Yandex

SCHOOL OF DATA ANALYSIS

Yandex intro.

Highlights of Machine Learning for High Energy Physics.

Andrey Ustyuzhanin, Maxim Borisyak, Alexander Panin

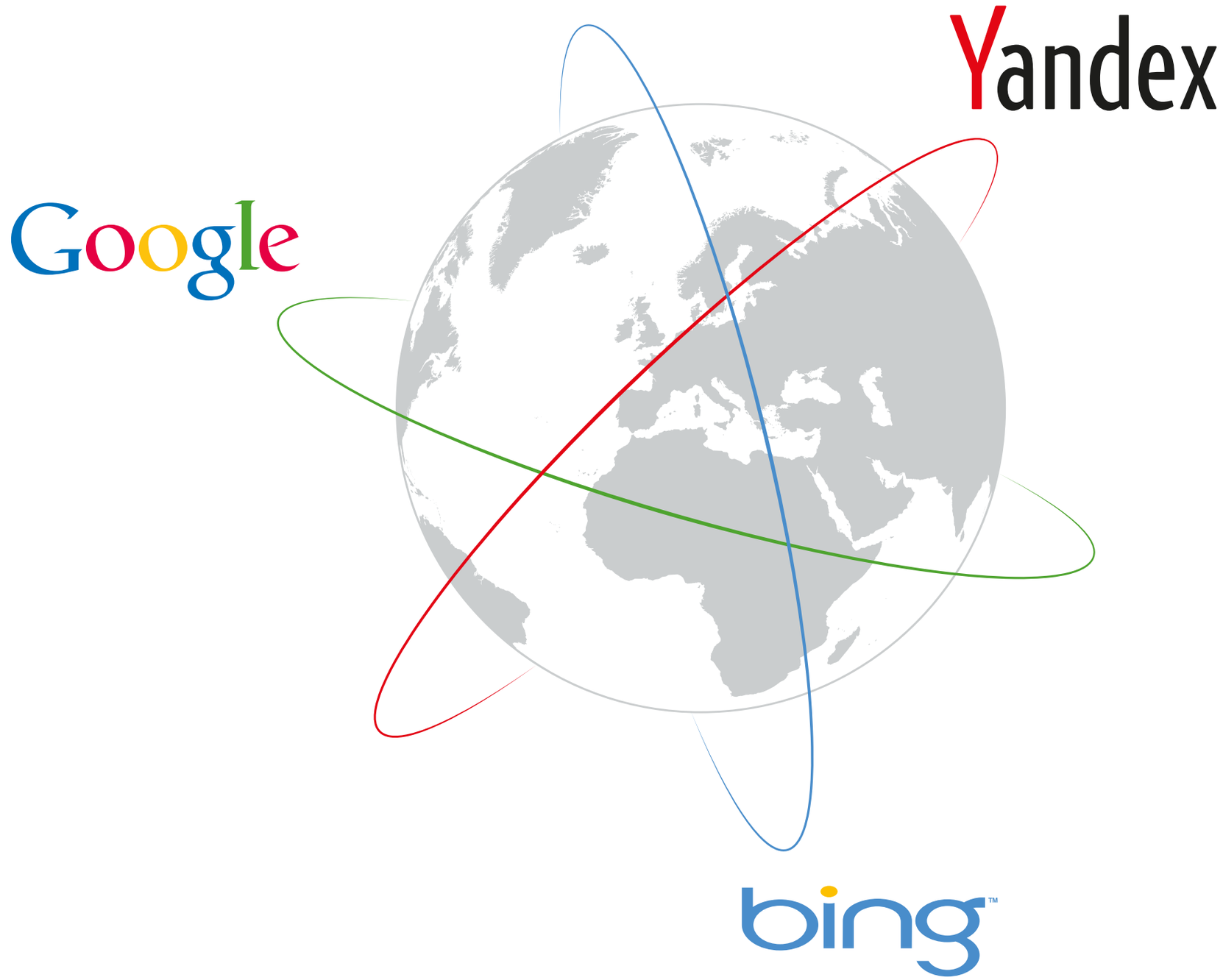
Yandex School of Data Analysis and

Higher School of Economics, Moscow, Russia

Helsinki

01 February 2017

Global Search Engines



1989 Technology launched

2011 IPO on NASDAQ ($9.6B capitalization)

More than 6000 people (3000+ engineers)

Yandex School of Data Analysis

Machine Learning Applications

- Search (ranking, snippets, personalization, etc.)
- Advertising (targeting, click prediction)
- Collaborative filtering (recommendations)
- Traffic jams on maps
- Language translation
- Images, voice, video recognition
- Spam filtering
- News clustering
- Anomaly detection

Yandex School of Data Analysis

- Non-profit
- Solid program for 2 years
- 2 years — 4 sessions
- 1 session — 3+ courses

Main subjects

Algorithms & Data Structures

Discrete Mathematics

Probability & Statistics

Programming Languages (C++ and Python)

Machine Learning

Statistics in ML

Parallel and Distributed Computing

Big Data Storage & Processing

Natural Language Processing

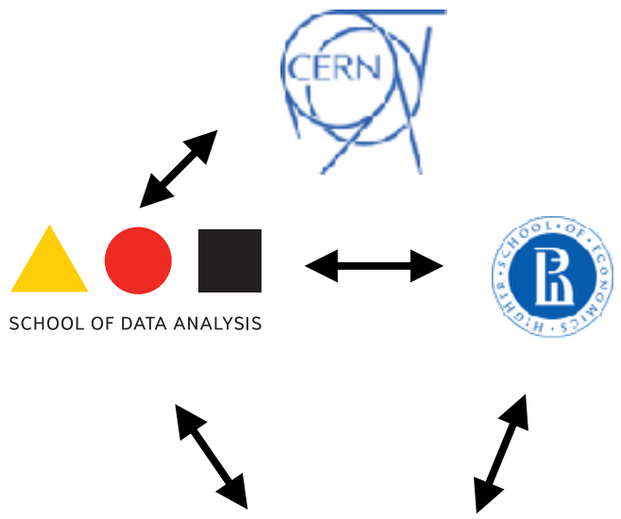
Images & Video processing

Information Retrieval

Deep Learning

Our group

- Group working on data analyses in Natural sciences
  - 2 physicists and 8 computer scientists (out of them 6 students)
- Part of a nonprofit Yandex School of Data Analysis
- Members of the LHCb, SHiP, OPERA collaborations



Yandex

leading search engine
in Russia (and not only)

Aim to apply machine learning to the real scientific world problems

Which tasks?

- Any, that can be formalised as a Machine Learning task
  - good accessible dataset
  - formalisable additional conditions
  - clear rules to select winning solution
- Examples include:
  - Storage/Speed optimisation for triggers
  - Jet and flavour tagging algorithms
  - Brain cognitive studies
  - Ultra-High Energy Cosmic Ray Detection

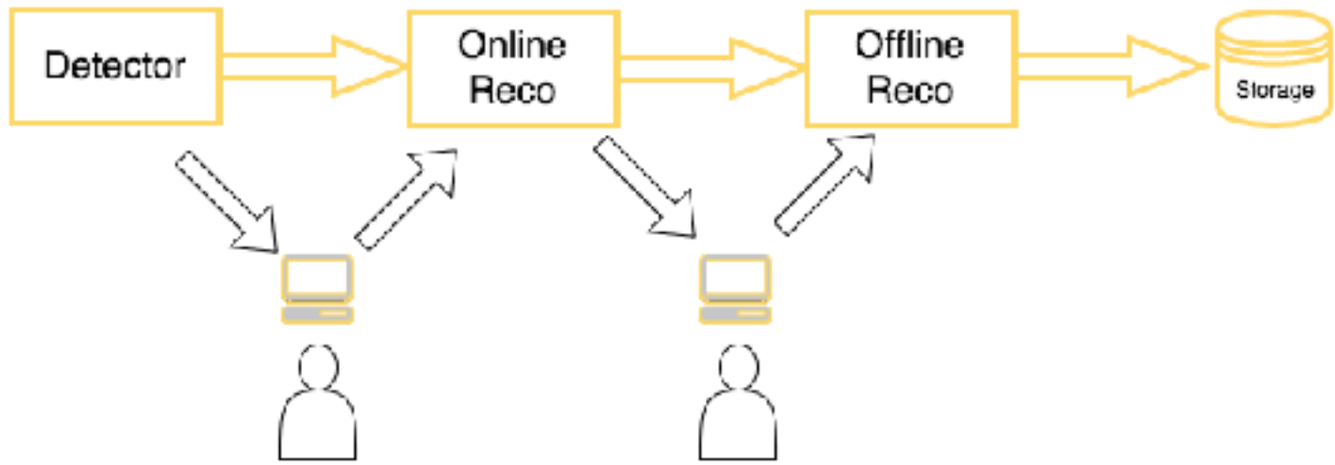
Outline

- Recent examples of ML algorithm developed for an LHC experiment
  - Anomaly Detection
  - High Level Trigger
- Generalised ML algorithms useful for analysis
  - BDT reweighting

