SCHOOL OF DATA ANALYSIS

Yandex Data Factory

Data analysis and reproducibility tools

for HEP and beyond

Tim Head[3], Andrey Ustyuzhanin[1,2] on behalf of Yandex

2016-05-03, HSF workshop, Orsay

[1]Yandex School of Data Analysis, [2]Imperial College London, [3]Wild Tree Technologies

Contents

Yandex overview (est 1997)

- Web search
- Image search
- Speech recognition
- Car traffic prediction
- Mail and spam filtering
- Natural language translation
- Market (shopwindow for internet shops)
- Yandex Data Factory (https://yandexdatafactory.com)
- Yandex School of Data Analysis
    - (YSDA - full member of LHCb since Dec'15)

YSDA & LHCb projects (since 2011)

- Strong physics background since April 2015
- Topological trigger with MatrixNet formula optimized for speed

YSDA & LHCb projects (since 2011)

- Strong physics background since April 2015
- Topological trigger with MatrixNet formula optimized for speed
    - improved up to 60% signal efficiency in Run-II vs Run-I
    - used in 60% of LHCb papers

YSDA & LHCb projects (since 2011)

- Strong physics background since April 2015
- Topological trigger with MatrixNet formula optimized for speed
    - improved up to 60% signal efficiency in Run-II vs Run-I
    - used in 60% of LHCb papers
- Grid Data Storage optimization

YSDA & LHCb projects (since 2011)

- Strong physics background since April 2015
- Topological trigger with MatrixNet formula optimized for speed
    - improved up to 60% signal efficiency in Run-II vs Run-I
    - used in 60% of LHCb papers
- Grid Data Storage optimization
    - up to 40% disk storage saving

YSDA & LHCb projects (since 2011)

- Strong physics background since April 2015
- Topological trigger with MatrixNet formula optimized for speed
  - improved up to 60% signal efficiency in Run-II vs Run-I
  - used in 60% of LHCb papers
- Grid Data Storage optimization
  - up to 40% disk storage saving
- Event metadata indexing
  - run-event number access
  - used for optimization of streams

YSDA & LHCb projects (since 2011)

- Strong physics background since April 2015
- Topological trigger with MatrixNet formula optimized for speed
  - improved up to 60% signal efficiency in Run-II vs Run-I
  - used in 60% of LHCb papers
- Grid Data Storage optimization
  - up to 40% disk storage saving
- Event metadata indexing
  - run-event number access
  - used for optimization of streams
- LHCb data quality monitoring and anomaly prediction

YSDA & LHCb projects (since 2011)

- Strong physics background since April 2015
- Topological trigger with MatrixNet formula optimized for speed
  - improved up to 60% signal efficiency in Run-II vs Run-I
  - used in 60% of LHCb papers
- Grid Data Storage optimization
  - up to 40% disk storage saving
- Event metadata indexing
  - run-event number access
  - used for optimization of streams
- LHCb data quality monitoring and anomaly prediction
- Data & physics analysis
  - $B_s \to 2\mu, B_s \to 4\mu, \tau \to 3\mu$

Data Analysis Tools

- Reproducible Experiment Platform (https://github.com/yandex/rep)
- hep_ml (https://github.com/arogozhnikov/hep_ml)
    - reweighting
    - uniform boosting
- MatrixNet-as-a-Service
    - MatrixNet is a custom Yandex implementation of GBDT
- everware - service for managing Jupyter-based research environments using git and Docker (http://everware.xyz)

Reproducible Experiment Platform

- Python-based (numpy, pandas, ...), Jupyter-friendly
- Unified scikit-learn-like API to many ML packages(Sklearn, XGBoost, uBoost, TMVA, Theanets, ...)

Reproducible Experiment Platform

- Python-based (numpy, pandas, ...), Jupyter-friendly
- Unified scikit-learn-like API to many ML packages(Sklearn, XGBoost, uBoost, TMVA, Theanets, ...)
- Meta-algorithms pipelines («REP-Lego»)

Reproducible Experiment Platform

- Python-based (numpy, pandas, ...), Jupyter-friendly
- Unified scikit-learn-like API to many ML packages(Sklearn, XGBoost, uBoost, TMVA, Theanets, ...)
- Meta-algorithms pipelines («REP-Lego»)
- Configurable interactive reporting & visualization to ensure model quality (e.g. check for overfitting)

Reproducible Experiment Platform

- Python-based (numpy, pandas, ...), Jupyter-friendly
- Unified scikit-learn-like API to many ML packages(Sklearn, XGBoost, uBoost, TMVA, Theanets, ...)
- Meta-algorithms pipelines («REP-Lego»)
- Configurable interactive reporting & visualization to ensure model quality (e.g. check for overfitting)
- Pluggable quality metrics

