Common framework for ML analysis in ALICE

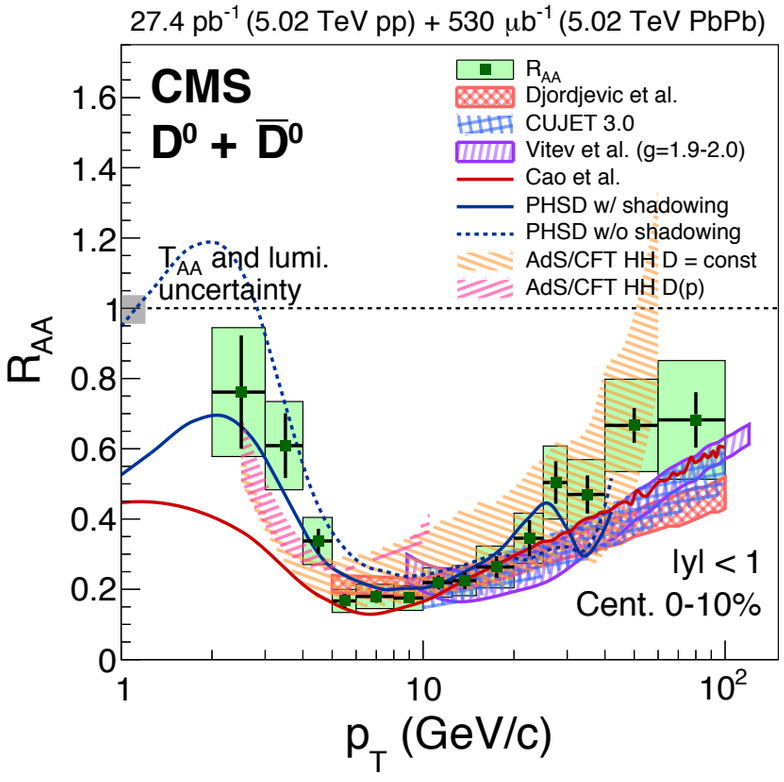
04 December 2018
CERN workshop on Machine Learning for QA

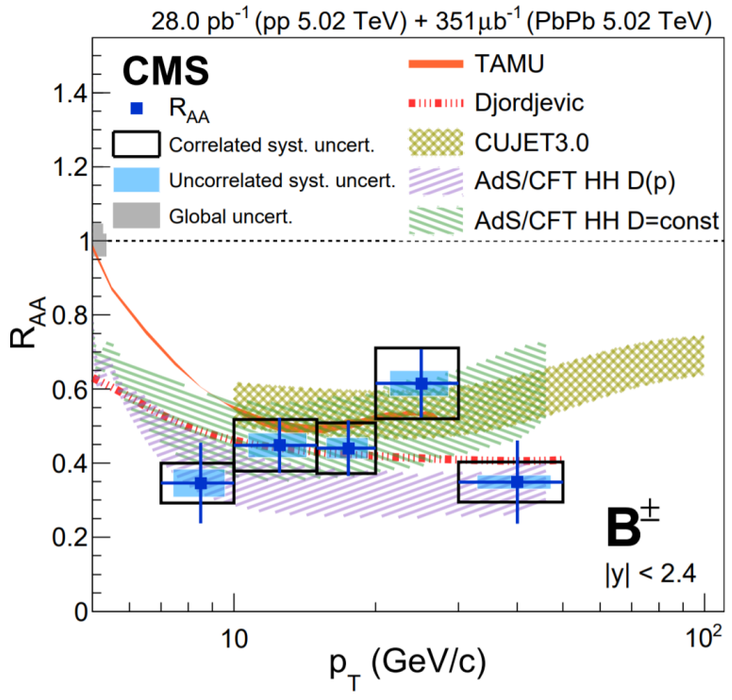
Gian Michele Innocenti (CERN)

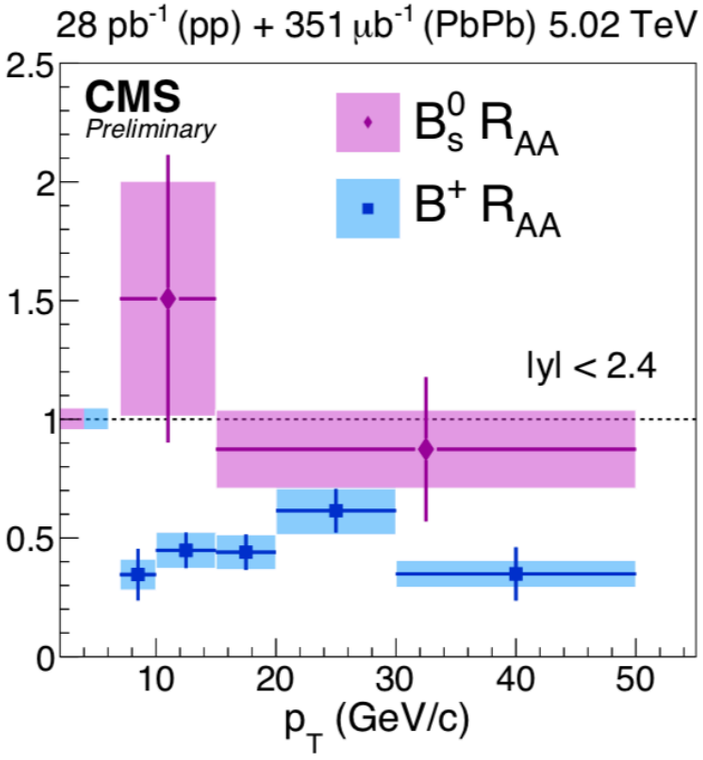
Overview of the talk

- Some ideas for ALICE:
  - An overview of the ML common framework
  - HF selection
  - Jet tagging (gluon vs quark, b/c tagging)
  - PID identification
  - Nuclei/hyper-nuclei studies
  - Data/MC reweighing

Some HI analysis already done with ML







Many of the analyses I performed in CMS were already using BDT to improve the statistical of the signal at low p_T

- Extraction of the low p_T B^+ and B_s signal would not be possible at all without ML

Our project for heavy-ions in ALICE: basic ideas and overview of the ML framework

Overview of the ALICE project

Prerequisite: A new ML software based on the most recent packages like Scikit, XGBoost and TensorFlow (python software with interface with ROOT framework)

Analysis developments:

- use traditional models like BDT to improve the precision of HI analysis especially at very low p_T (HF, b-jets, hyper-nuclei, PID)

- Explore advantages of deep learning to improve performances in cases where low level features can be used (e.g. tracks or calorimeter clusters)

- Develop new methods for MC-data reweighing to produce more reliable MC simulations in view of high precision Run3 analysis

Reconstruction/calibration:

- ML and deep learning for improving online calibration and corrections (golden case: correction of TPC distortions)

- Fast MC production

- DQM and data quality monitoring

Overview of the ALICE project

Prerequisite: A new ML software based on the most recent packages like Scikit, XGBoost and TensorFlow (python software with interface with ROOT framework)

Analysis developments:

- use traditional models like BDT to improve the precision of HI analysis especially at very low p_T (HF, b-jets, hyper-nuclei, PID)

- Explore advantages of deep learning to improve performances in cases where low level features can be used (e.g. tracks or calorimeter clusters)

- Develop new methods for MC-data reweighing to produce more reliable MC simulations in view of high precision Run3 analysis

Reconstruction/calibration:

- ML and deep learning for improving online calibration and corrections (golden case: correction of TPC distortions)

- Fast MC production

- DQM and data quality monitoring

A new common framework for ML

Proving flexible tool to perform Machine Learning analysis. It includes:

- **Common Ntuplizer for TTree creation that can run on the Grid using LEGO trains**
  (effort led by Andrea Festanti, important help from Markus/Jan for the LEGO part)

- **ROOT to Pandas DataFrame conversion:**
  - convert the root TTree of MC and Data into Pandas data frames
  - create training samples mixing MC and data
  - create testing and training samples

- **Training/Testing and common validation routines with Scikit/TensorFlow:**
  - implement most common ML algorithms and Deep Neural network for classification using SciKit and TensorFlow
  - Automatic validation with cross score validation, confusion matrix, learning curves, ROC, etc et

- **Testing on large samples for analysis and new TTree creation:**
  - new decision flag is added to the data frame
  - a new TTree is created including flags and probabilities of all the ML algorithm
  - Possibility of exporting the model in C++ for running testing on the Grid

What is included?

Several algorithms are already available:

- SVM
- Boosted Decision Trees and AdaBoost
- Random Forests
- Logistic regression
- Keras Deep Neural Networks
  - Sequential Network
  - Multi-Layer-Perceptron

Ready to perform:

1) Signal/Background classification for:

- D_s, D^0, Λ_c , B^+, Hypertritium..
- TPC/TOF PID
- jet tagging with high-level variables

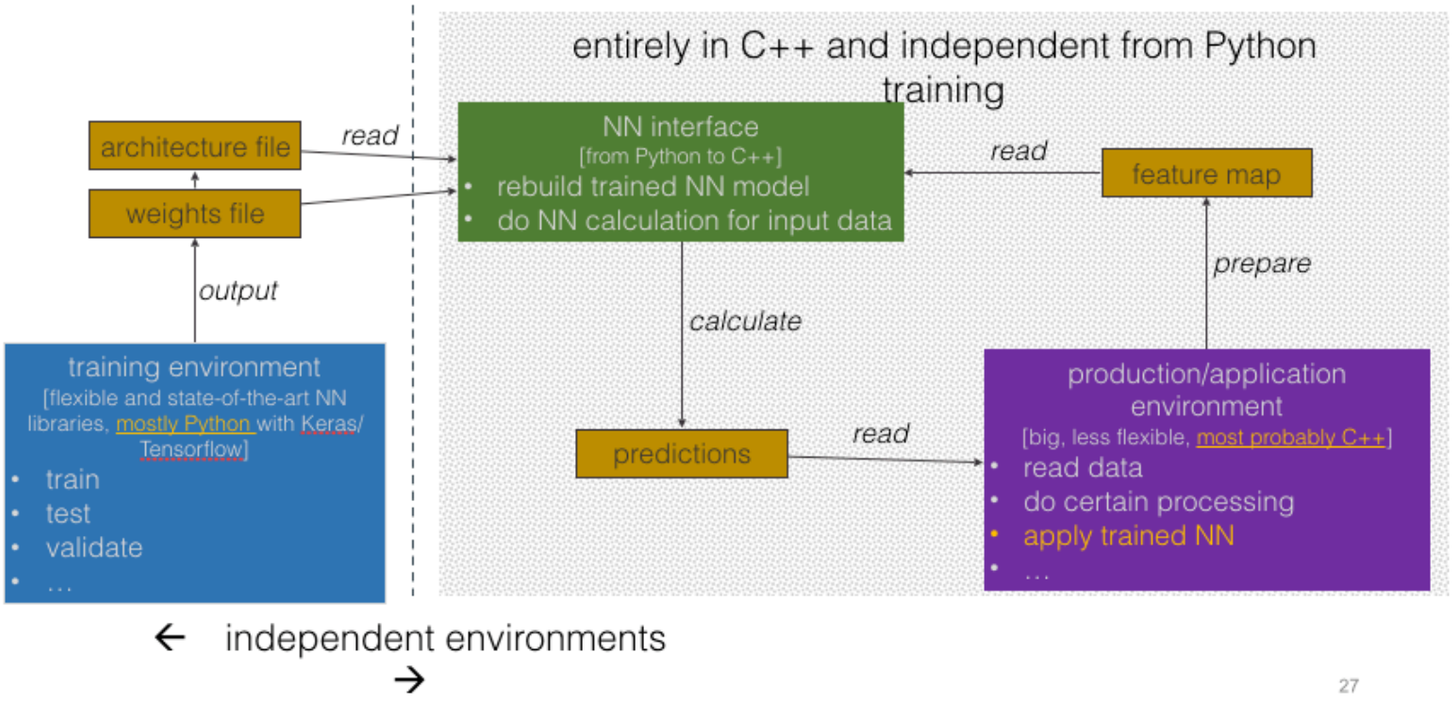
2) Linear and non linear regression

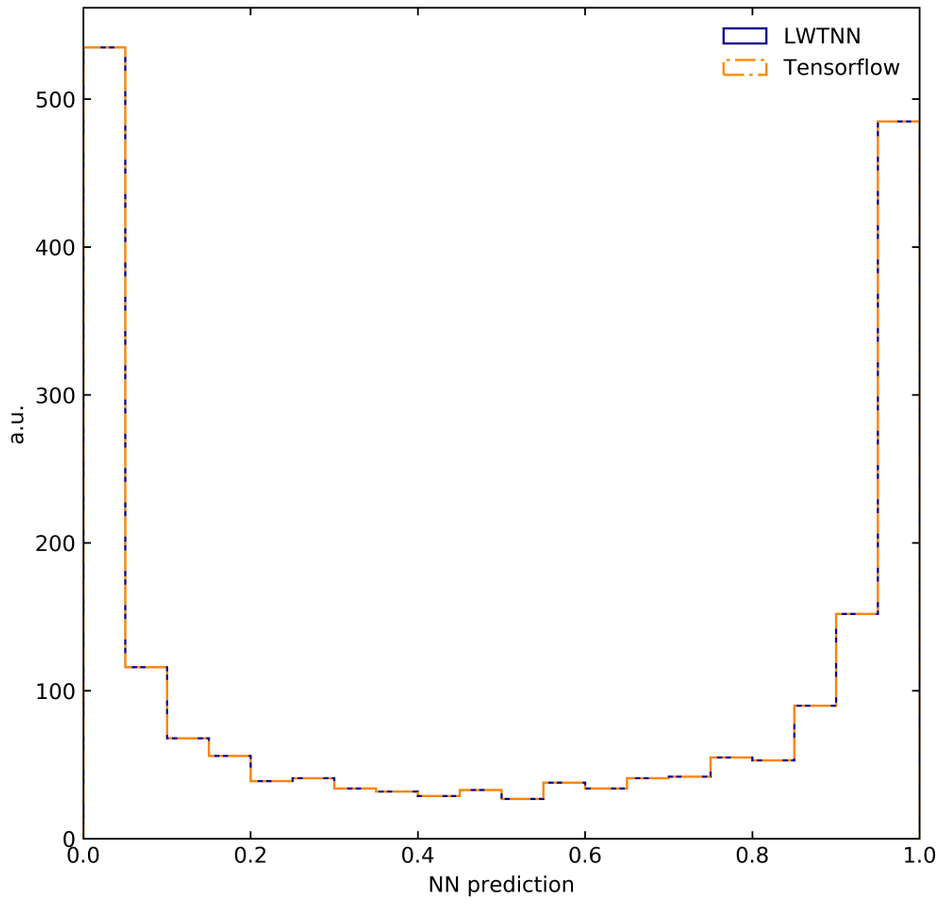
3) A new tool for MC-data reweighing (discussed later)

A C++ converter for ML models

- we developed a framework on an existing library to convert a python-trained DNN model to C++ in order to be run on Grid.