ML for Anomaly Detection

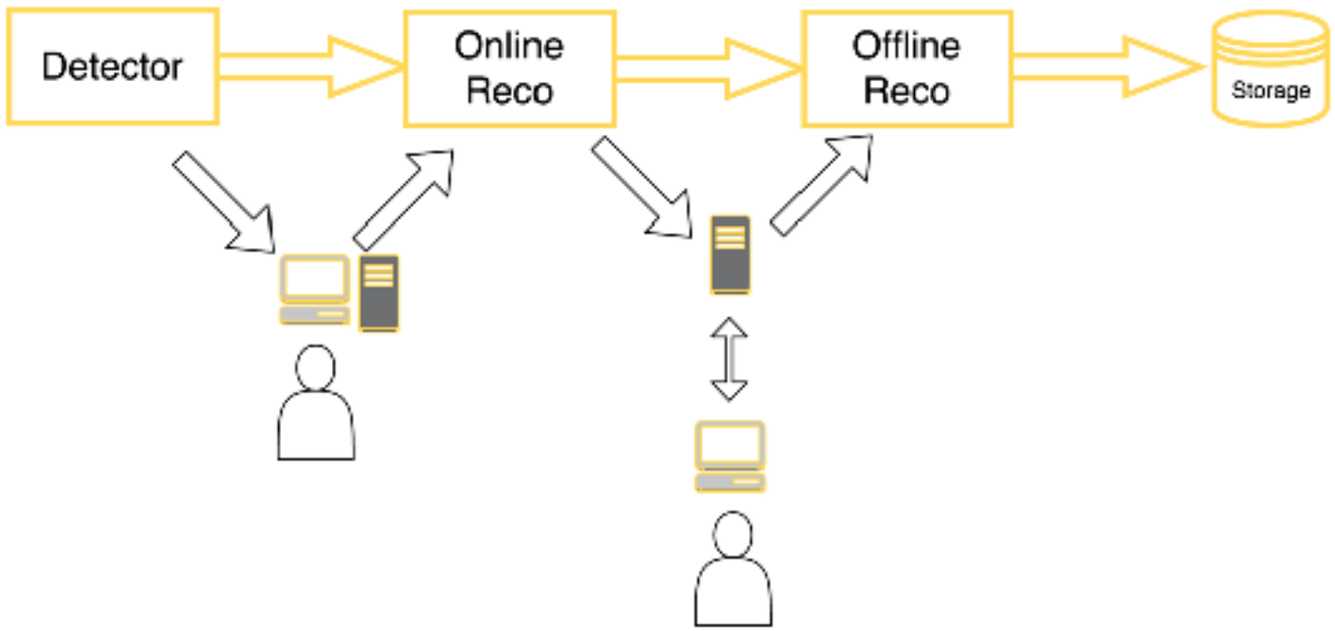
Typical Workflow

- Several people are typically on shifts controlling the flow of data from detector into the storage



Updated Workflow

- The monitoring systems can be updated with:
  - **helper**, a recommendation system for a shifter
  - **solver**, automated decision maker
  - **both**

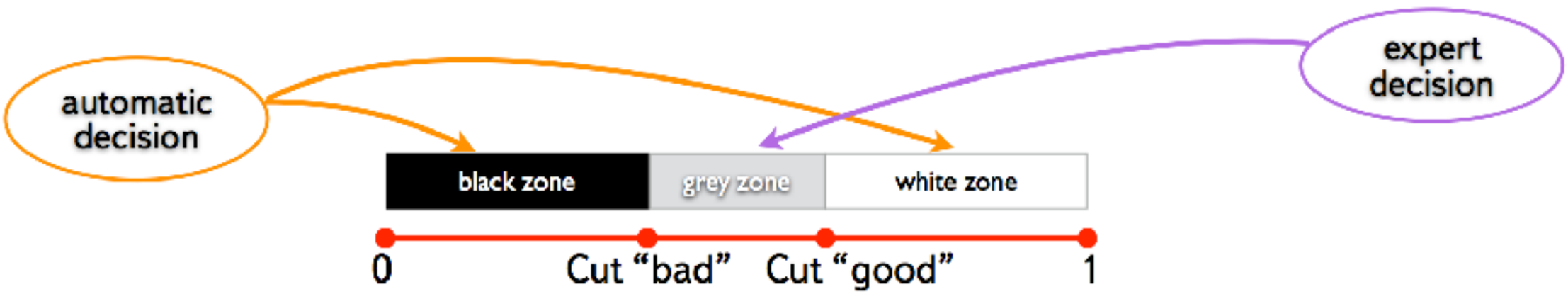


Approaches

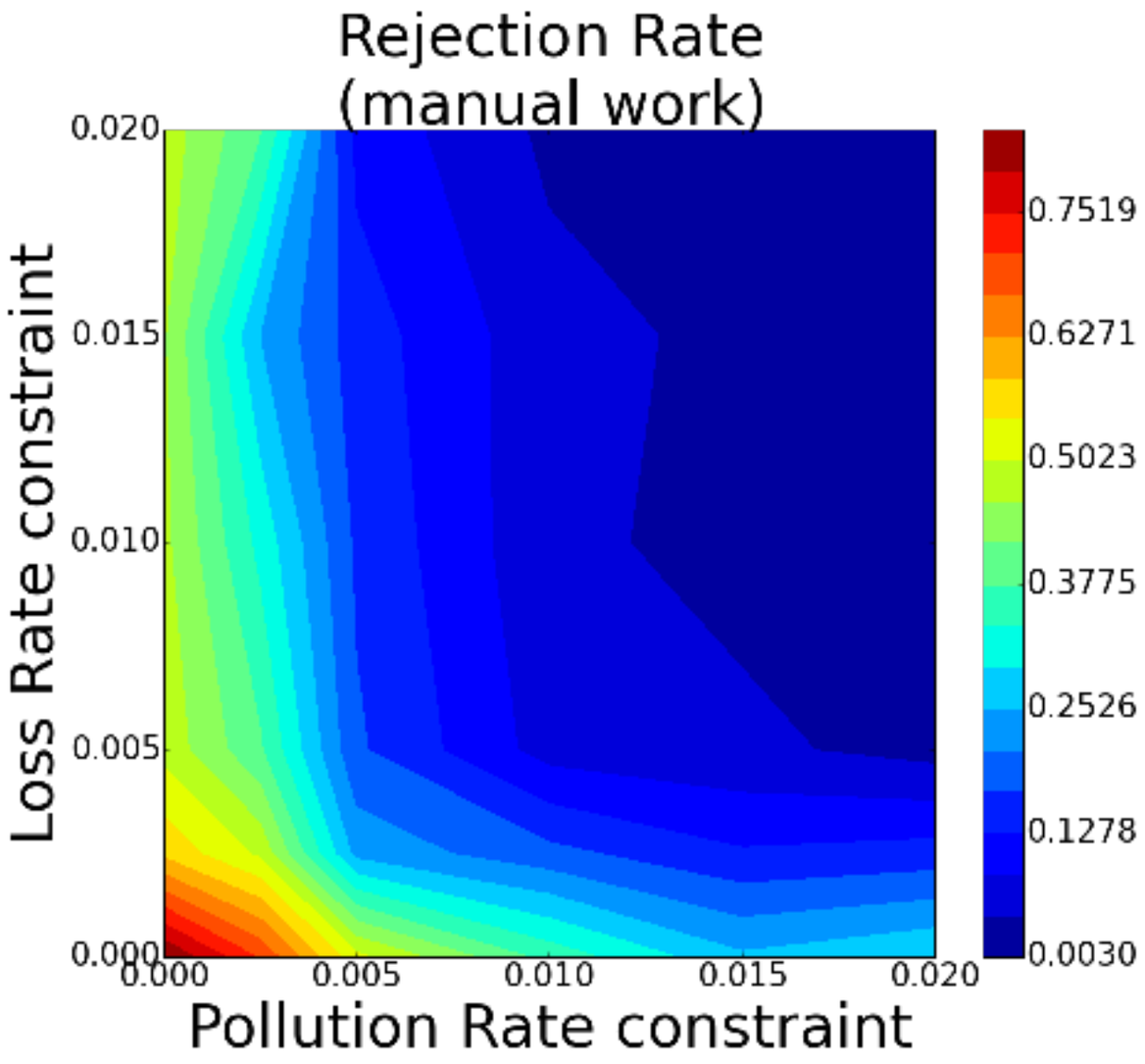
- Two ML approaches are possible in this case:
- Supervised approach
  - uses historical data processed by expert
  - ML algorithm learns the pattern that lead to the experts' decision
  - can prevent experts mistakes
  - problem: impossible to outperform the expert in quality
- Unsupervised approach
  - use time series to catch changes in data behaviour
  - problem: hard to validate