Reproducible Experiment Platform

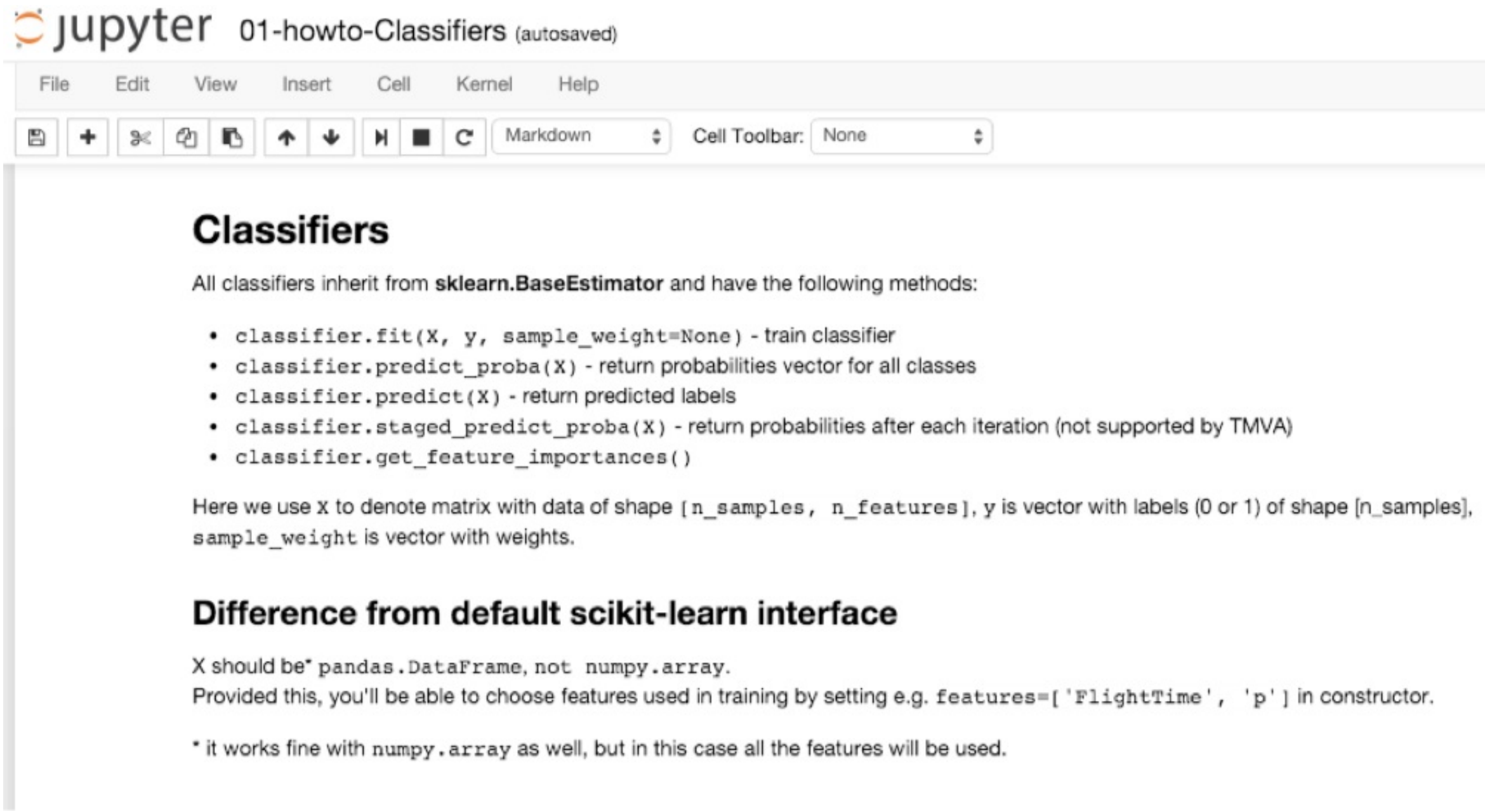
- Python-based (numpy, pandas, ...), Jupyter-friendly
- Unified scikit-learn-like API to many ML packages(Sklearn, XGBoost, uBoost, TMVA, Theanets, ...)
- Meta-algorithms pipelines («REP-Lego»)
- Configurable interactive reporting & visualization to ensure model quality (e.g. check for overfitting)
- Pluggable quality metrics
- Paralleled training of classifiers & grid search (IPython parallel)

