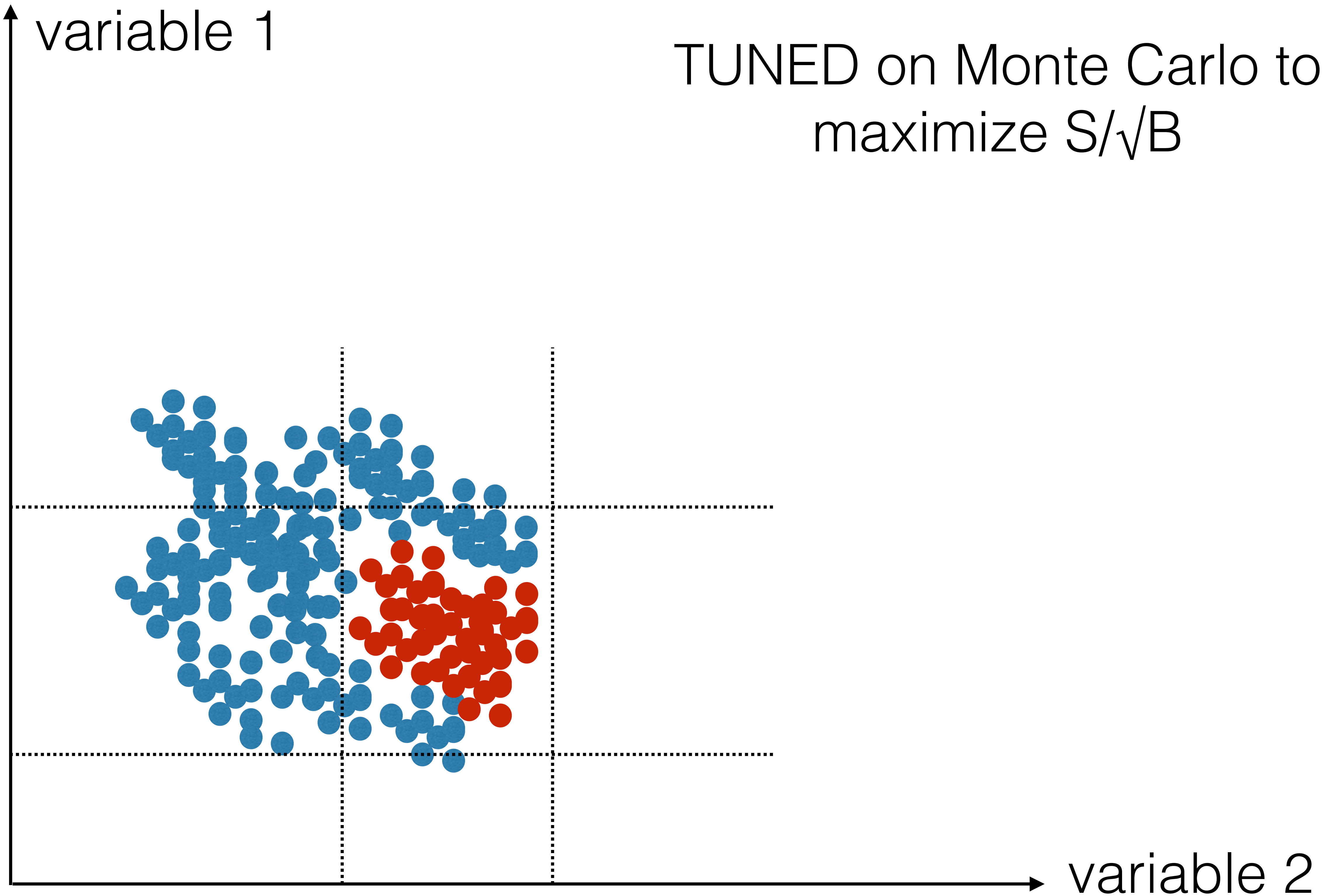


Conclusions

- Machine learning could have a big impact on distributed computing
  - Increasing efficiency of computing resources utilisation
  - Detecting faults
  - Detecting security violations
  - Improvement energy efficiency
- How relevant this will be if we make more use of commercial clouds remains to be seen
- Important to use it when it can help, and not to try to use it when it can't
- This is interesting work: potential to recruit students to work on these topics
  - ML experience becoming essential for many computing-related jobs in industry
- Personal comment: computing is under-represented at HEP meetings on machine learning (IML, ATLAS ML forum…)