Supervised Learning

- Problem: CMS Data Certification
- Data: CMS 2010B run open data
- Aim: automated classification of LumiSections as “good” or “bad”
- Features: particle flow jets, Calorimeter Jets, Photons, Muons
- The dataset was flagged by experts (3 FTE)



Supervised Learning

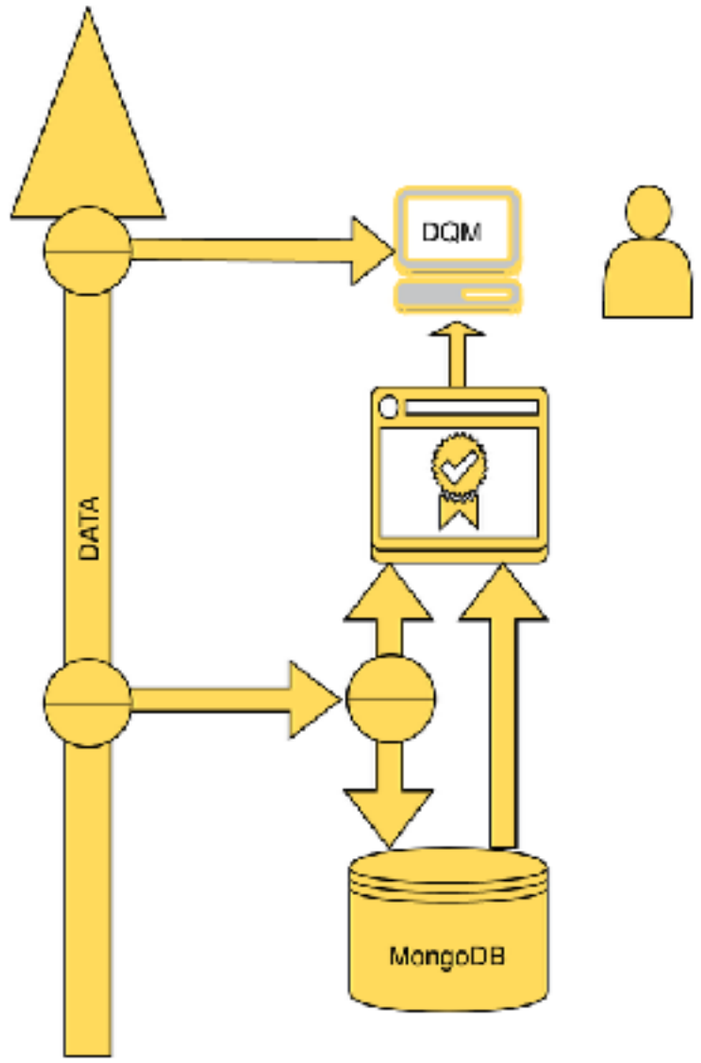


The aim is to minimise the Manual work with low Loss Rate (“good” classified as “bad”) and Pollution Rate (“bad” classified as “good”).

~90% saving on manual work is feasible for Pollution rate at 5%

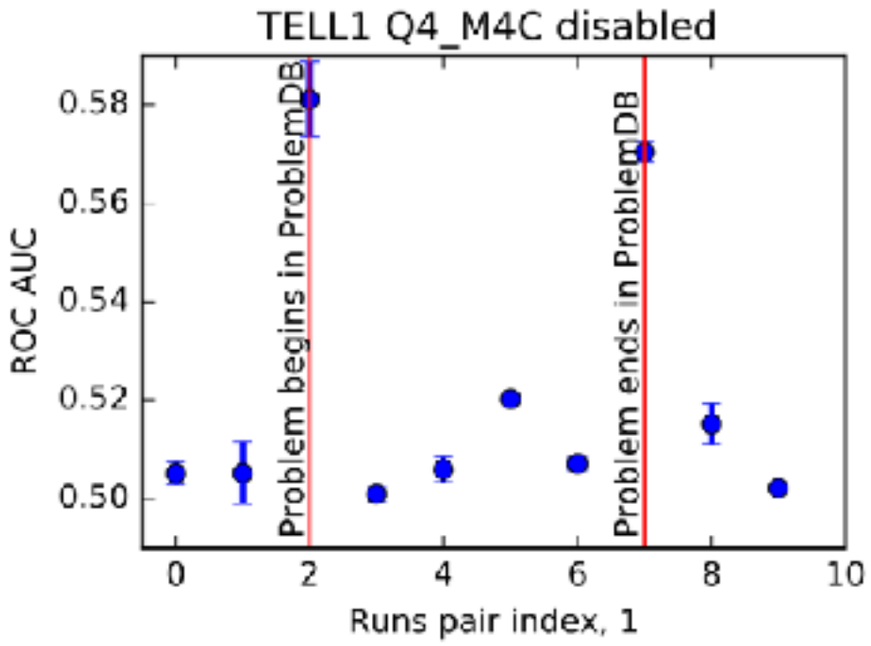
Unsupervised Learning

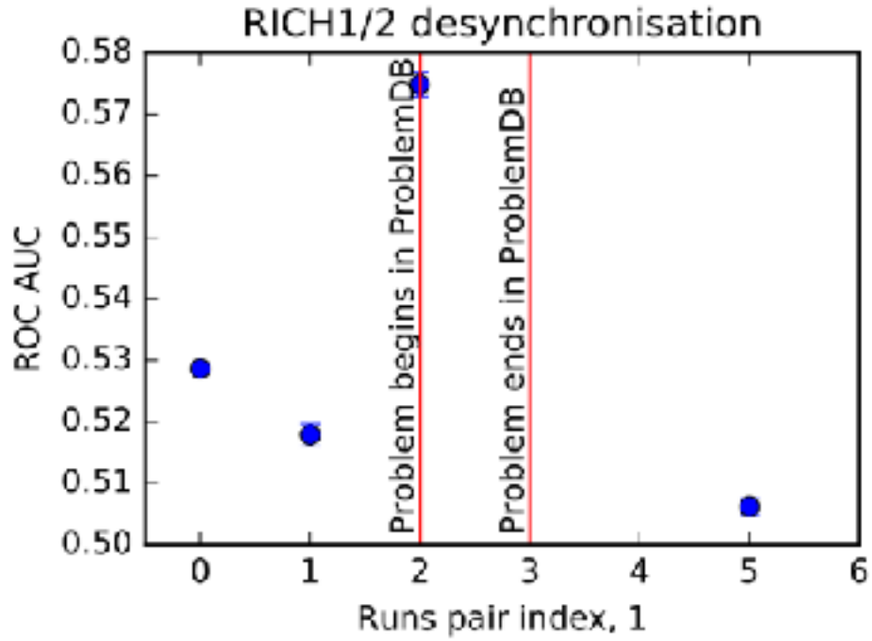
- Problem: LHCb Detector Monitoring
- Data: LHCb trigger streams
- Aim: Identification of problems using previous state of the system
- Features: trigger line decisions, other trigger objects.