Reproducible Experiment Platform

- Python-based (numpy, pandas, ...), Jupyter-friendly
- Unified scikit-learn-like API to many ML packages(Sklearn, XGBoost, uBoost, TMVA, Theanets, ...)
- Meta-algorithms pipelines («REP-Lego»)
- Configurable interactive reporting & visualization to ensure model quality (e.g. check for overfitting)
- Pluggable quality metrics
- Paralleled training of classifiers & grid search (IPython parallel)
- Open-source, Apache 2.0: https://github.com/yandex/rep
- Well-documented, supported by Yandex, http://yandex.github.io/rep/

Unified classifer method interface

https://github.com/yandex/rep/blob/master/howto/01-howto-Classifiers.ipynb



Meta Machine Learning (REP-Lego)

- Factory
- Folding, https://github.com/yandex/rep/blob/master/howto/04-howto-folding.ipynb
- Predictive model optimization (grid search)
  - GridOptimalSearch
  - Folding Scorer
  - Various Optimization algorithms
- Parameter optimizer interface
- Stacking

REP: Reporting

- Draws set of reports upon model training completion. Supported libraries:
  - Matplotlib
  - ROOT
  - Bokeh (Javascript)
  - plot.ly (going to be deprecated due to limitations)
- Extensible!

https://github.com/yandex/rep/blob/master/howto/02-howto-Factory.ipynb

Running REP

- Locally (virtualenv, conda)
  - https://github.com/yandex/rep/wiki/Installing-manually

Running REP

- Locally (virtualenv, conda)
  - https://github.com/yandex/rep/wiki/Installing-manually
- Locally (docker, kitematic)
  - https://github.com/yandex/rep/wiki/

Running REP

- Locally (virtualenv, conda)
  - https://github.com/yandex/rep/wiki/Installing-manually
- Locally (docker, kitematic)
  - https://github.com/yandex/rep/wiki/
- Openstack(CERN) or any cloud provider
  - https://github.com/yandex/rep/wiki/Install-REP-at-openstack

Running REP

- Locally (virtualenv, conda)
  - https://github.com/yandex/rep/wiki/Installing-manually
- Locally (docker, kitematic)
  - https://github.com/yandex/rep/wiki/
- Openstack(CERN) or any cloud provider
  - https://github.com/yandex/rep/wiki/Install-REP-at-openstack
- Tutorial
  - https://github.com/yandex/rep-tutorial
  - run me @everware or from PDF: this link
  - https://github.com/yandex/rep-deployment

HEP ML package

ML-inspired tools for HEP

- UGBoost, http://bit.ly/uBoost
  - training classifier uncorrelated with given feature (mass)
- GBReweighting, http://bit.ly/GBReweight
  - finding event weights to make distributions of two samples match each other

Everware. Sharing Research

Developed in close collaboration with LHCb researchers (Tim Head, Igor Babuschkin, et al: https://github.com/everware)

- Jupyterhub-based
- Docker-empowered
- github-backed

Supported by Mozilla Science Lab, Yandex

How it works

- user pastes link to git repo to everware only input field
- github repository has to have proper `Dockerfile`
- everware clones repository, creates image according to `Dockerfile`
- everware runs image and puts repository inside of it
- user access the running container by browser
- user can fork/push the repository from within everware

Beta-testing program http://everware.xyz (will grant access upon registering)

Everware evaluation

Pros

- can support almost any software environment configuration

Everware evaluation

Pros

- can support almost any software environment configuration
- 0-effort to spawn new environment

Everware evaluation

Pros

- can support almost any software environment configuration
- 0-effort to spawn new environment
- allows for 'bring own resources' computation model

Everware evaluation

Pros

- can support almost any software environment configuration
- 0-effort to spawn new environment
- allows for 'bring own resources' computation model

Cons (restrictions)

- works well for Jupyter-enabled configurations
- no good model to access restricted data

Everware use-cases

- Sharing research/learning from others
- Outreach
- Hackathons
- Training (onsite, remote)

Everware examples

- https://github.com/everware/everware-dimuon-example -- mass of J/psi
- https://github.com/arogozhnikov/GW150914 -- Gravitational Waves study
- https://github.com/yandex/rep-tutorial -- REP tutorial
- https://github.com/lhcb/opendata-project -- CP violation demo

Everware infrastructure needs

- container registry
- identification & authorization service
    - authorized data access

Discussion. Containerization

Requires

- user training
- bits of infrastructure
- tools
- standard images to start from (HSF?)