- This package will allow to run the python-trained models on the Grid without the need of producing ntuples for the entire dataset





- 100% match between the ROC curve obtained in python and the one obtained in C++ after converting the model to C++ !

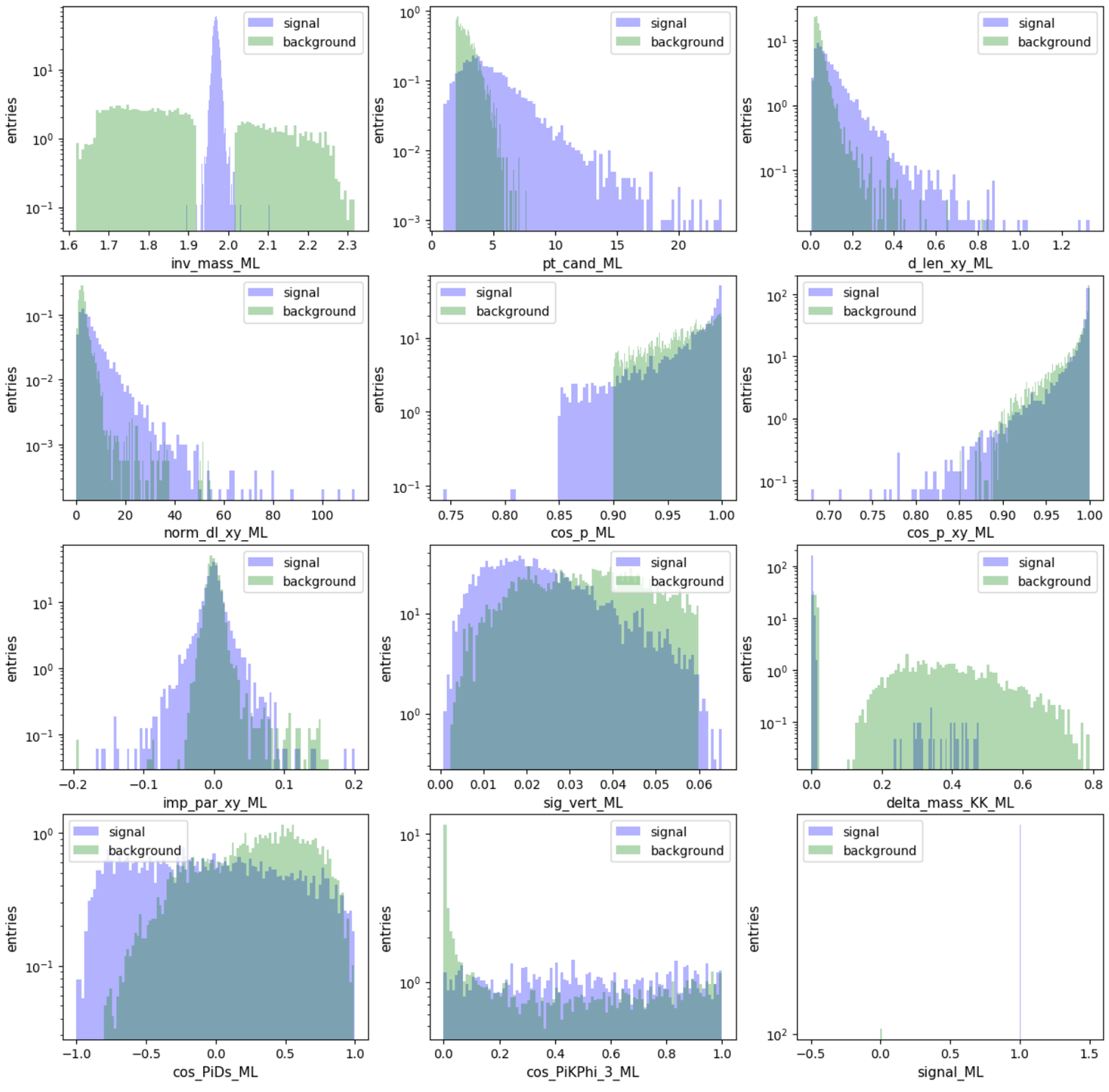
An example of a standard classification analysis*

* this is what you get automatically with the current framework once you configure the handler to “digest” your TTree