Unsupervised Learning

〉 First attempts look promising





〉 work is ongoing

References

〉 http://arxiv.org/abs/1510.00132

〉 https://github.com/yandexdataschool/cms-dqm

〉 F. Ratnikov @ DSHEP https://indico.hep.caltech.edu/indico/conferenceOtherViews.py?confId=102&view=standard

〉 M. Borisyak @ CHEP https://indico.cern.ch/event/505613/contributions/2227973/

ML in Trigger

LHCb topological trigger

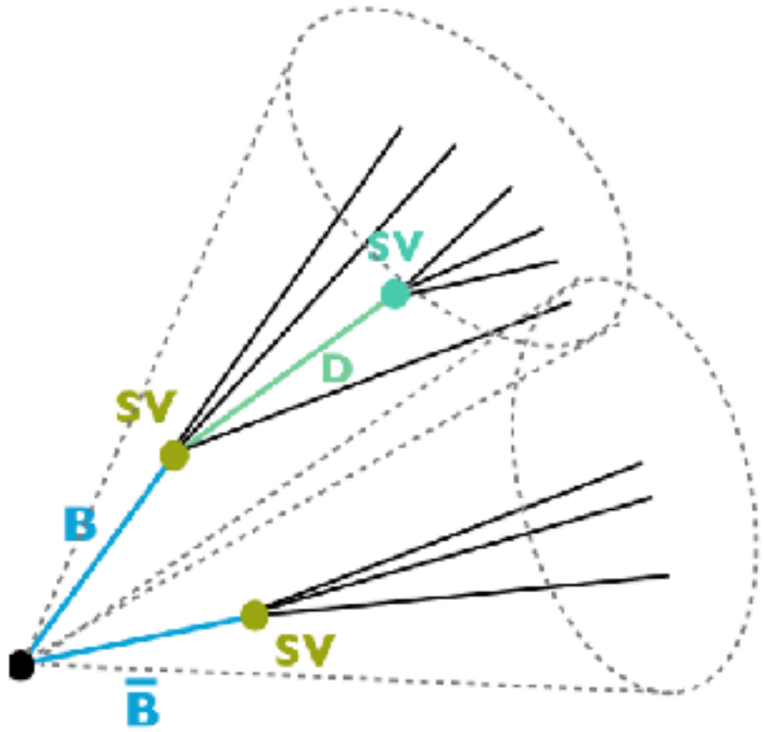
- Generic trigger for decays of beauty and charm hadrons
- Inclusive for any B decay with at least 2 charged daughters including missing particles
- Look for 2, 3, 4 track combinations in a wide mass range

Event

〉 one proton-proton collision

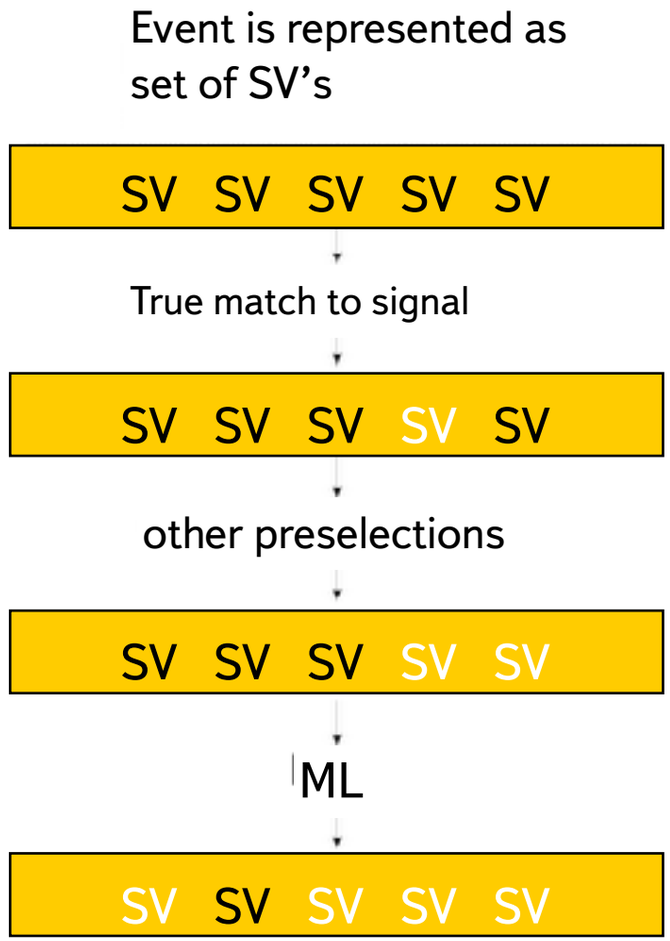
〉 Event consists of:

- tracks (track description)
- secondary vertices (SV description)
- hits data



Data

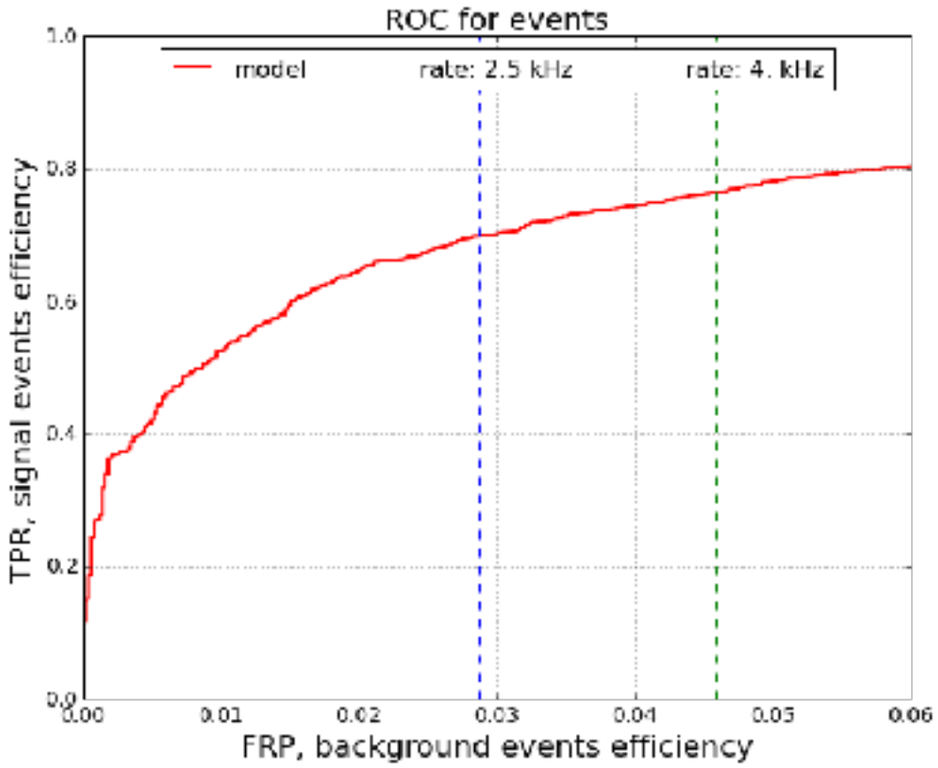
- Monte Carlo samples (used as signal) are simulated with B decays of various topologies
- Generic proton-proton collisions are used as background sample (also includes some signal)
- Most events have many secondary vertices
- Goal is to improve efficiency for each type of signal events with fixed efficiency for background