Discussion. Containerization

Requires

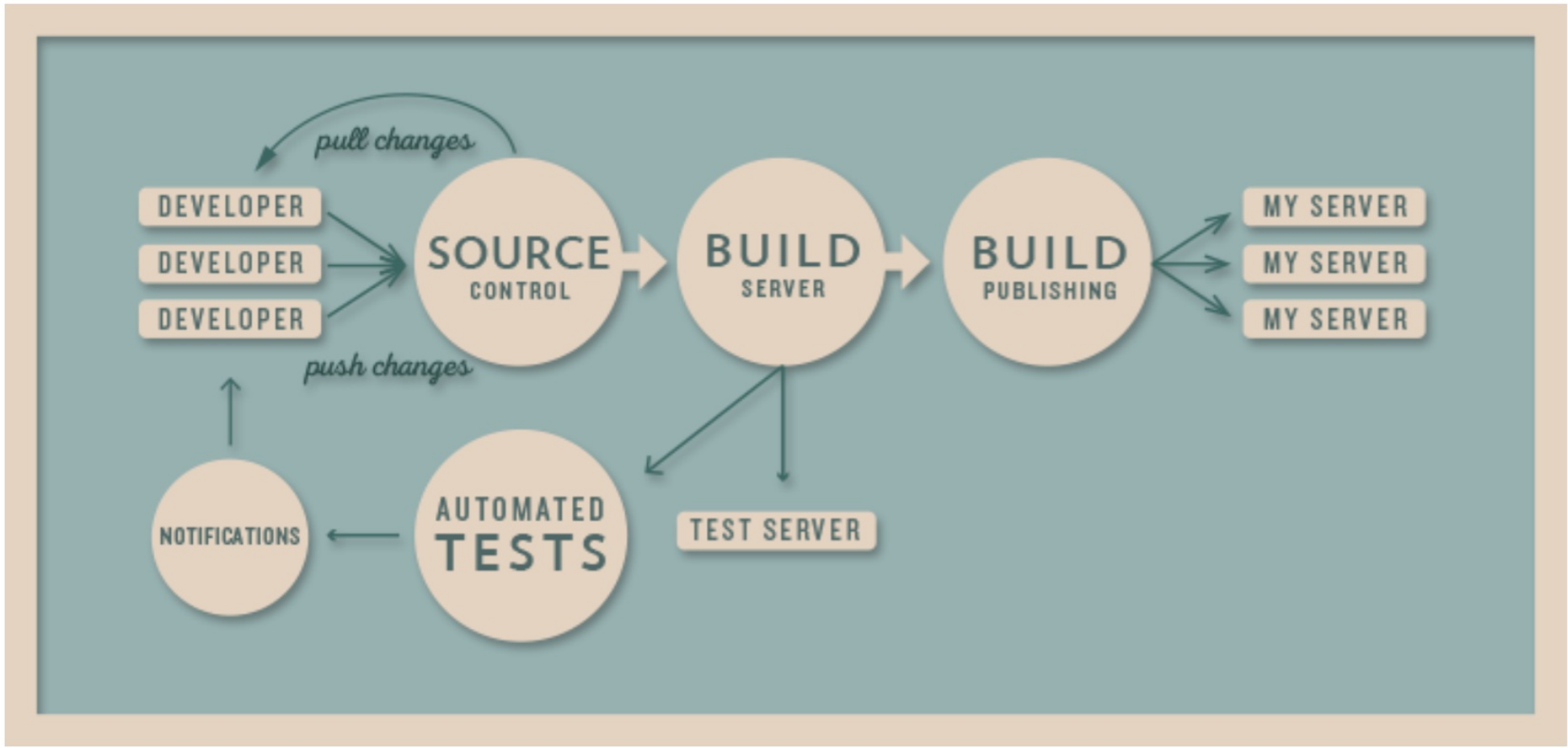
- user training
- bits of infrastructure
- tools
- standard images to start from (HSF?)

Simplifies

- reproducibility
- maintainability
- preservation

Discussion. Software workflows

Continuous Integration (CI)



Andrey.Ustyuzhanin@cern.ch

Discussion. Collaborative workflows

Being discussed and developed at LHCb within Analysis Preservation Group (Sebastian Neubert, Silvia Amerio)

- Collaboration requires pieces of infrastructure
  - testing
  - integration
  - publishing

Discussion. Collaborative workflows

Being discussed and developed at LHCb within Analysis Preservation Group (Sebastian Neubert, Silvia Amerio)

- Collaboration requires pieces of infrastructure
  - testing
  - integration
  - publishing
- From software that control user memory to cloud managment and service orchestration systems
  - MatrixNet as a service
  - AzureML
  - Terraforming of clouds and oppurtunistic resources

Conclusion

- REP - building & verify complex training & optimization schemes
- ML HEP package - bundle of useful ML tricks
- everware - running other's research with a click

Open topics

- collaborative infrastructure missing pieces
- service orchestration
- embracing environment managment

Yandex team is willing to collaborate on those topics

Thank you!

Slideshow created using remark