Studies of input variables

signal

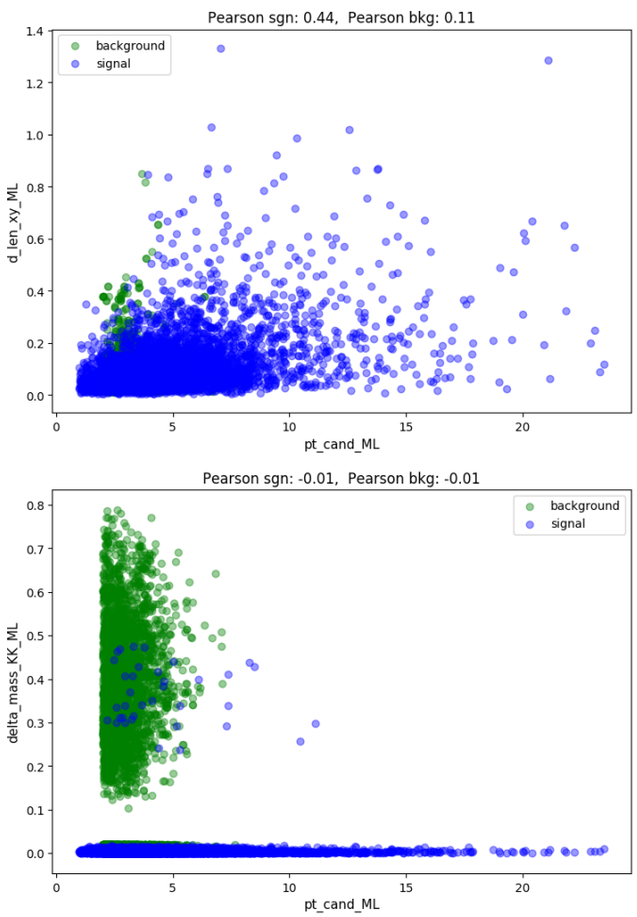
background

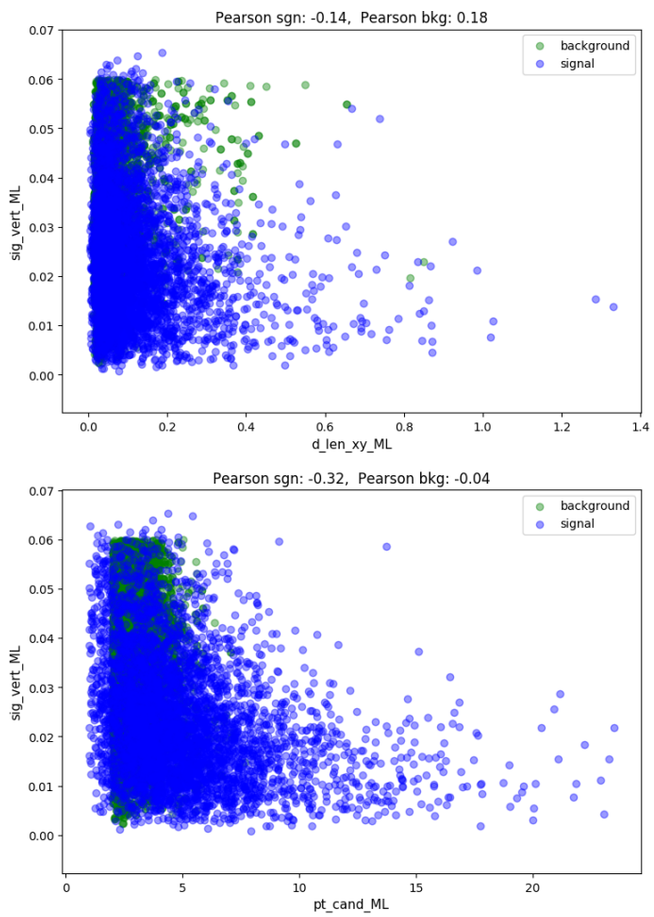


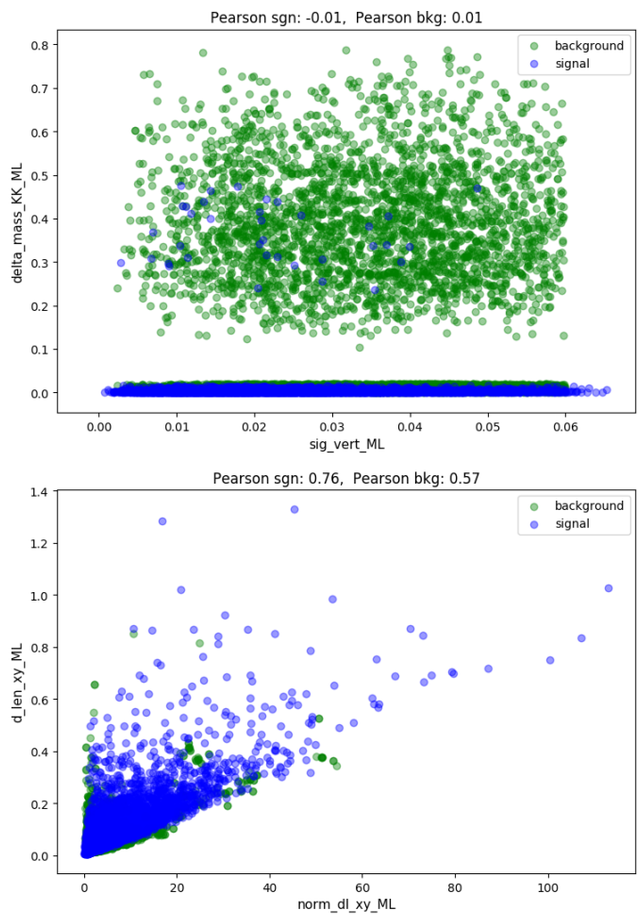
Variable correlations

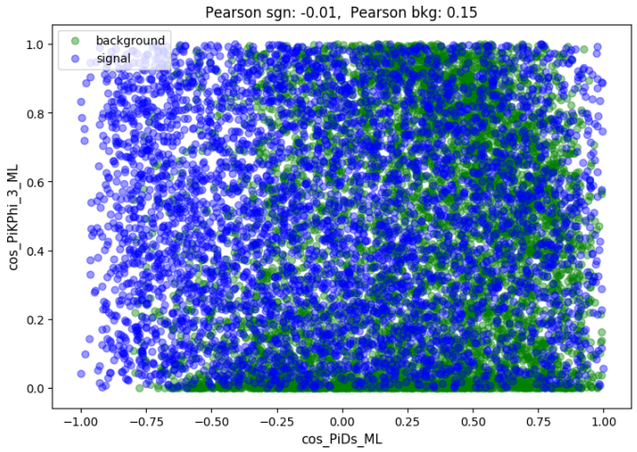
signal

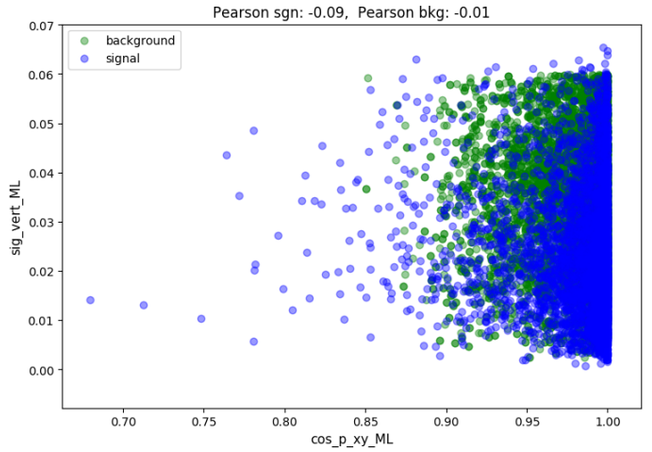
background

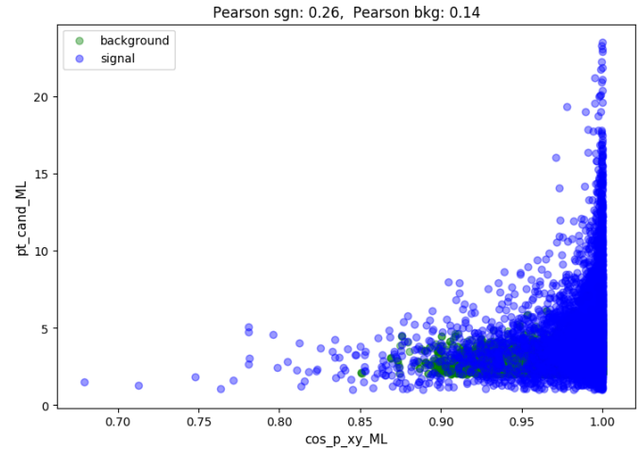




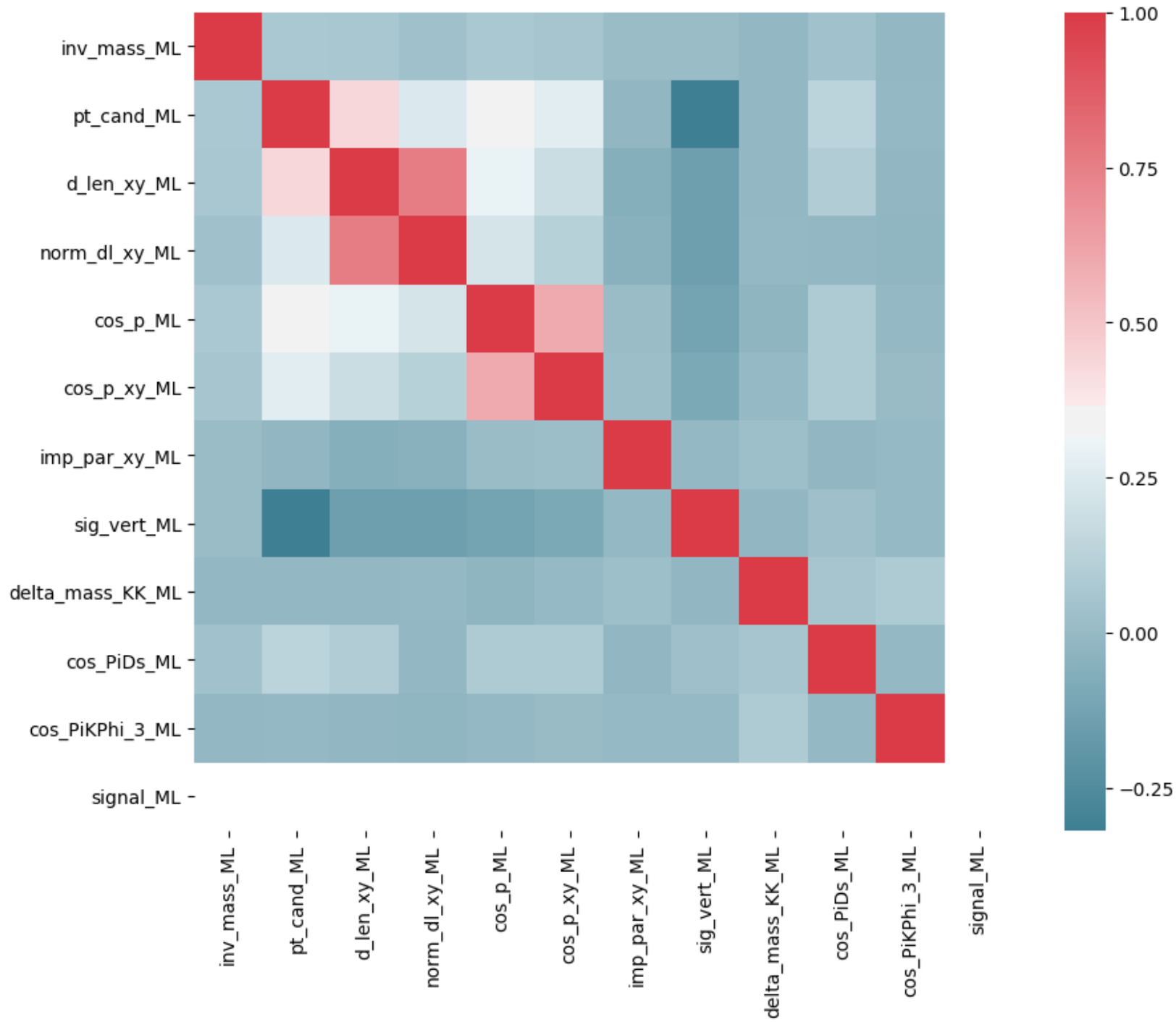








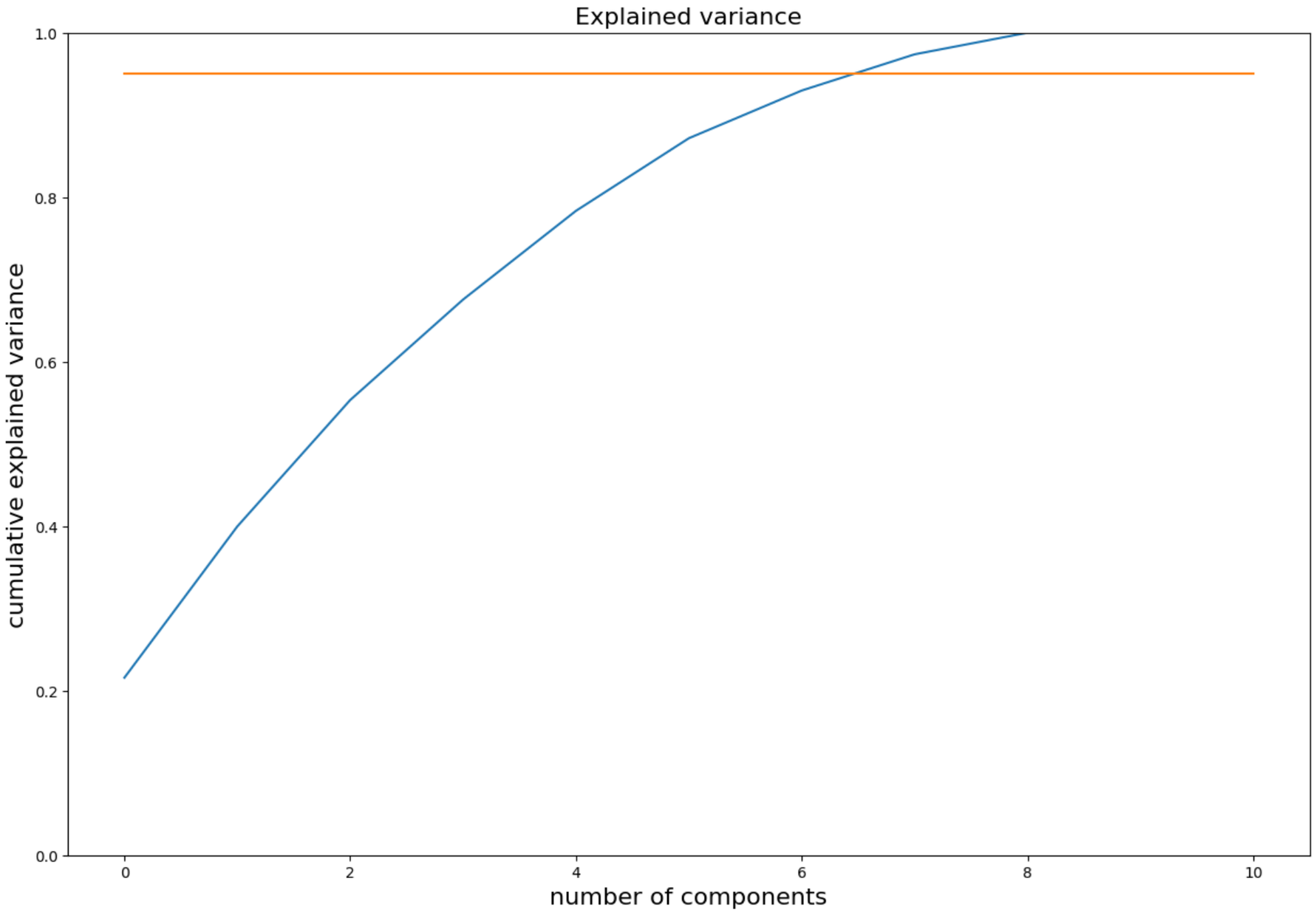
Variable correlations



- Study of correlation between variables is the first step to perform dimensional reduction studies of the selection variables like Principal Component Analysis (PCA) etc

PCA and standardization

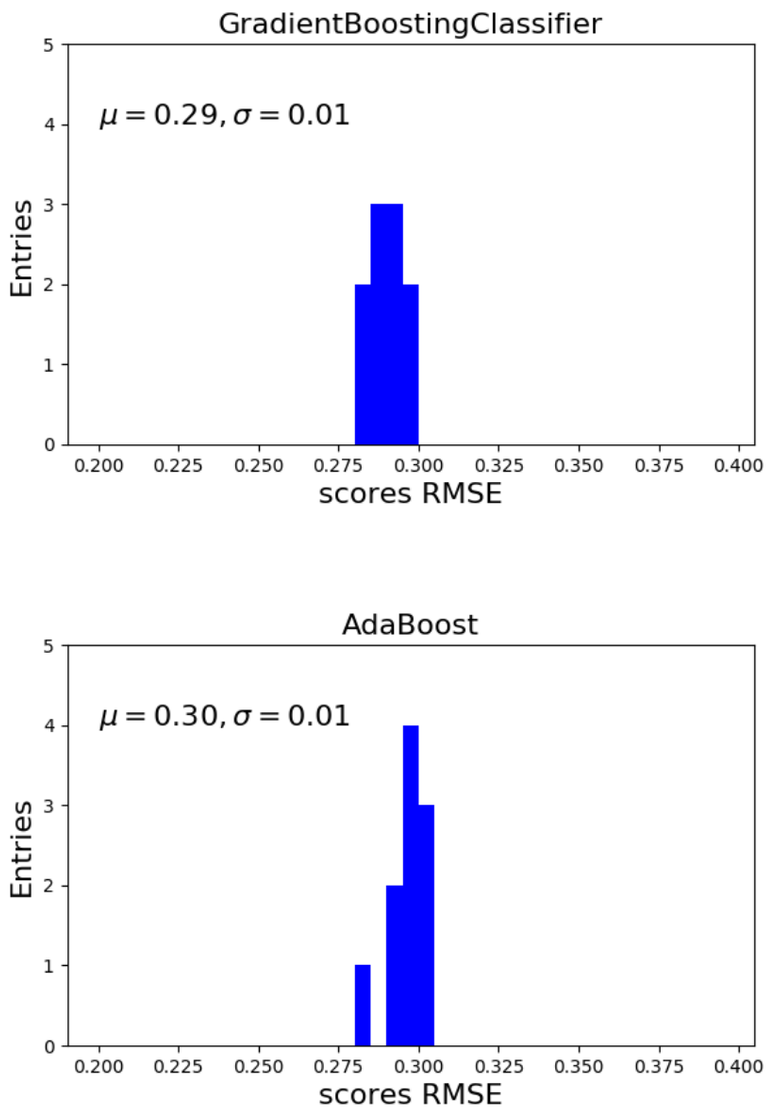
- You can automatically process the variables to performe Principal component analysis and variable standardization