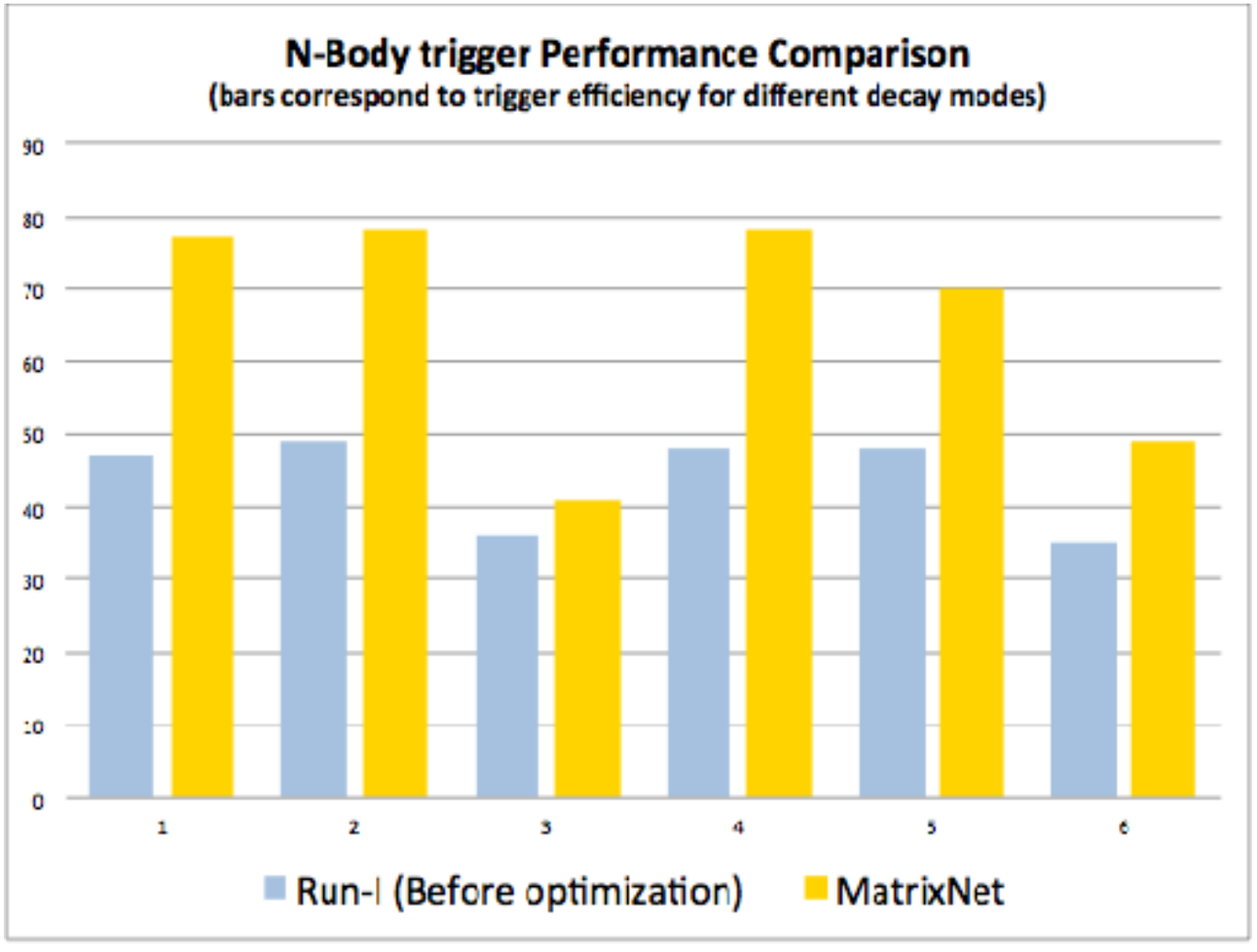
If at least one SV in the event passed all stages, the whole event passes trigger

ROC curve, computed for events

- Output rate = false positive rate (FPR) for events (since background = generic event)
- Optimise true positive rate (TPR) for fixed FPR for events
- Weight signal events in such way that channels have the same amount of events.
- Optimise ROC curve in a small FPR region



Topological trigger results

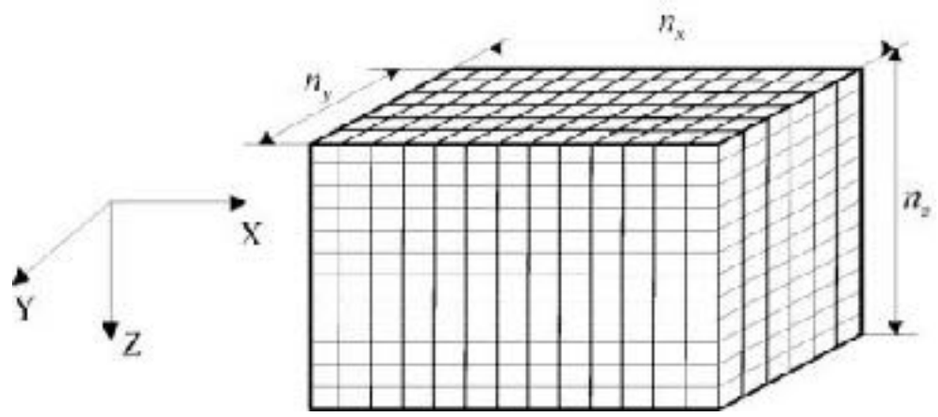


1. $B^0 \to K^*[K^+\pi^-]\mu^+\mu^-$
2. $B^+ \to \pi^+K^-K^+$
3. $B_s^0 \to D_s^-[K^+K^-\pi^-]\mu^+\nu_\mu$
4. $B_s^0 \to \psi(1S)[\mu^+\mu^-]K^+K^-\pi^+\pi^-$
5. $B_s^0 \to D_s^-[K^+K^-\pi^-]\pi^+$
6. $B^0 \to D^+[K^-\pi^+\pi^+]D^-[K^+\pi^-\pi^-]$

https://github.com/yandexdataschool/LHCb-topo-trigger

Online part using Bonsai BDT

- Features hashing using bins before training
- Converting decision trees to n-dimensional table (lookup table)
- Table size is limited in RAM (1Gb), thus count of bins for each features should be small (5 bins for each of 12 features)
- Discretisation reduces the quality



References

〉 https://github.com/yandexdataschool/LHCb-topo-trigger

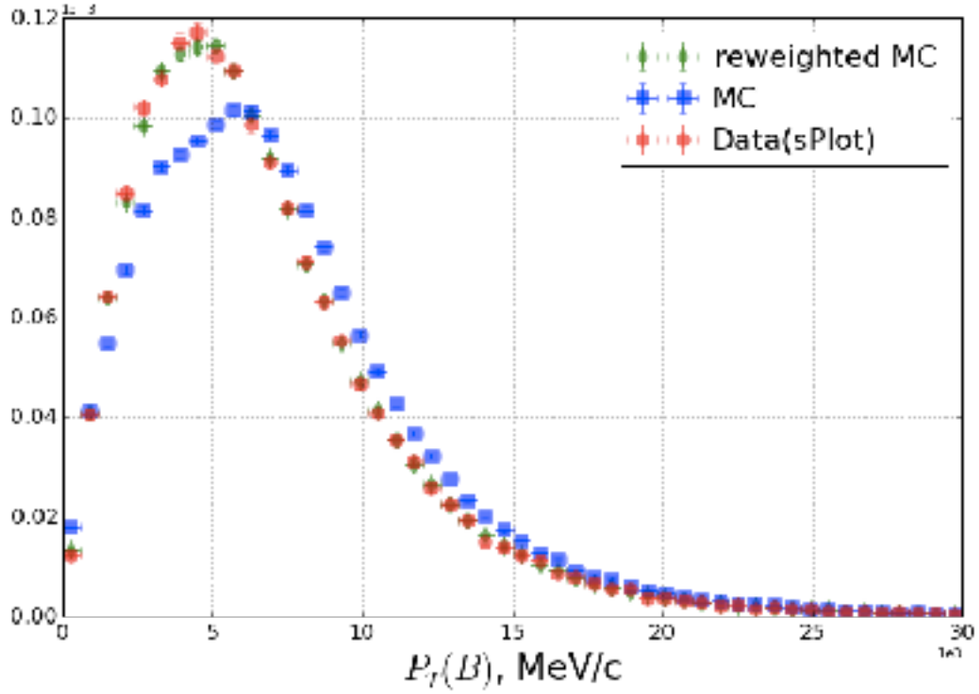
〉 https://cdsweb.cern.ch/record/1384380/files/LHCb-PUB-2011-016.pdf