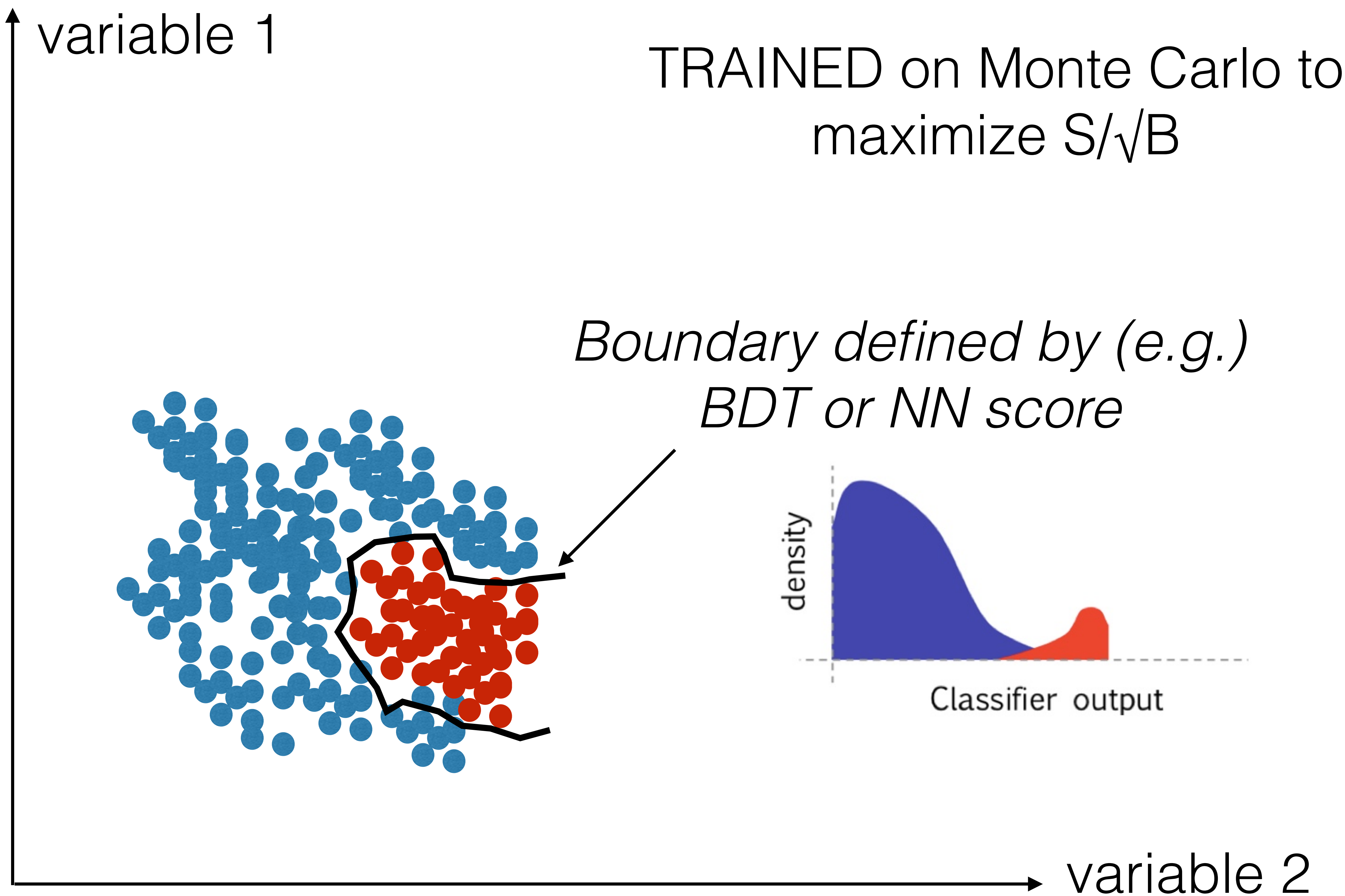
Backup

Classification without machine learning

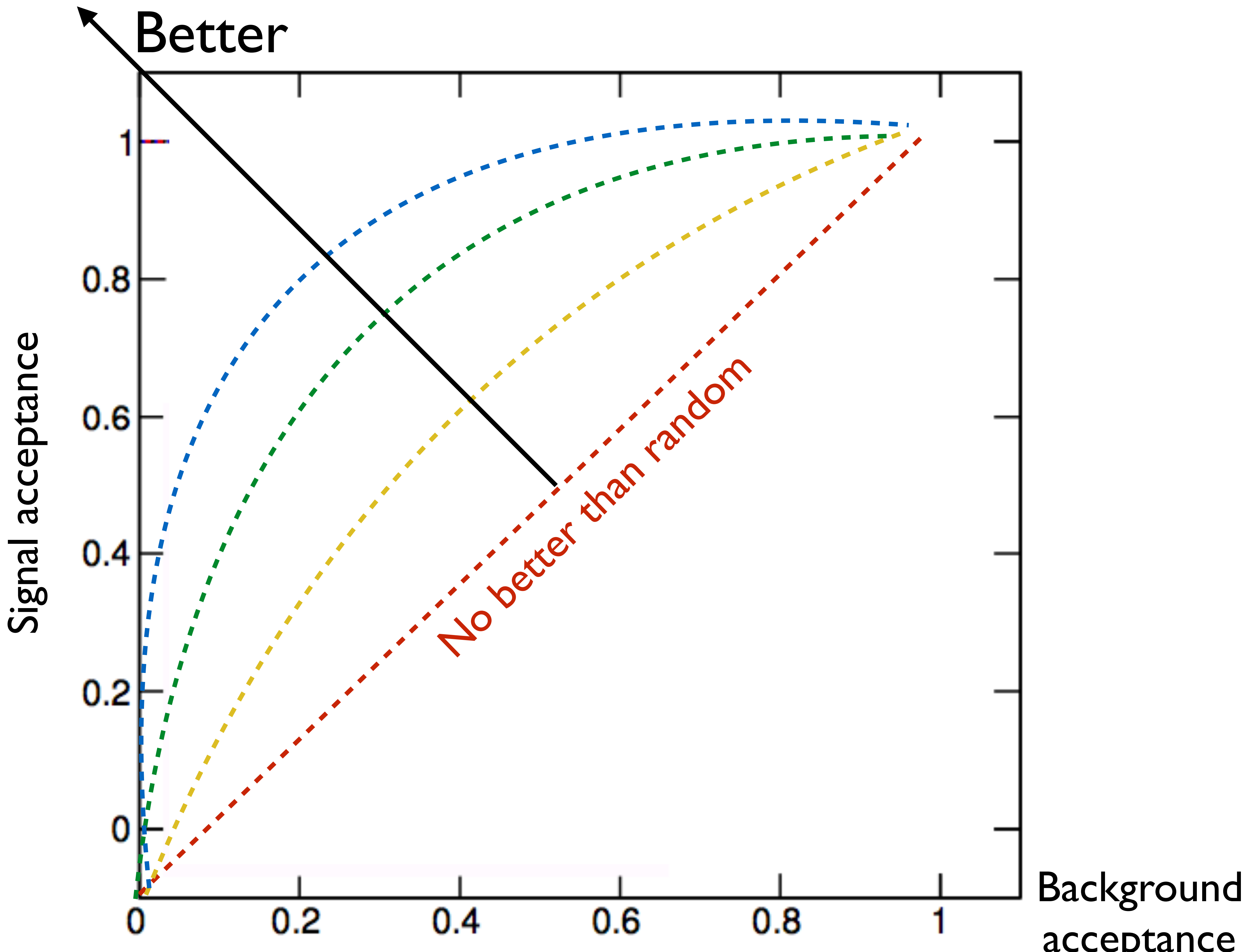


TUNED on Monte Carlo to maximize $S/\sqrt{B}$

Classification with machine learning



Response operator curve (ROC)



Interesting links

- Inter-experiment machine learning working group (IML)
  - https://iml.web.cern.ch
  - Meets regularly, all agendas public
  - ML Activities of the four LHC experiments
- CERN OpenData
  - http://opendata.cern.ch/?ln=en
- Last CHEP conference
  - http://chep2016.org