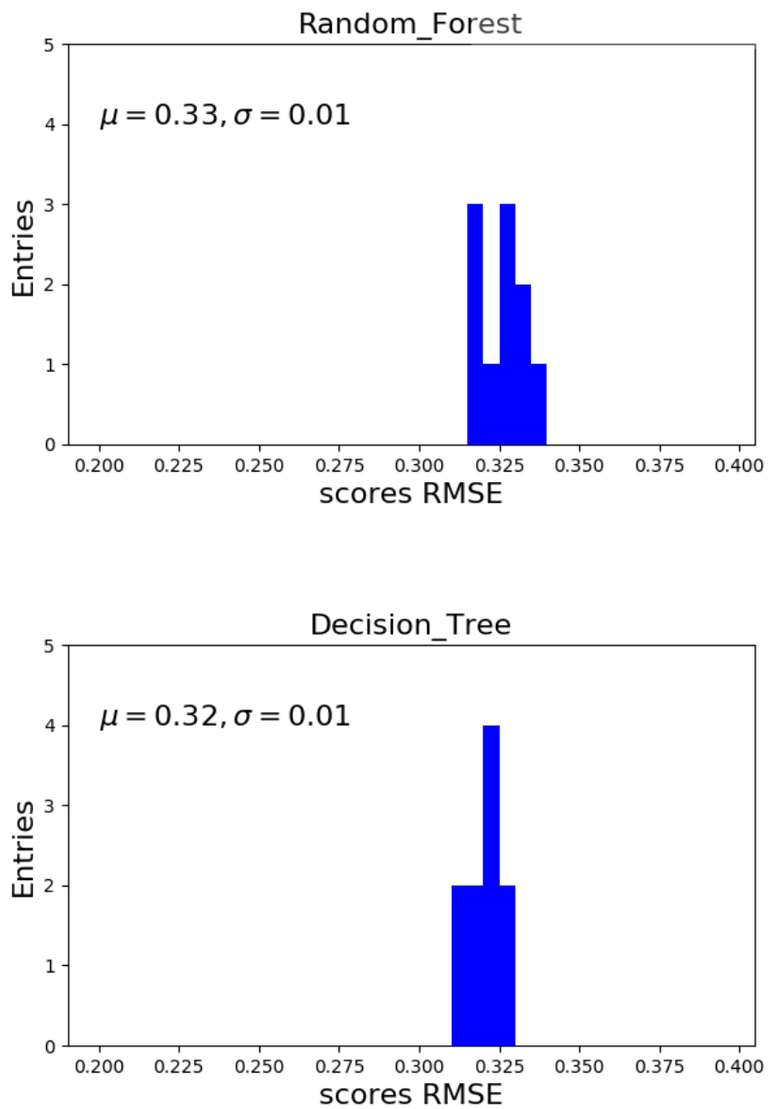


Algorithm performance with cross-validation

training sample = subtraining + validation

testing sample

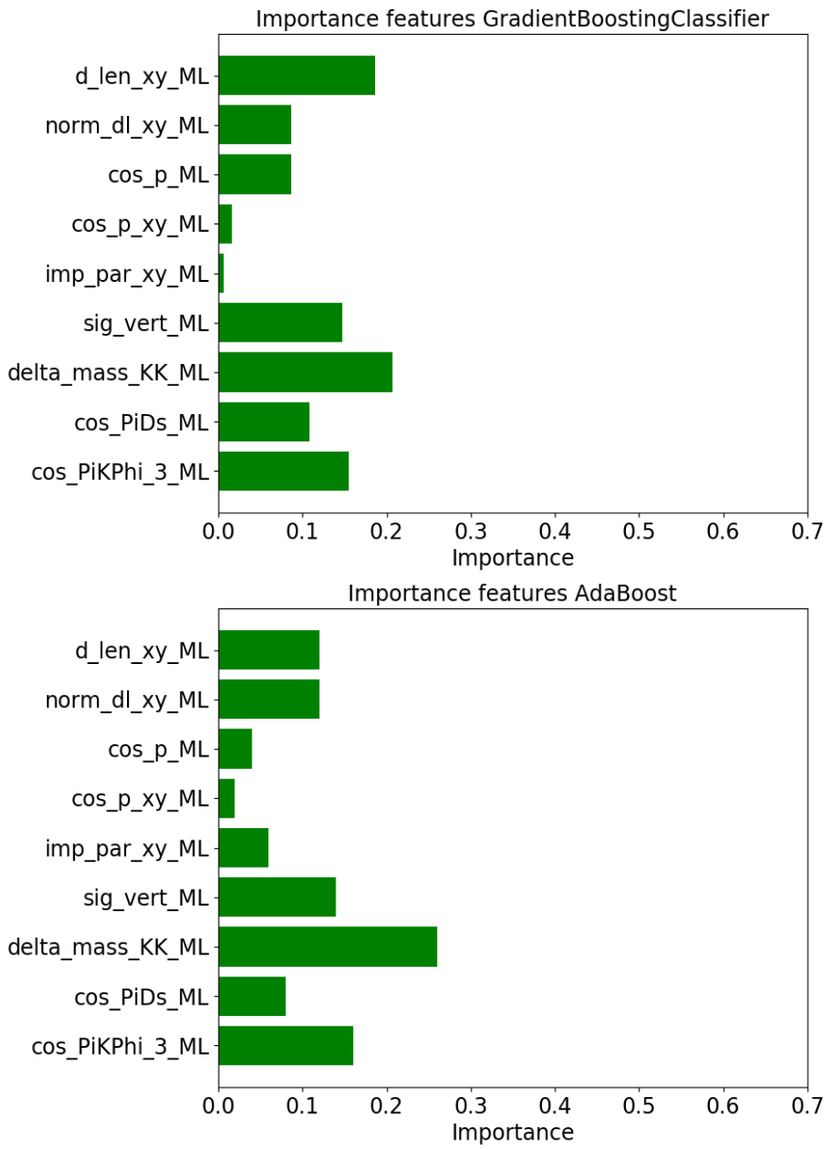


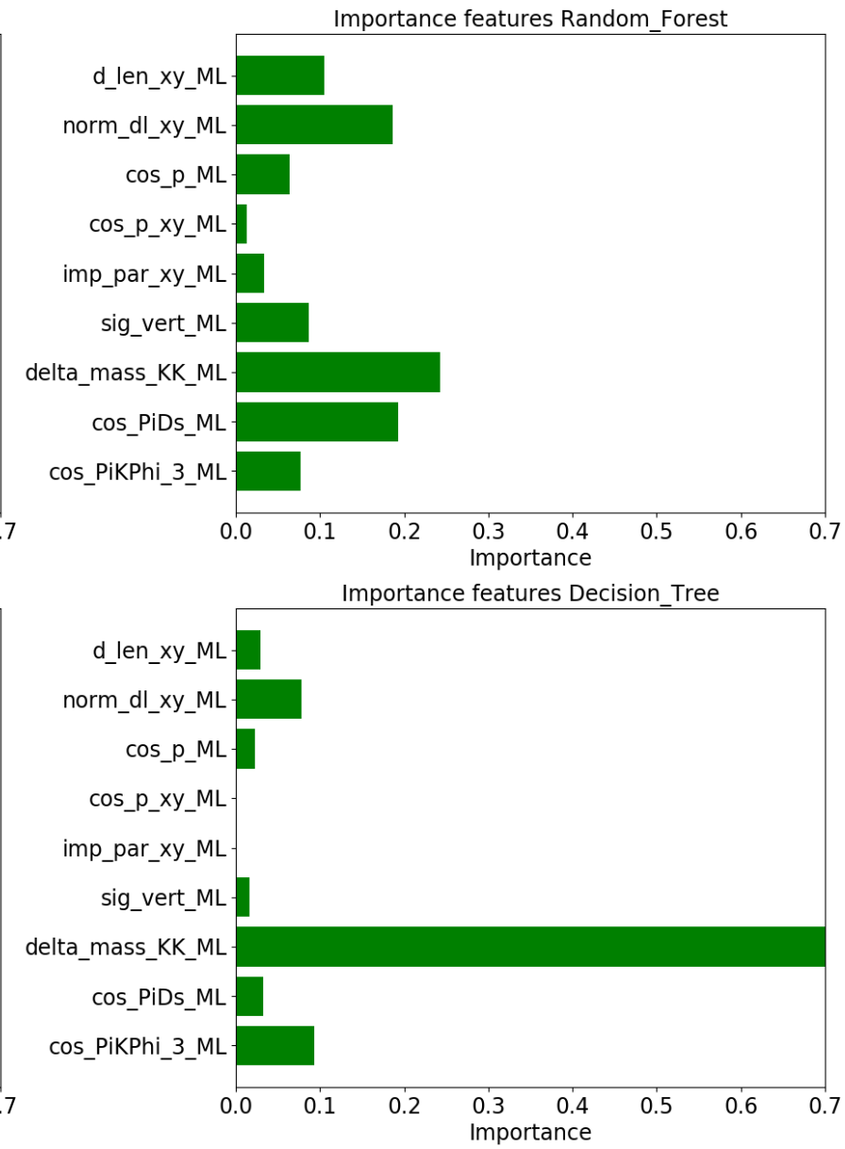


The study of the algorithm performance was performed with a standard 10-fold cross validation method using as scoring function the RMSE of ML decision calculated with respect to the signal/background flag.

Feature importance

For each of the algorithm, we also had a look at the relative “importance” of the selection features (~ the number of times a variable is used in the tree)