〉 http://arxiv.org/abs/1510.00572

Reweighting problem in HEP

Distributions reweighting

- Reweighting in HEP is used to minimize the difference between RD and MC samples
- The goal of reweighting: assign weights to MC s.t. MC and RD distributions coincide
- Known process is used, for which RD can be obtained (MC samples are also available)
- MC distribution is original, RD distribution is target



Histogram reweighting

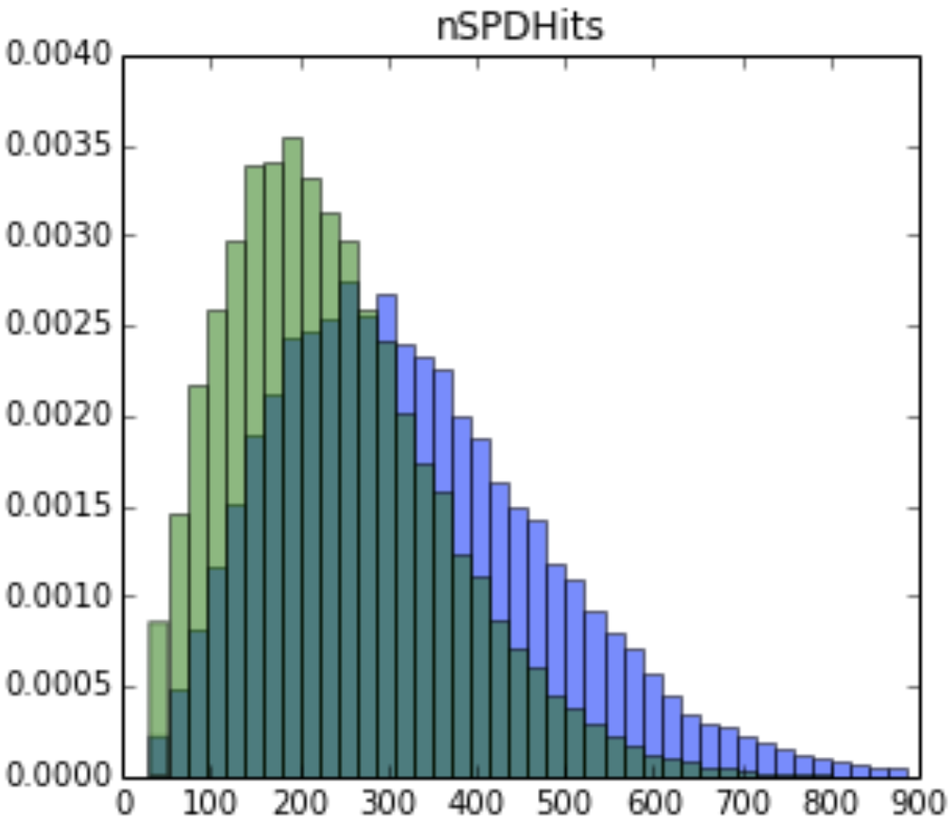
- variable is split into bins
- in each bin the MC weight is multiplied by:

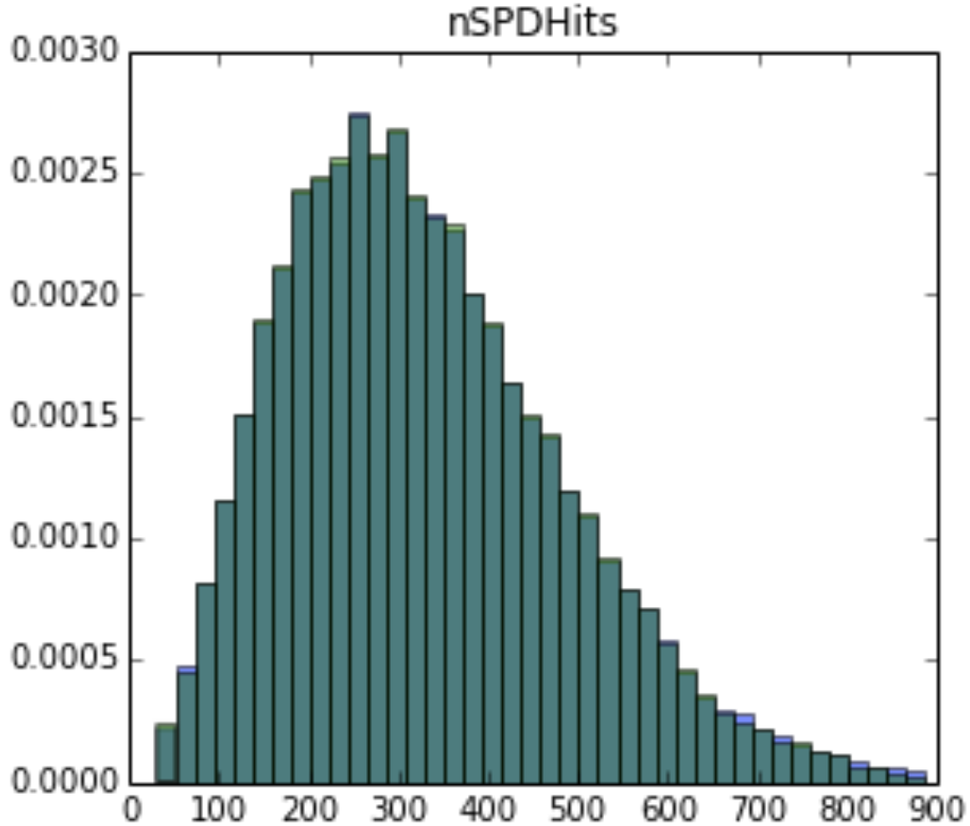
$$\text{multiplier}_{\text{bin}} = \frac{w_{\text{bin, target}}}{w_{\text{bin, original}}}$$

$w_{\text{bin, target}}$, $w_{\text{bin, original}}$ - total weights of events in a bin for target and original distributions

1. simple and fast
2. number of variables is very limited by statistics (typically only one, two)
3. **BUT:** reweighting in one variable may bring disagreement in others
4. which variable is preferable for reweighting?

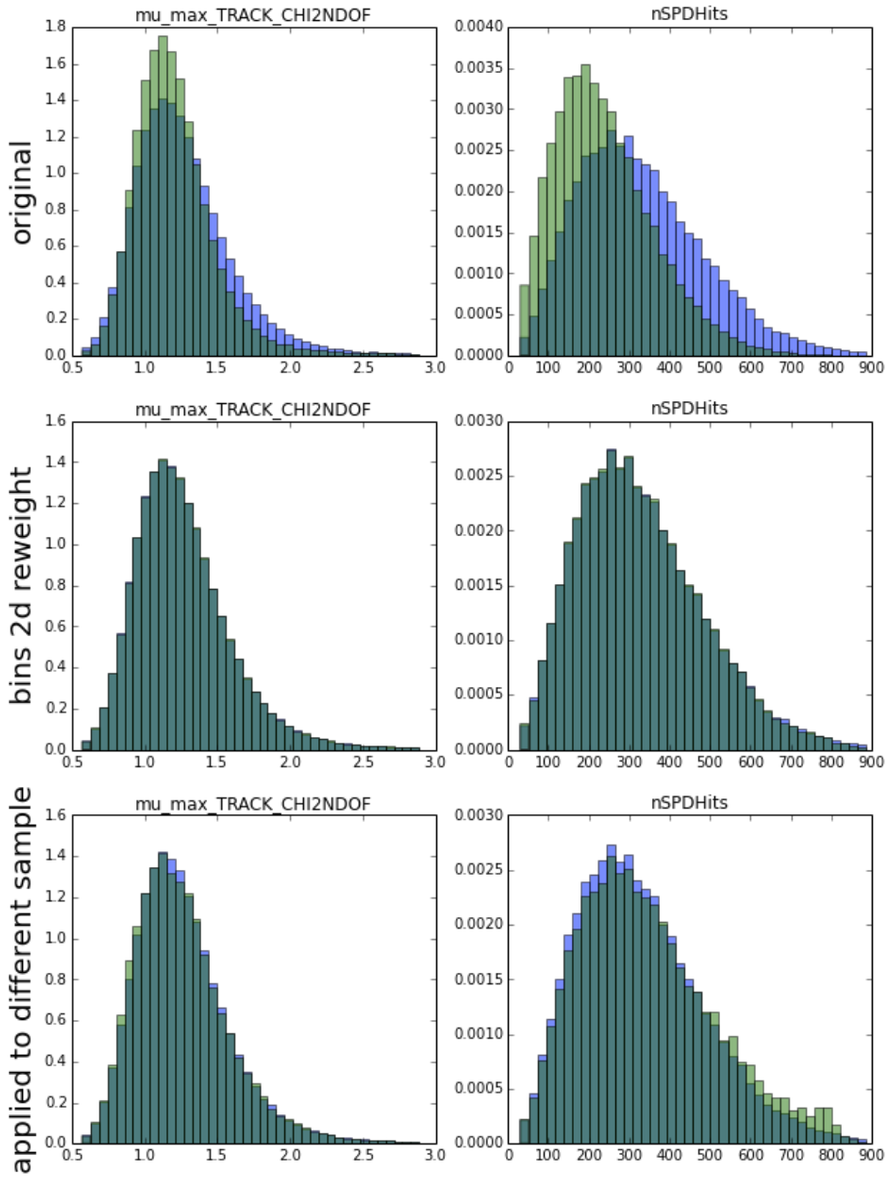
Typical approach: example





Typical approach: example

- Problems arise when there are too few events in a bin
- This can be detected on a **holdout**
- Issues:
    1. few bins - rule is rough
    2. many bins - rule is not reliable



Decision tree for reweighting

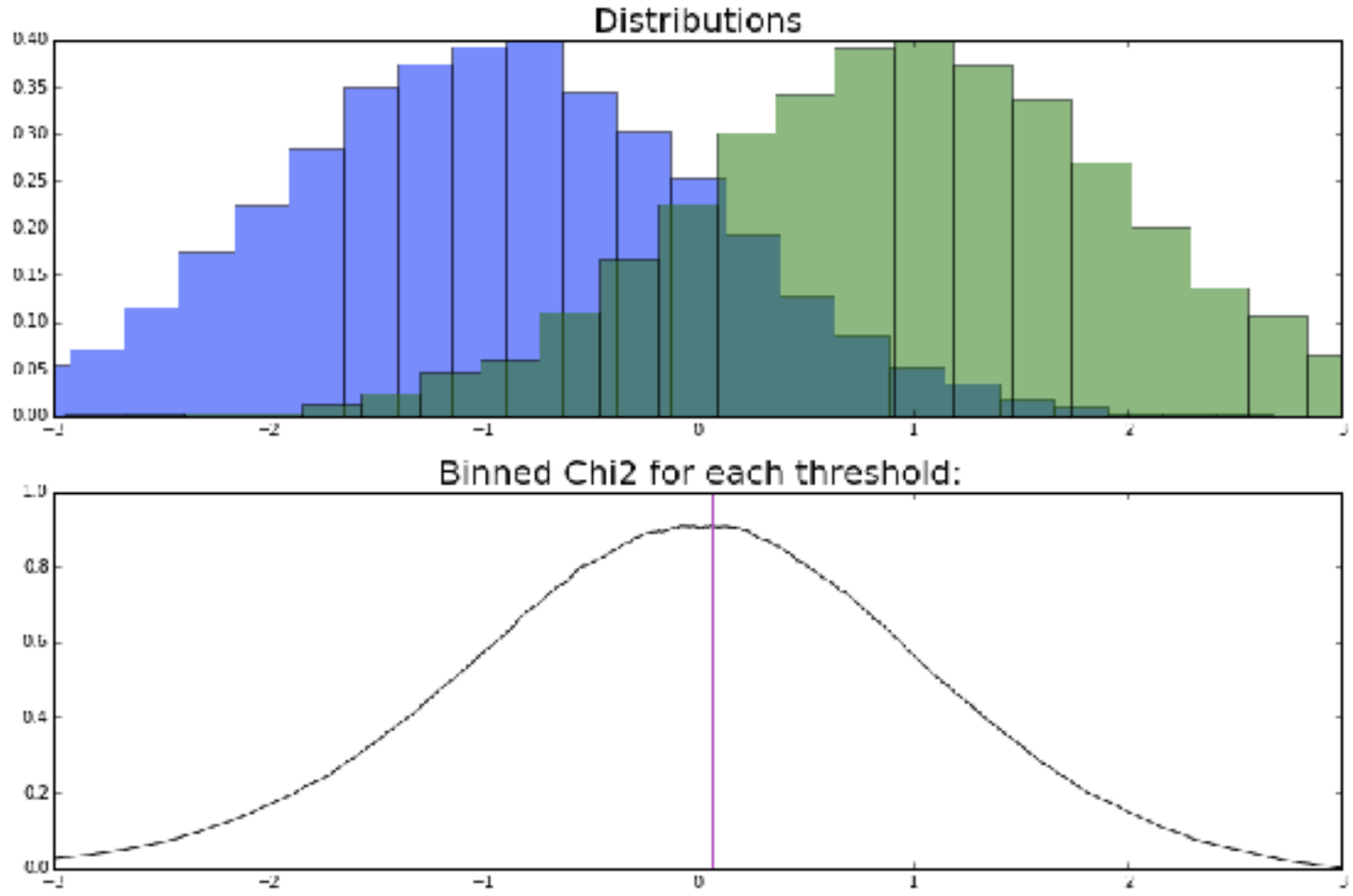
Write ML algorithm to solve directly reweighting problem:

- Tree splits the space of variables with orthogonal cuts (each tree leaf is a region, or bin)
- Find regions with the highest difference between original and target distribution

Spitting criteria

Finding regions with high difference between original and target distribution by maximising symmetrized

$$\chi^2 = \sum_{leaf} \frac{(w_{\text{leaf, original}} - w_{\text{leaf, target}})^2}{w_{\text{leaf, original}} + w_{\text{leaf, target}}}$$



A tree leaf may be considered as ‘a bin’;

$w_{\text{leaf, original}}, w_{\text{leaf, target}}$ - total weights of events in a leaf for target and original distributions.