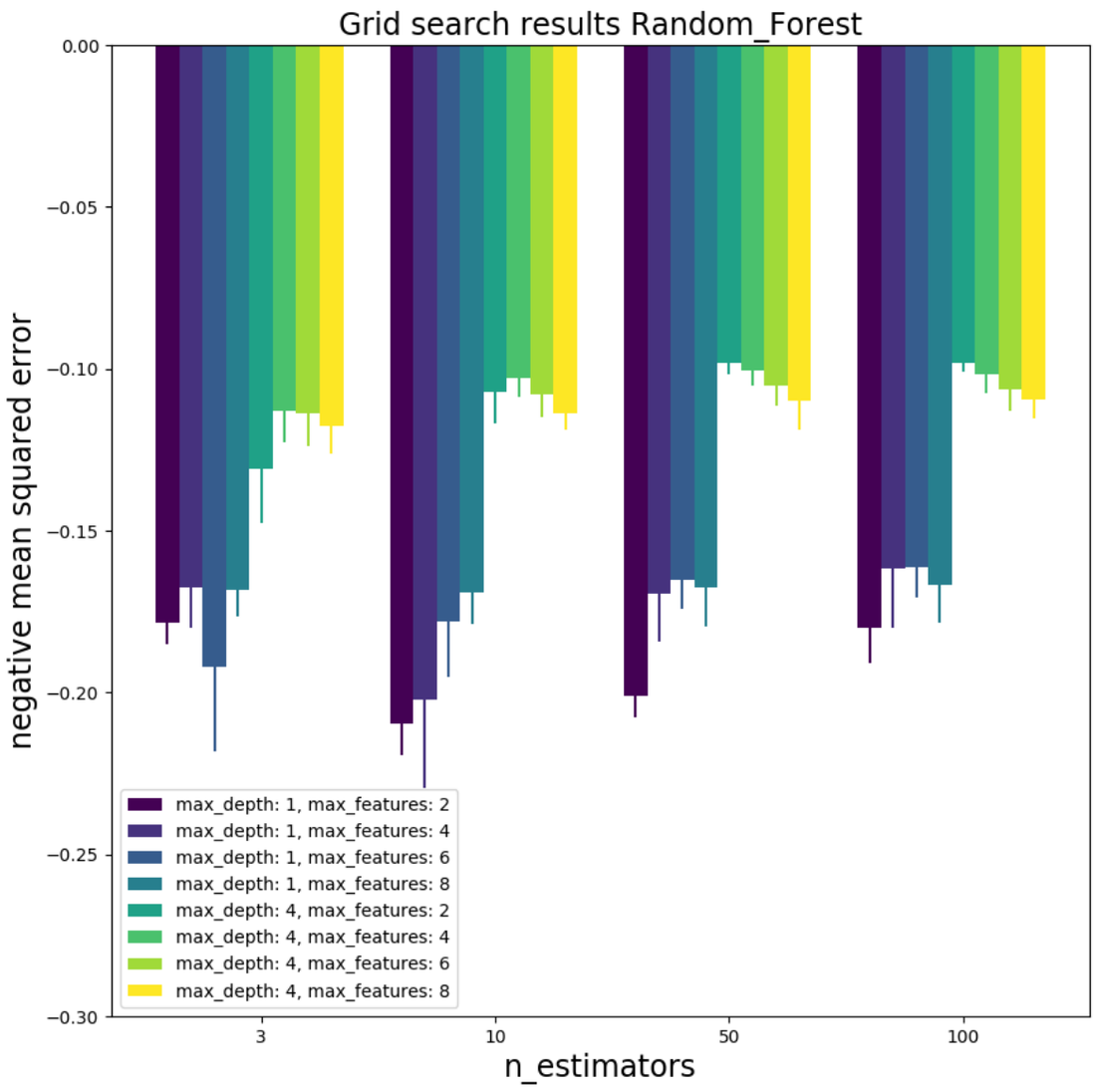


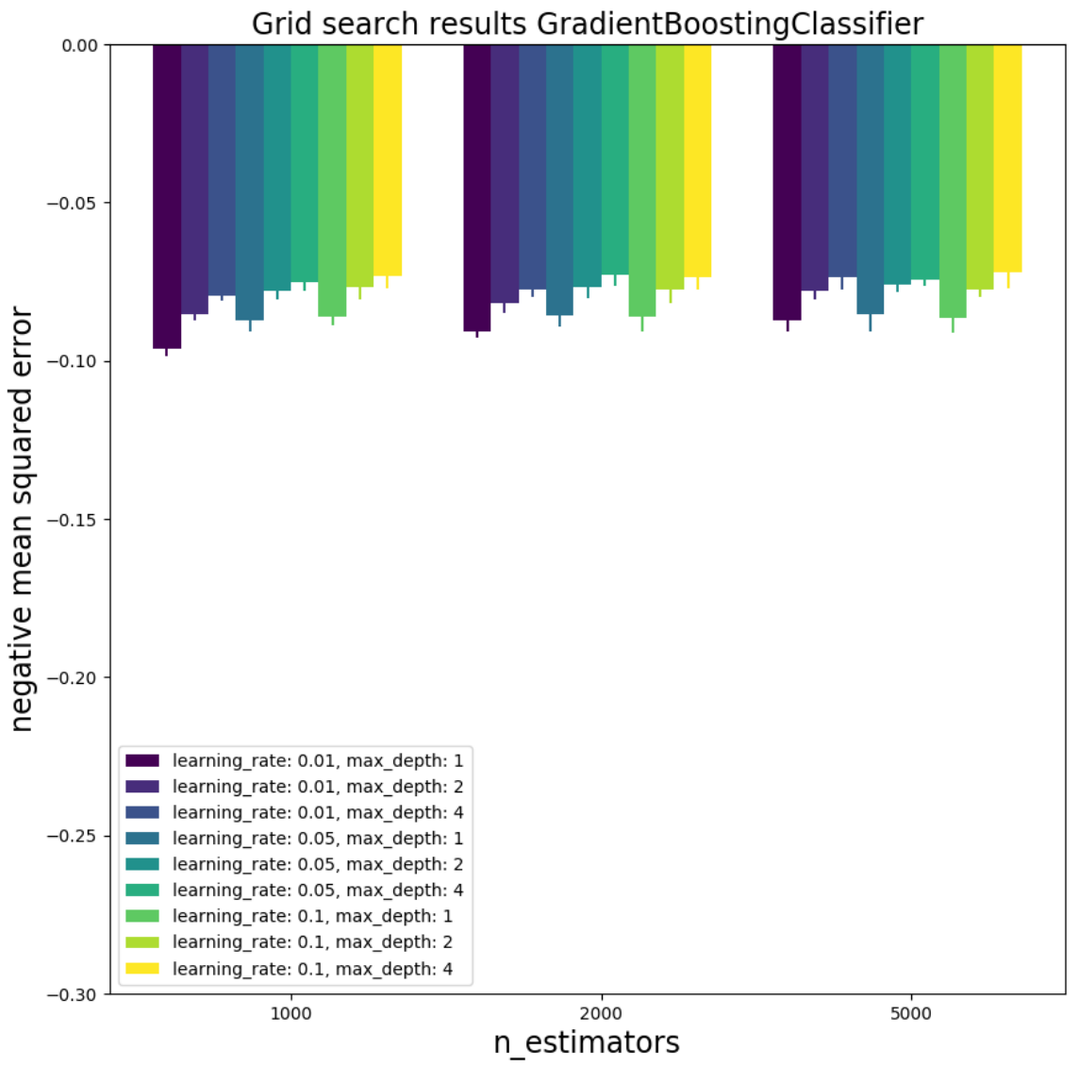


Very interesting to see how the “ensamble” and the boosting “redistributes” the importance of the selection variables in the simple binary tree.

Grid Search: first studies

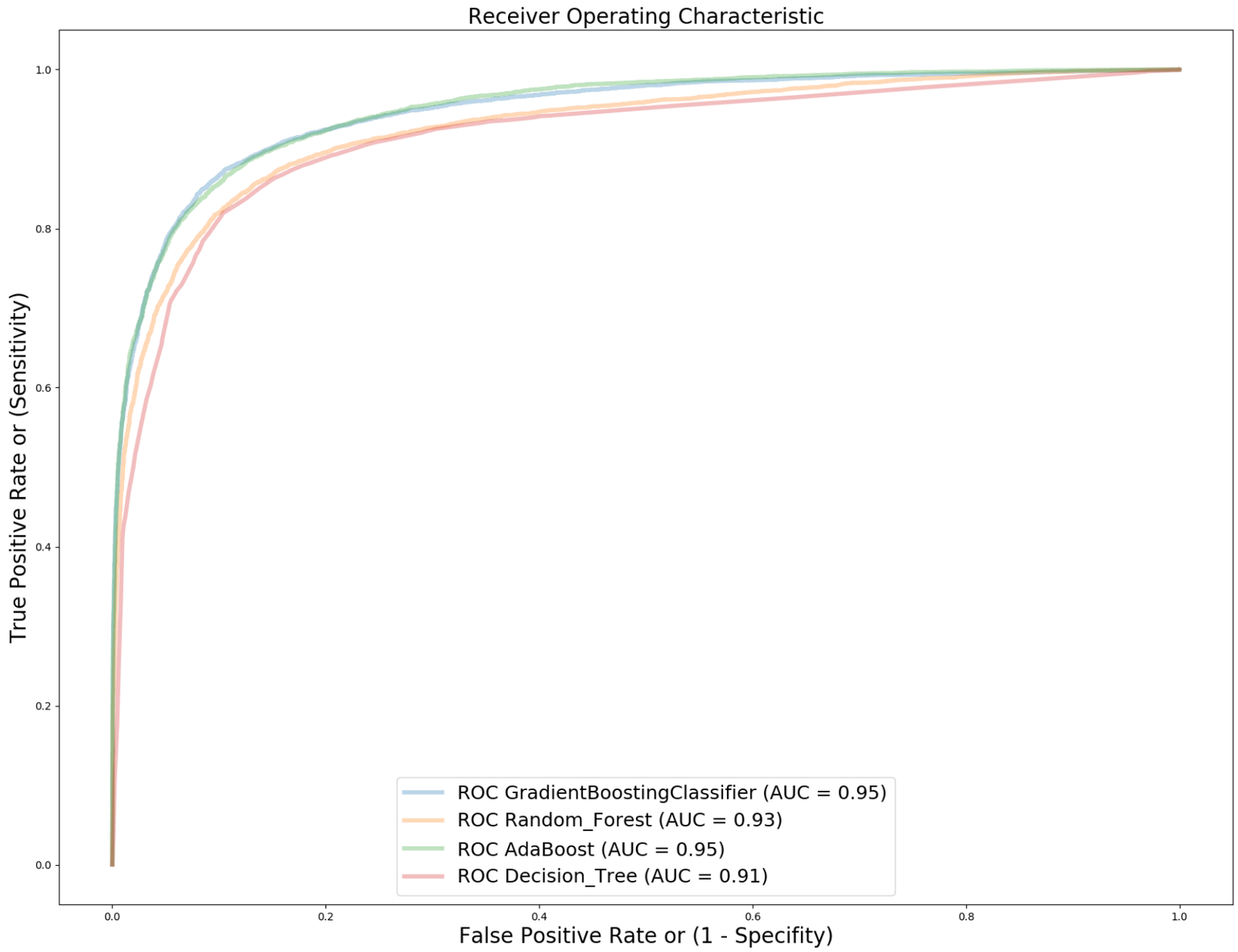
For a couple of algorithms, I started optimising the choice of the algorithm hyper-parameters using the SciKit *GridSearch* method



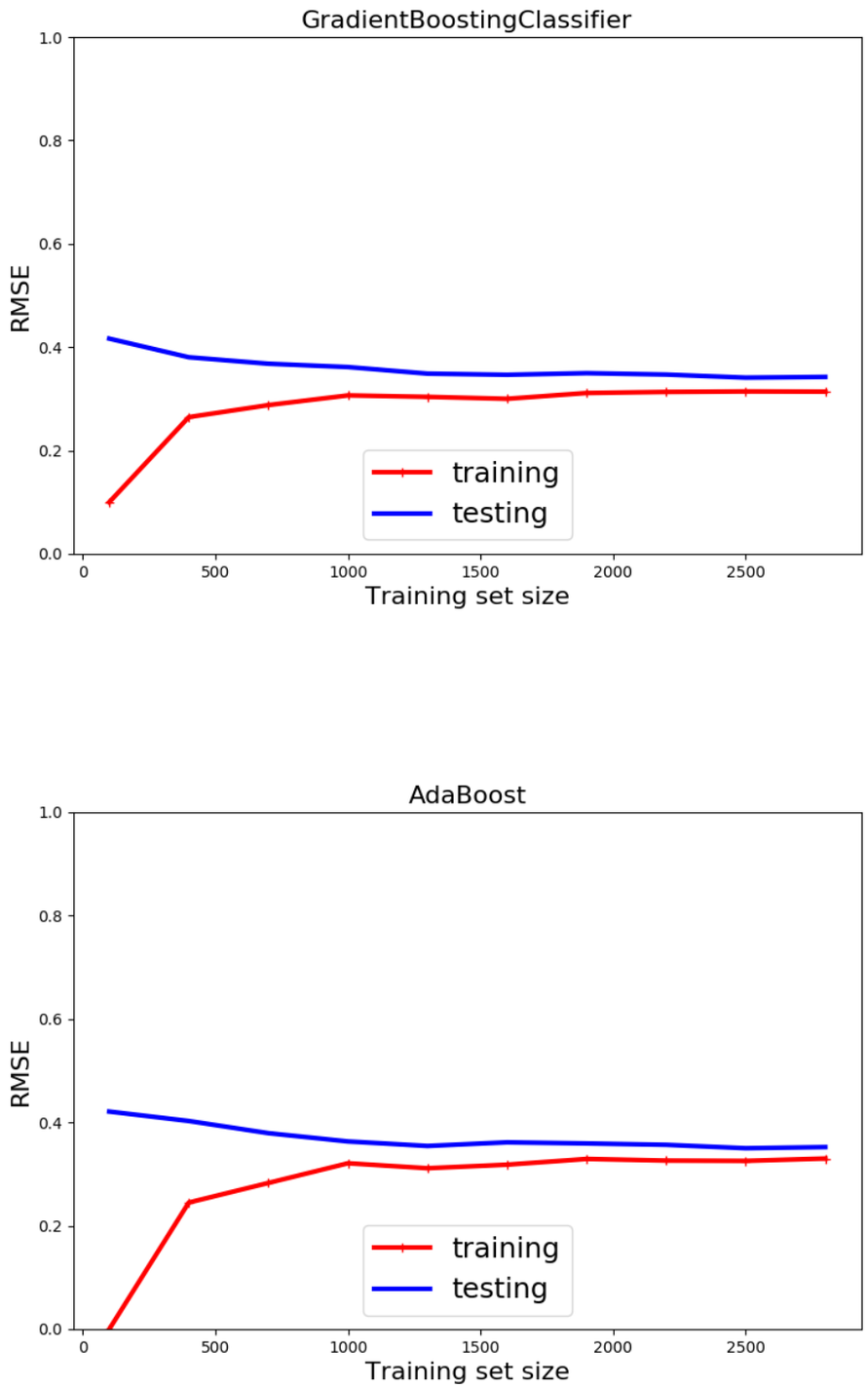


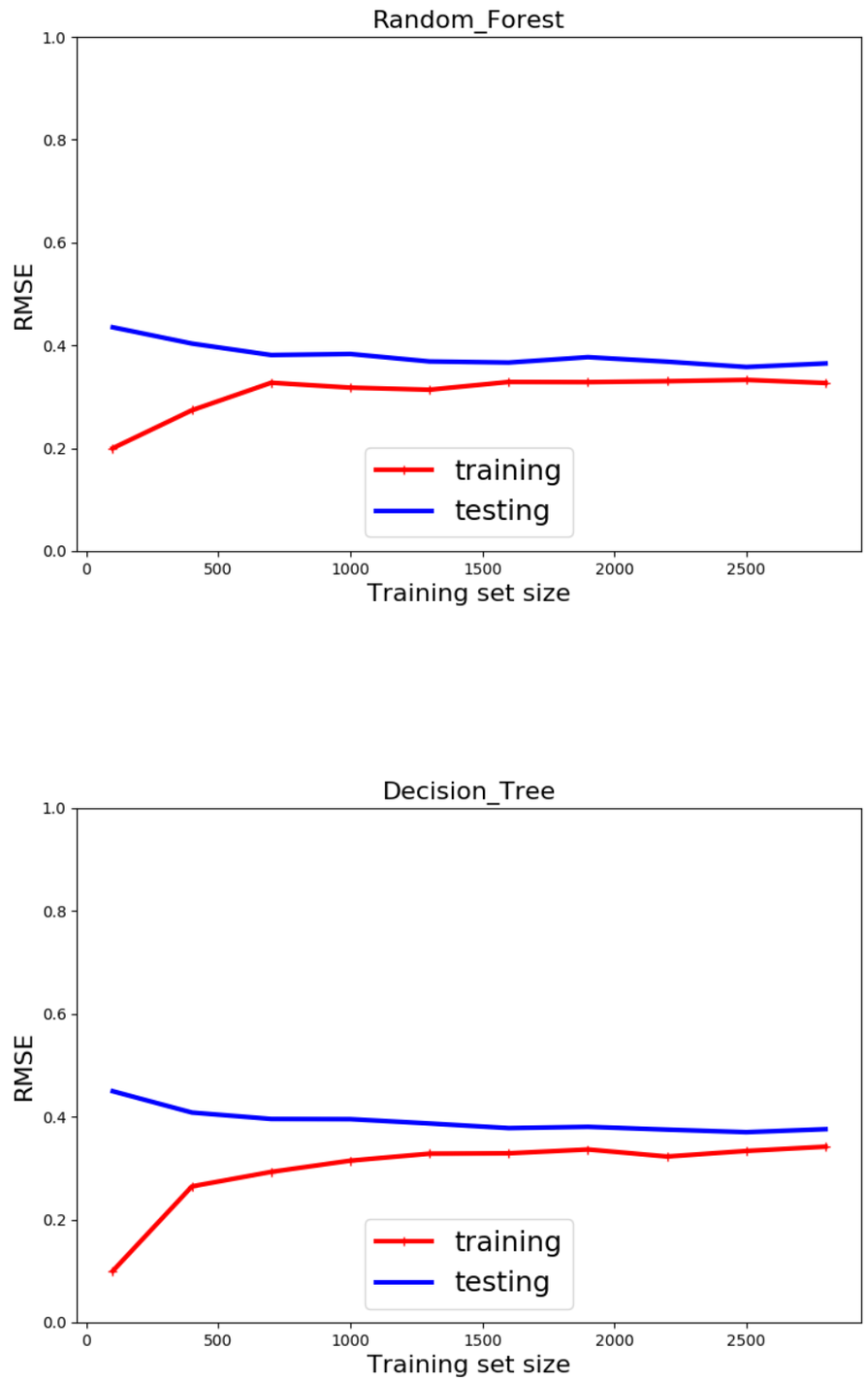
Very first look using a very limited phase space for the hyper-parameters. Need to scan wider phase space using RandomSearch algorithms.

ROC curves

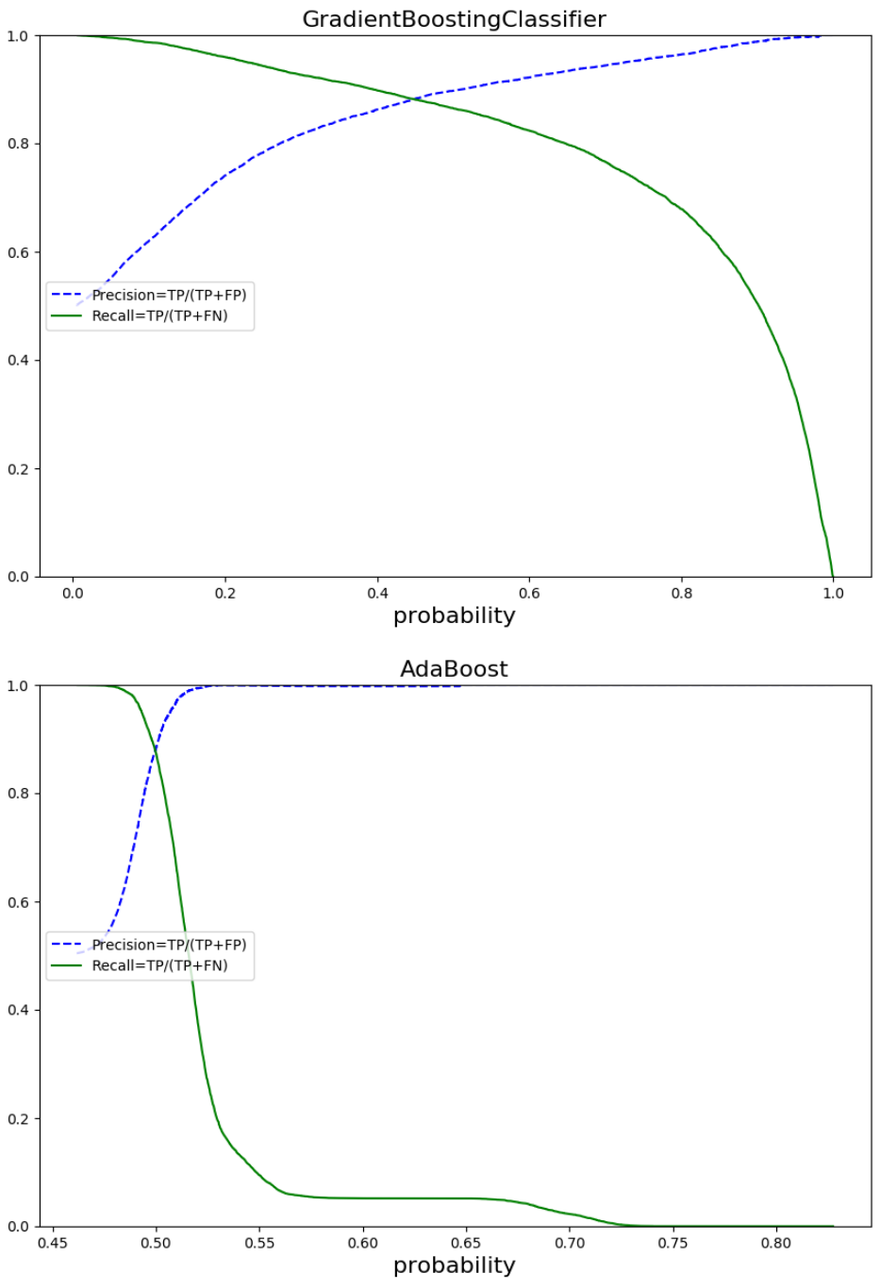

Receiver Operating Characteristic
True Positive Rate or (Sensitivity)
False Positive Rate or (1 - Specifity)
0.0
0.2
0.4
0.6
0.8
1.0
ROC GradientBoostingClassifier (AUC = 0.95)
ROC Random_Forest (AUC = 0.93)
ROC AdaBoost (AUC = 0.95)
ROC Decision_Tree (AUC = 0.91)