BDT reweighter

Many times repeat the following steps:

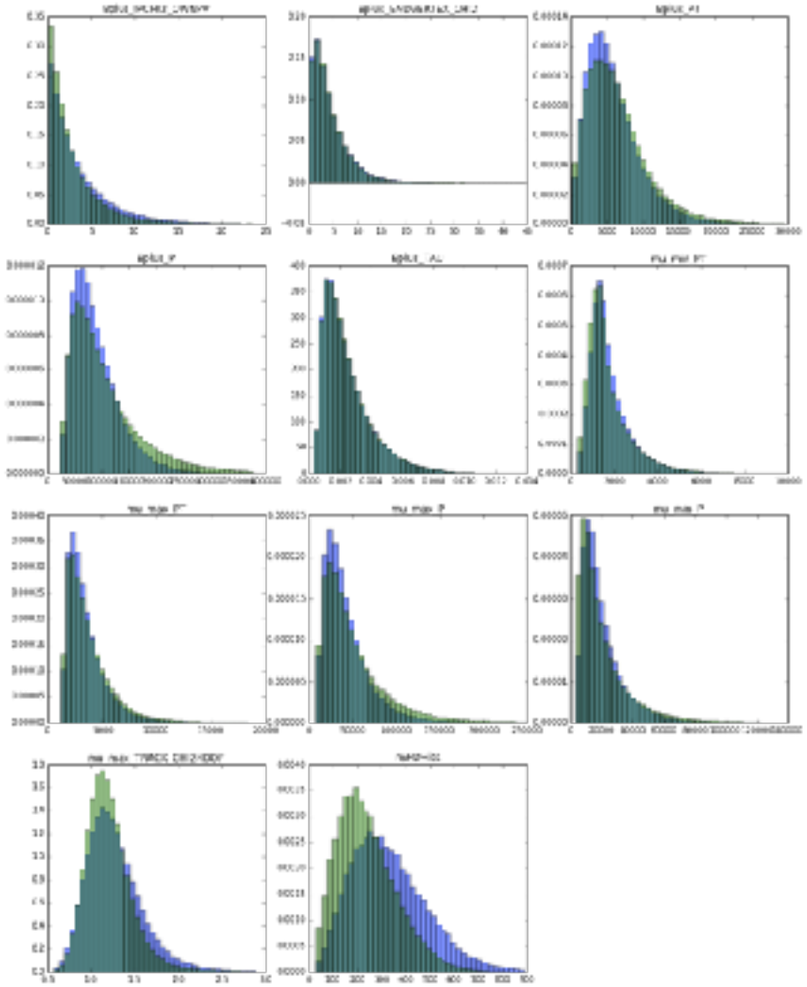
- build a shallow tree to maximise symmetrized χ^2
- compute predictions in leaves:

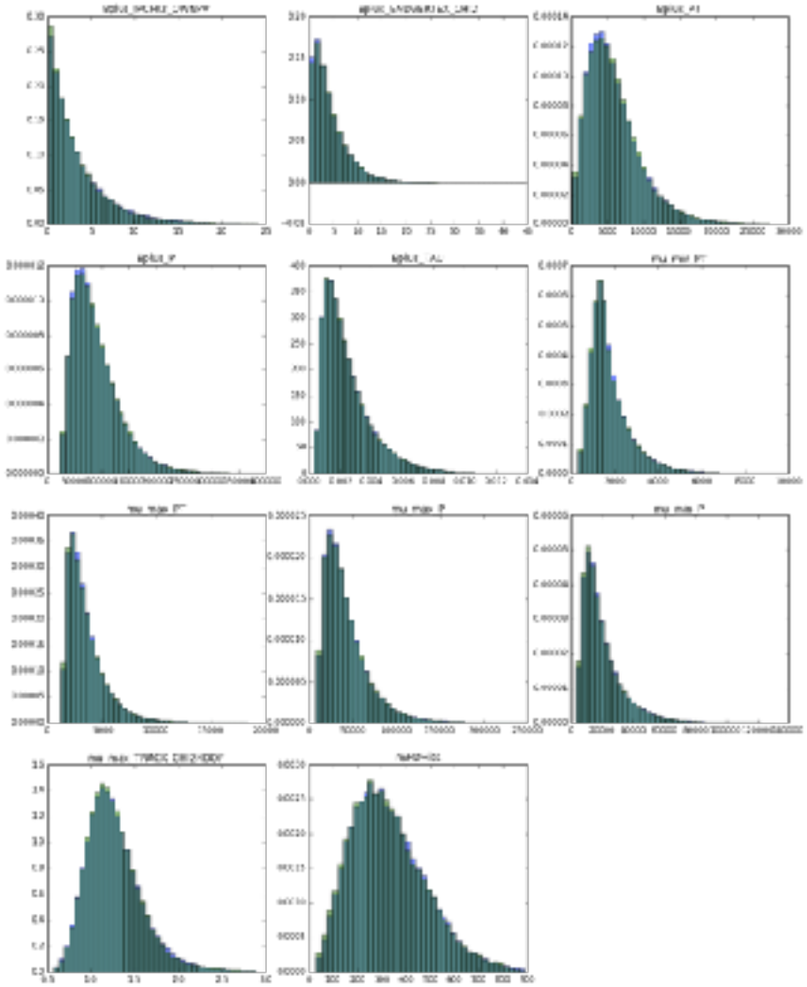
$$\text{leaf\_pred} = \log \frac{w_{\text{leaf, target}}}{w_{\text{leaf, original}}}$$

- reweight distributions:

$$w = \begin{cases} w, & \text{if event from target (RD) distribution} \\ w \cdot e^{\text{pred}}, & \text{if event from original (MC) distribution} \end{cases}$$

BDT reweighter DEMO





NB: 1D histograms might be misleading, please use multidimensional comparison, for example, using ROC AUCs of a classifier.

Kolmogorov-Smirnov distance for 1d

Bins reweighter uses only 2 last variables (60 × 60 bins); BDT reweighter uses all variables

	KS original	KS bins reweight	KS GB reweight
Feature			
Bplus_IPCHI2_OWNPV	0.080	0.064	0.003
Bplus_ENDVERTEX_CHI2	0.010	0.019	0.002
Bplus_PT	0.060	0.069	0.004
Bplus_P	0.111	0.115	0.005
Bplus_TAU	0.005	0.005	0.003
mu_min_PT	0.062	0.061	0.004
mu_max_PT	0.048	0.056	0.003
mu_max_P	0.093	0.098	0.004
mu_min_P	0.084	0.085	0.004
mu_max_TRACK_CHI2NDOF	0.097	0.006	0.005
nSPDHits	0.249	0.009	0.005

hep_ml library

```
from hep_ml.reweight import GBReweighter
gb = GBReweighter()
gb.fit(mc_data, real_data, target_weight=real_data_sweights)
gb.predict_weights(mc_other_channel)
```

Being a variation of GBDT, BDT reweighter is able to calculate feature importances. Two features used in reweighting with bins are indeed the most important.

	importance
feature	
mu_max_TRACK_CHI2NDOF	0.240272
nSPDHits	0.209090
Bplus_P	0.122314
mu_min_P	0.115245
Bplus_PT	0.080641
Bplus_IPCHI2_OWNPV	0.068209
mu_max_P	0.060518
mu_max_PT	0.037863
mu_min_PT	0.037761
Bplus_ENDVERTEX_CHI2	0.026598
Bplus_TAU	0.001489

https://github.com/yandexdataschool/hep_ml

Reweighter Summary

BDT reweighter

- uses each time few large bins (construction is done intellectually)
- is able to handle many variables
- requires less data (for the similar performance).

References

〉 https://arxiv.org/abs/1608.05806

〉 http://arogozhnikov.github.io/2015/10/09/gradient-boosted-reweighter.html

〉 https://arogozhnikov.github.io/hep_ml/

Other topics include

- Particle Identification for LHCb, CERN (DNN and DT)
- Pattern recognition for Velo LHCb, CERN (Artificial Retina Algorithm)
- B-Tagging and jet tagging for LHCb (New model)
- Tracking and ghost tracks removal at SHiP, CERN (Artificial Retina Algorithm)
- Tracking for Comet, KEK (Gradient Boosting + Inverse Hough Transform)
- Triggers for CRAYFIS (Cosmic RAYs Found In Smartphone)
- e-shower identification for OPERA photo-emulsion data
- gradient boosting for uniform predictions

Summary

In High-Energy Physics

- good accessible dataset
- formalisable additional conditions
- clear rules to select winning solution

Provides interesting problems for Machine Learning

Which than can be generalised into common approaches

Thanks for attention

Andrey Ustyuzhanin
head of joint research projects
anaderi@yandex-team.ru

Special thanks

- To people from Yandex group who were involved in preparing the slides:
  - Denis Derkach
  - Tatiana Likhomanenko
  - Fedor Ratnikov
  - Alex Rogozhnikov
  - Mikhail Hushchyn