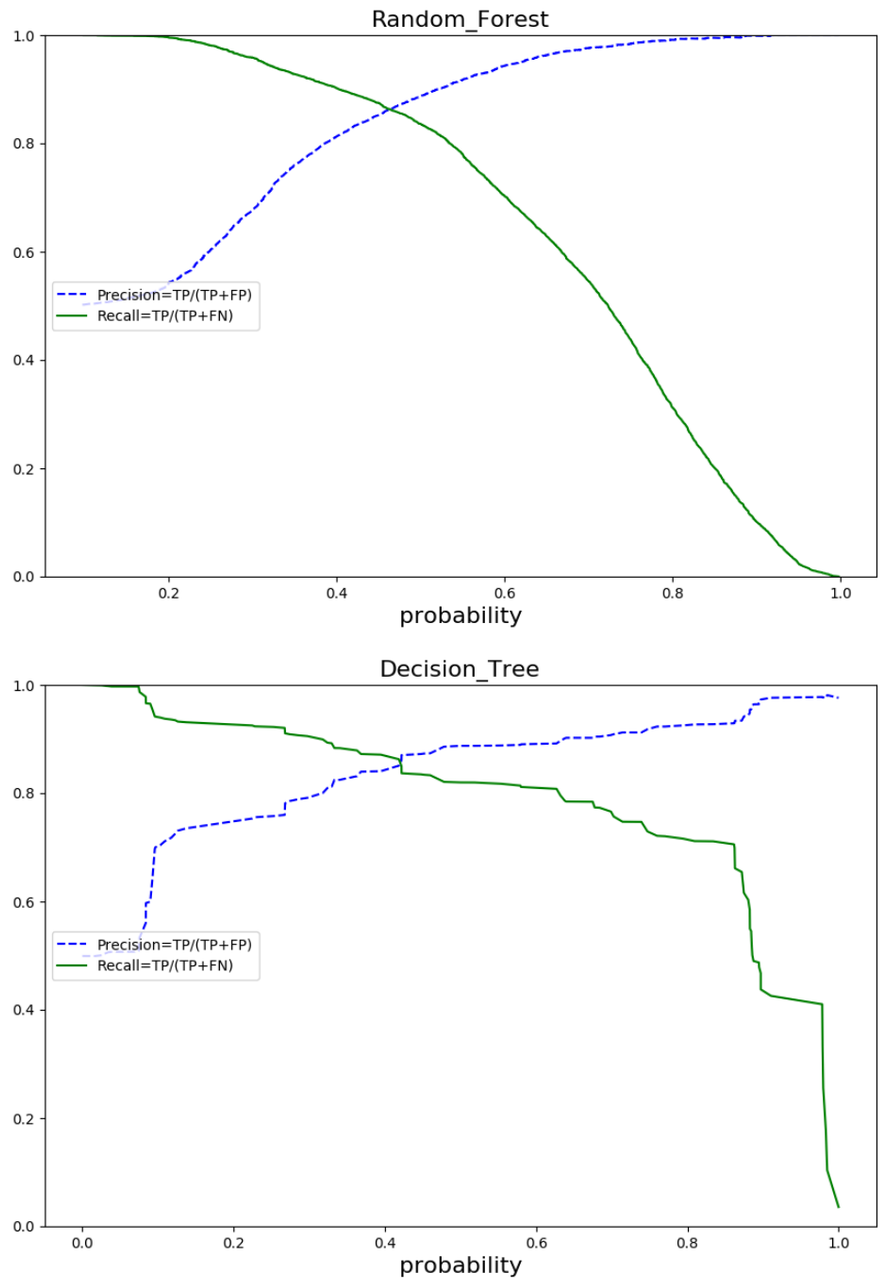
Learning curves



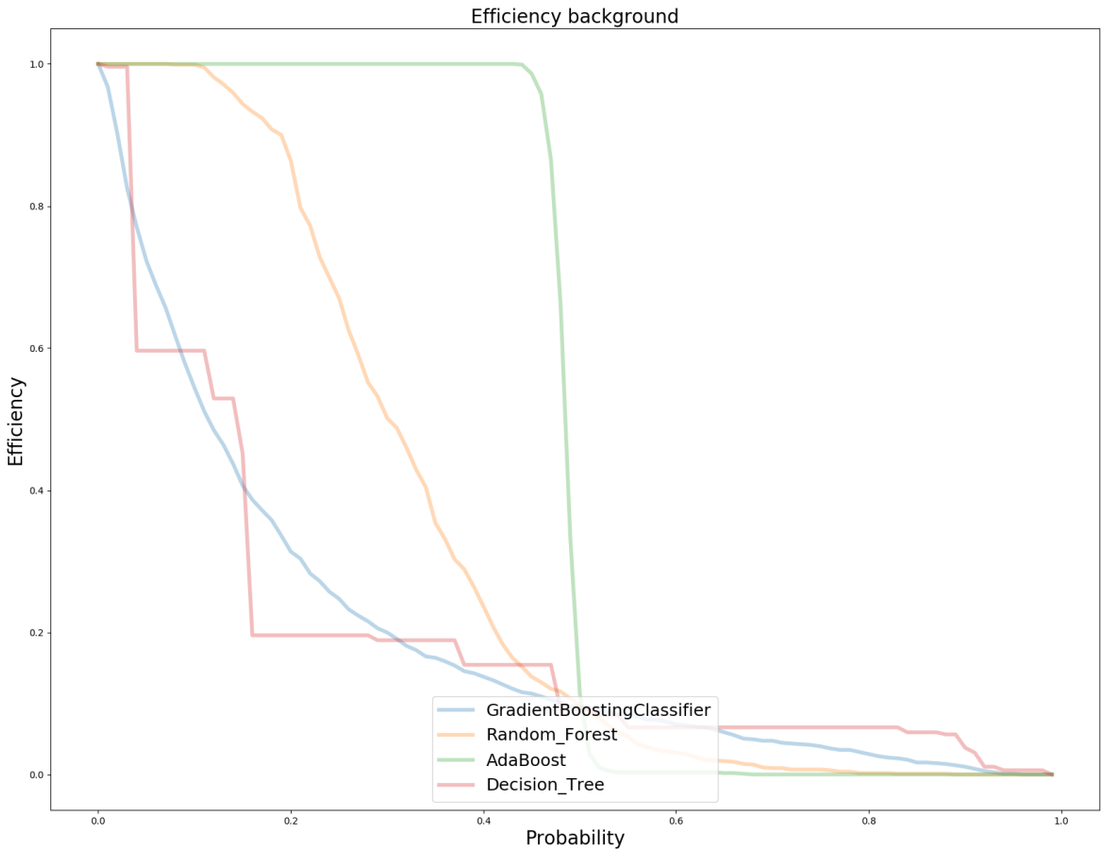


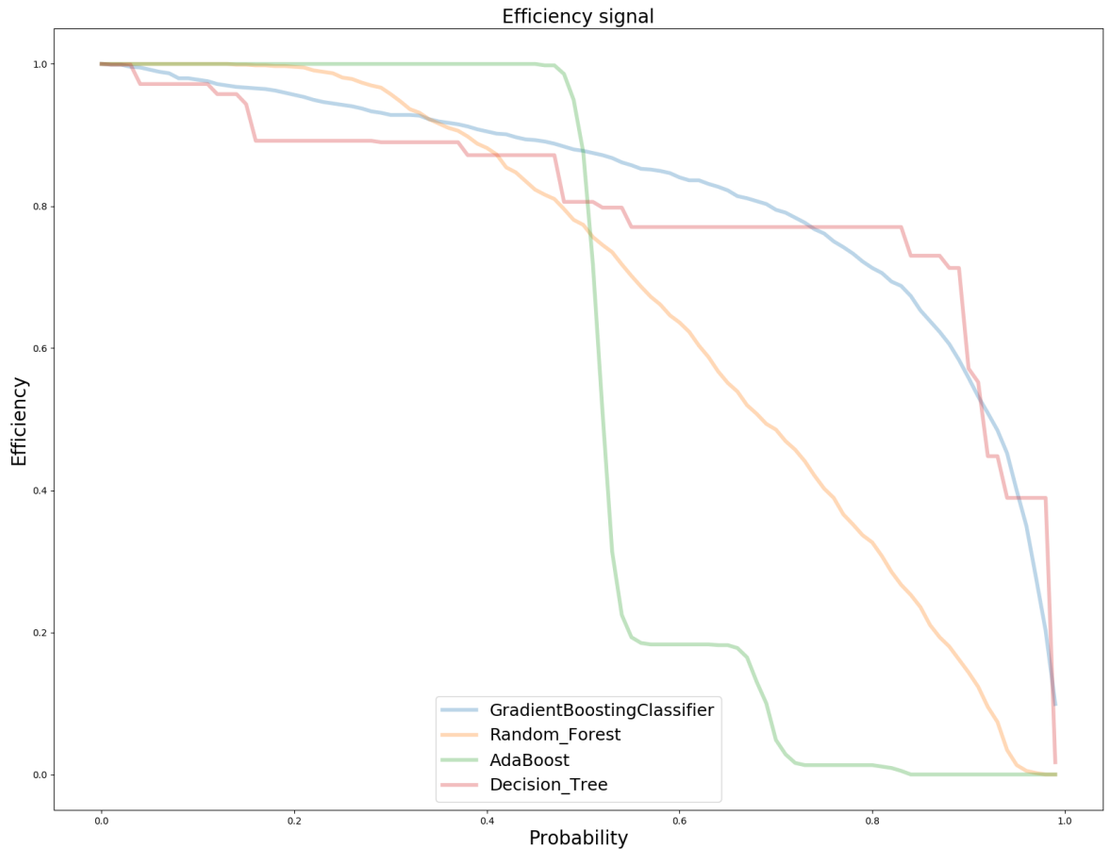
Precision recall function



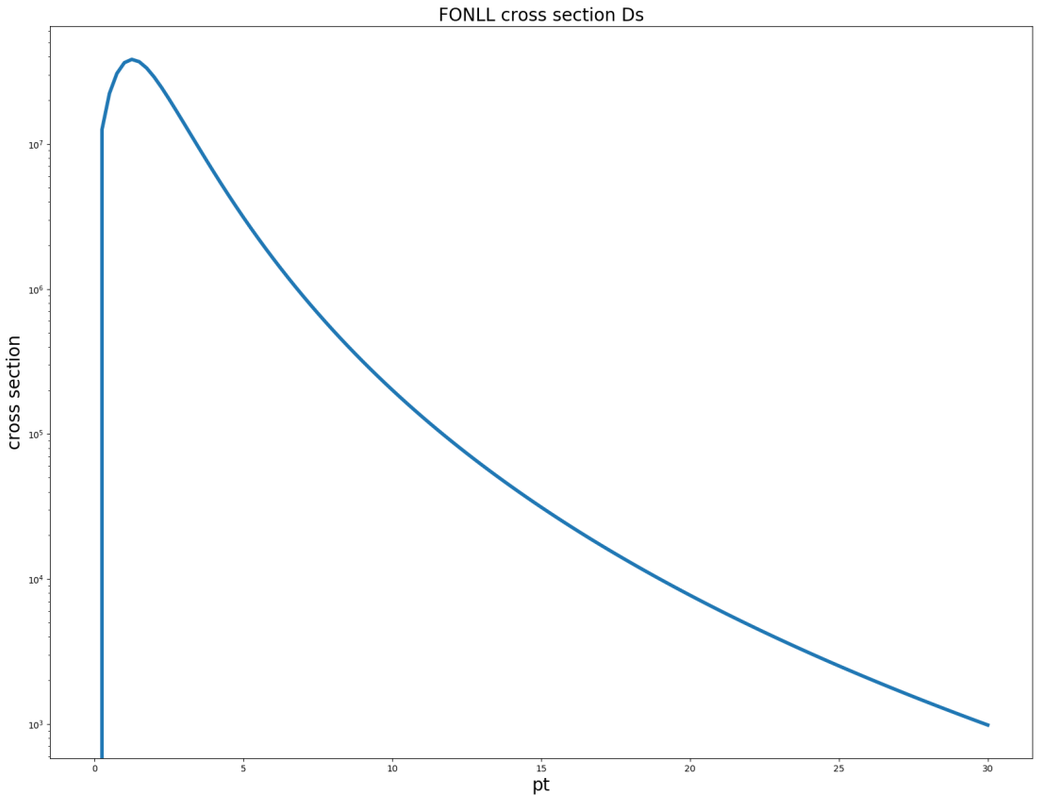


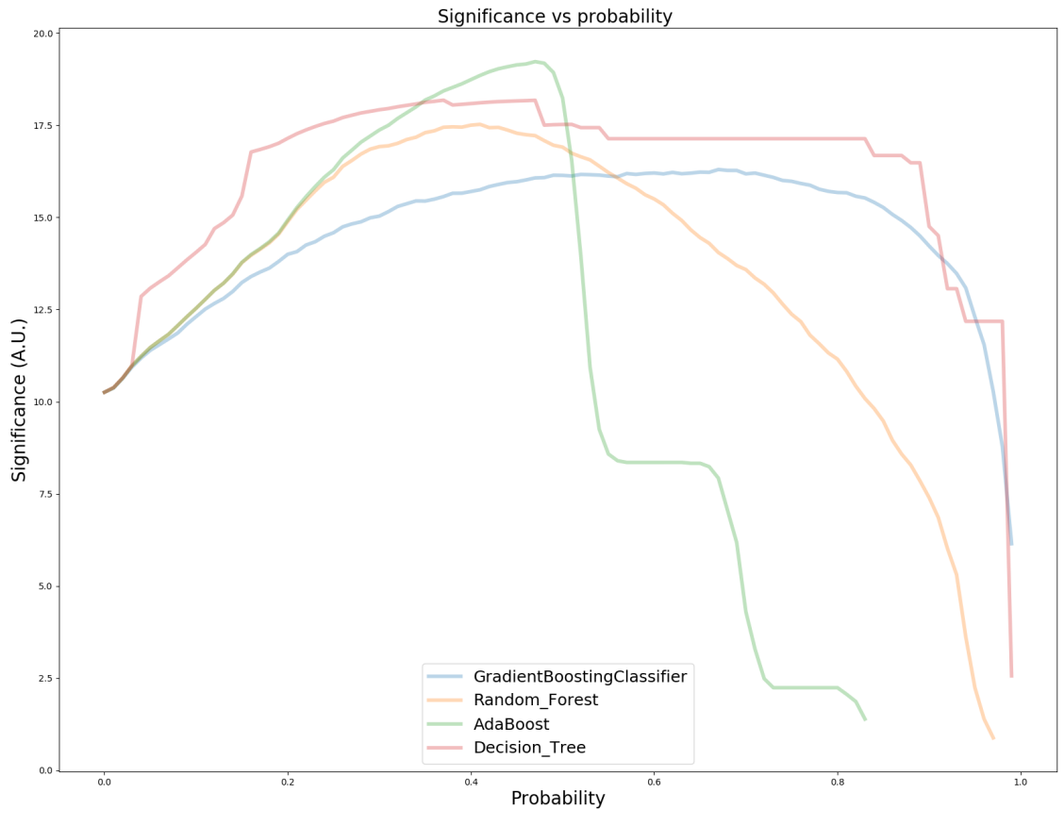
Efficiencies





Significance optimisation (work on going)

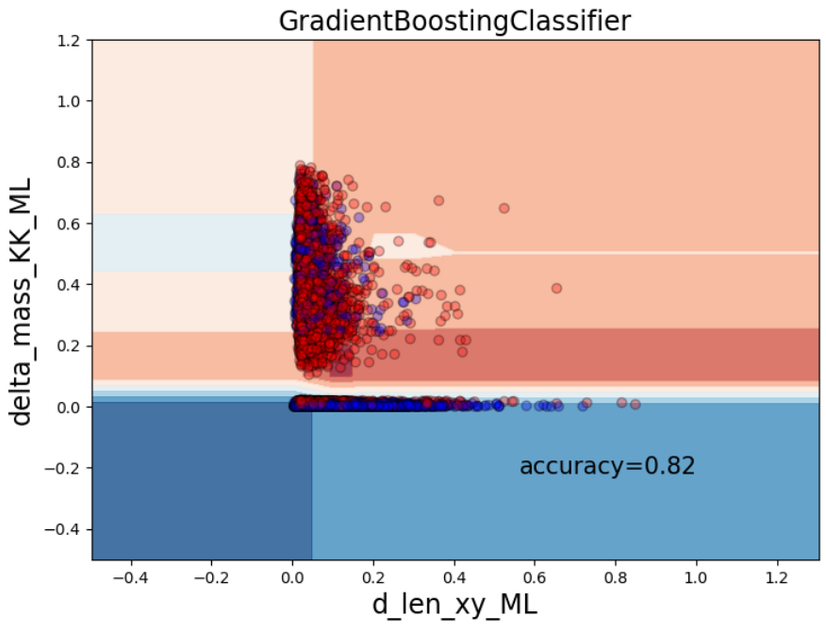


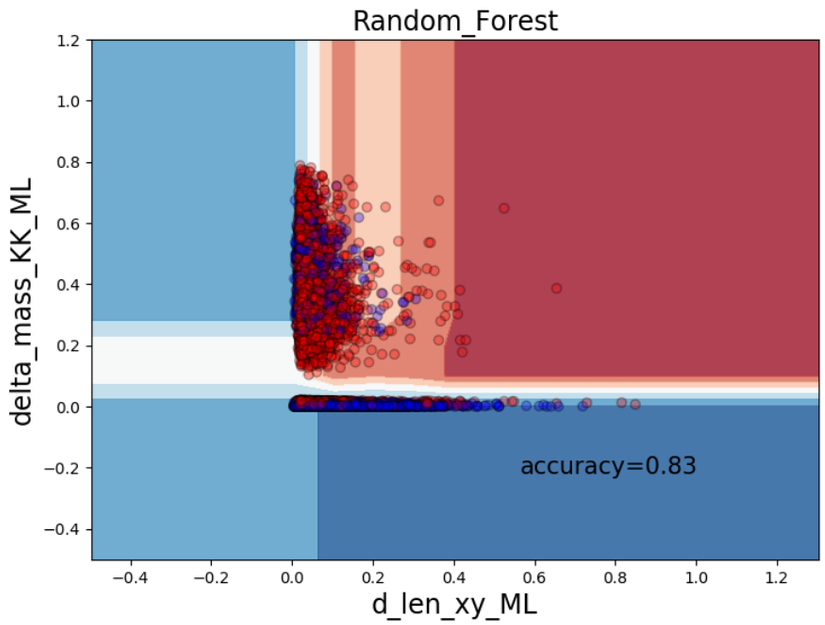


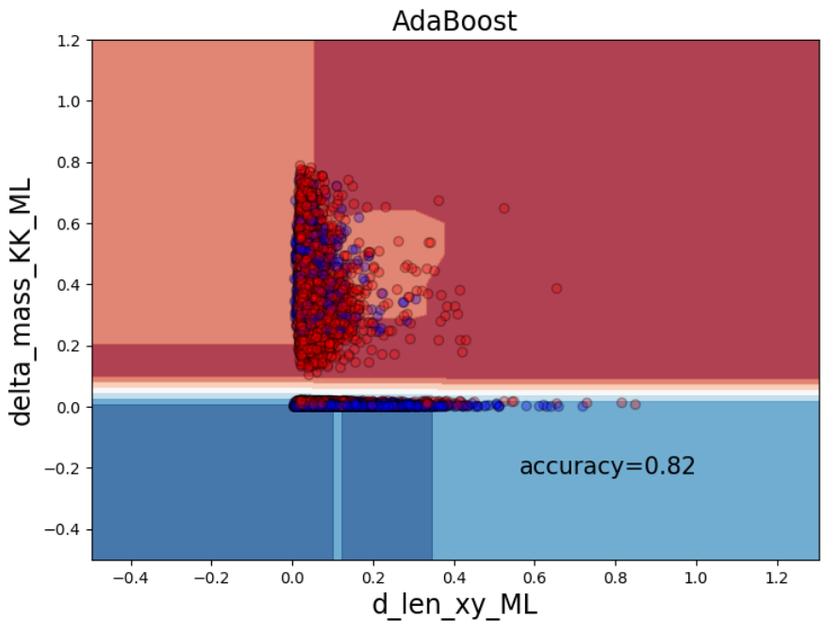
- The code automatically loads a FONLL text file generated with FONLL website.
- Extract the signal in the proper pt range used for the optimisation
- Extract the number of background events in the testing sample
- Combine the signal and background with their efficiencies and extract the value of significance as a function of the threshold on probability

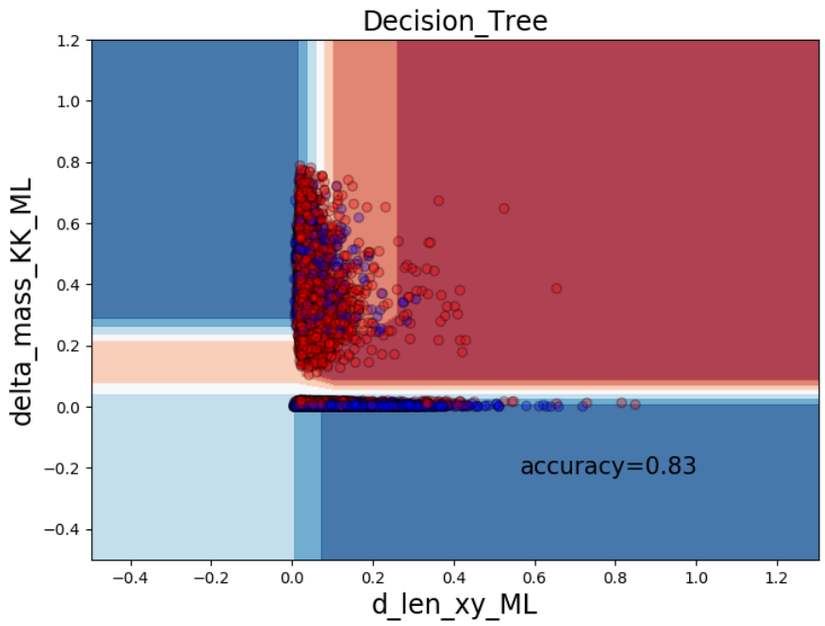
* still need to implement proper estimation for number of background events (after correcting for side band subtraction). Current significance normalization not realistic)

Decision boundaries









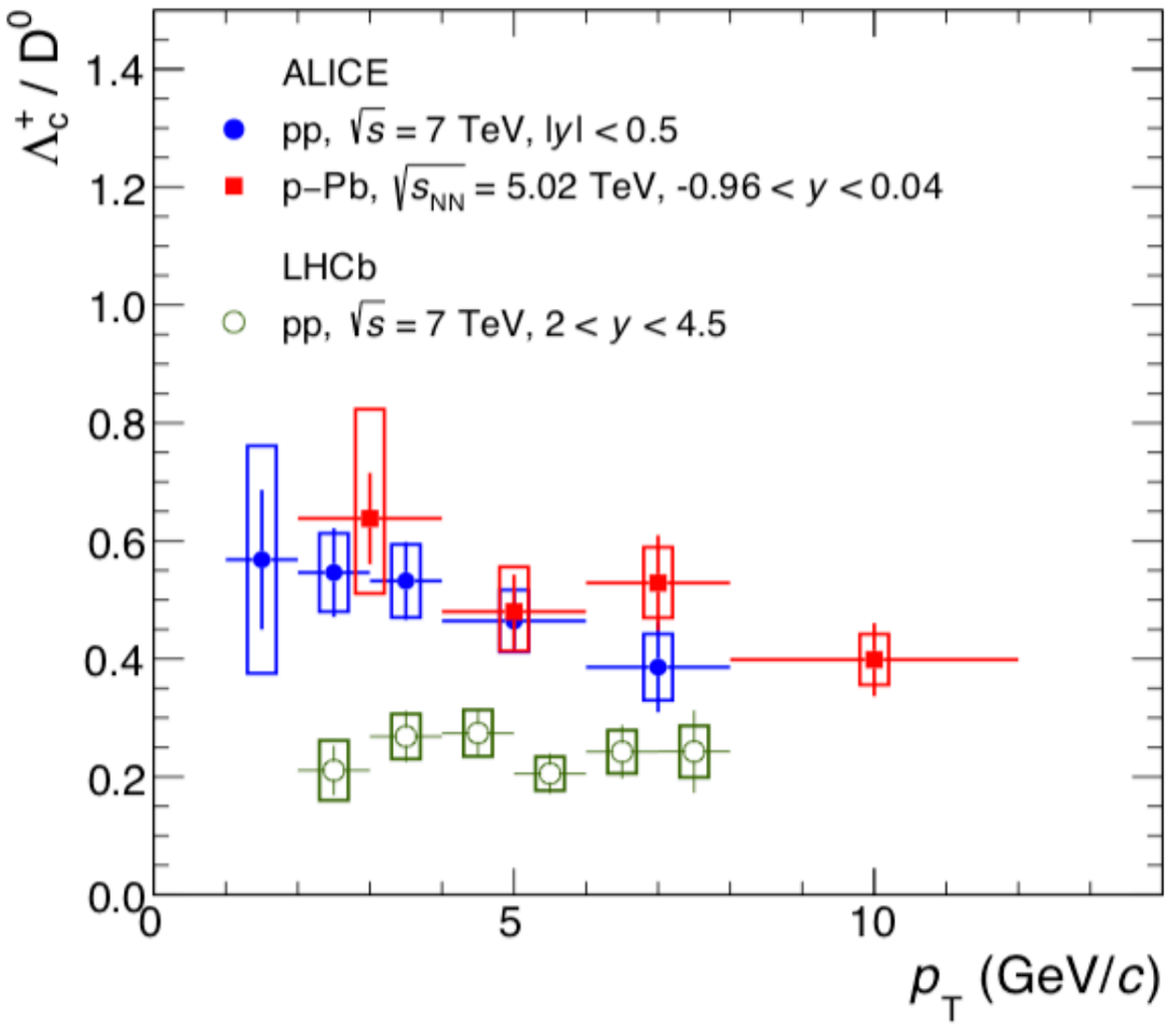
Overview of the current activities

Open heavy-flavour measurements

- push open HF measurements down to very low p_T with more refined candidate selection strategies (Λ_c,D_s ..)

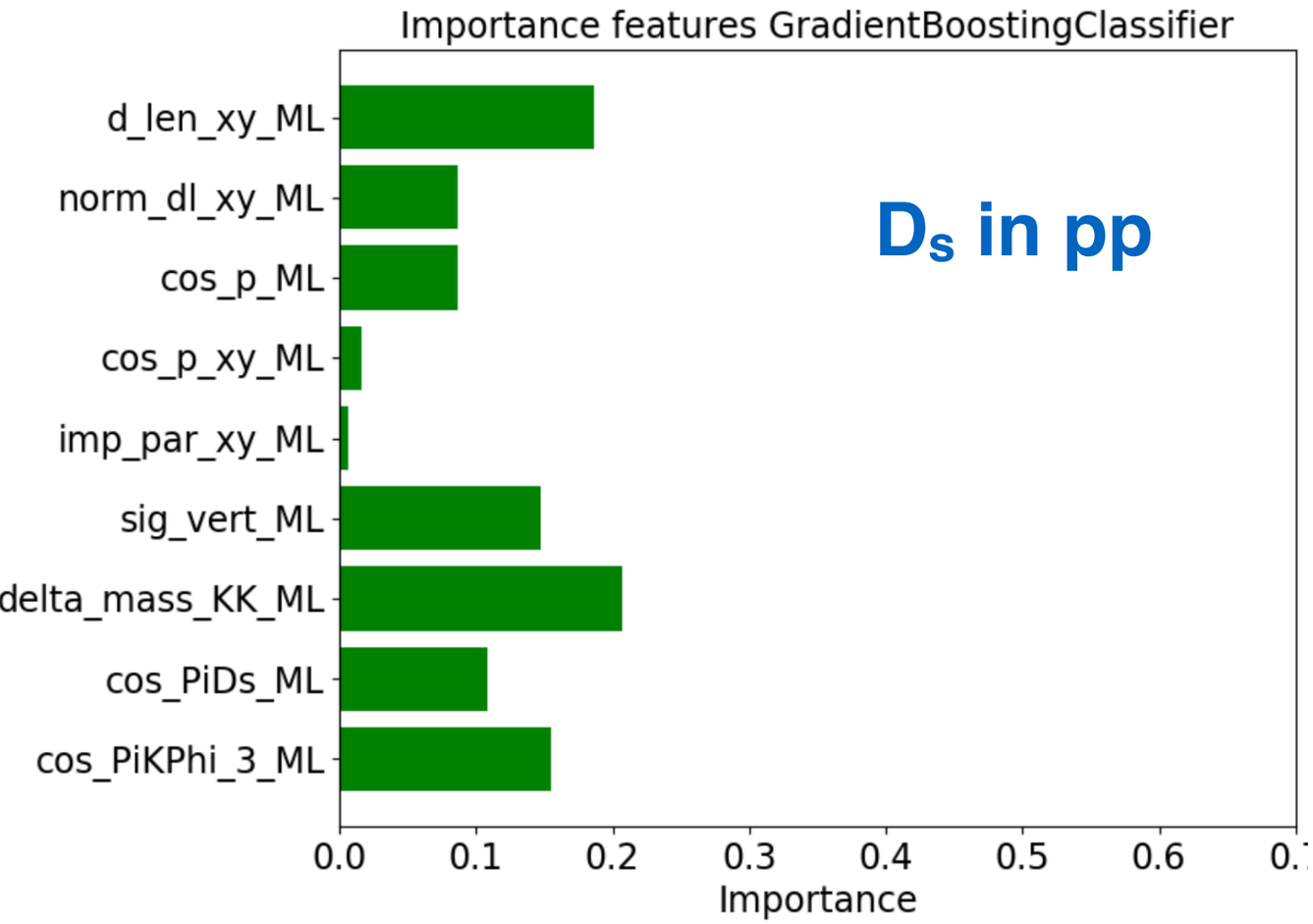
- Optimize dedicated selection techniques for minimizing errors on ratio of particles (e.g. D_s/D^0, Λ_c/D^0, Λ_b/D^0) with multi-class classification algorithms. Not only important for HF analysis

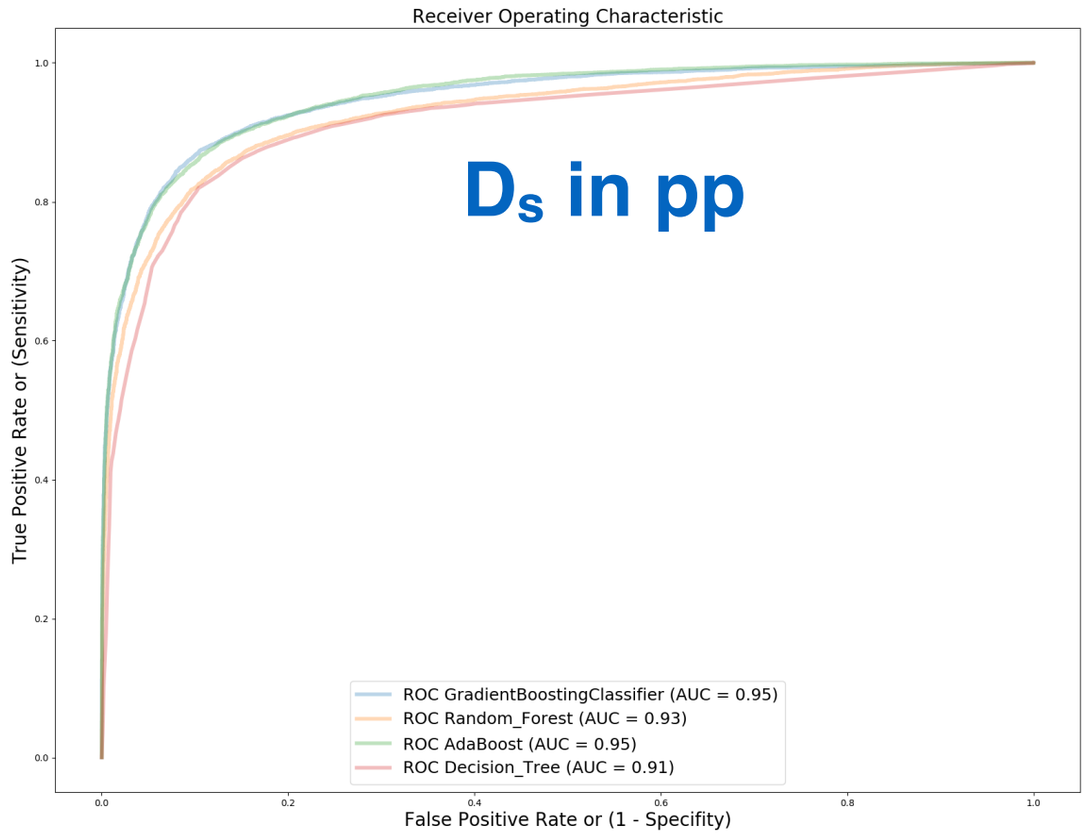
- Optimize selections to minimize $\sqrt{(\sigma_{stat}^2+\sigma_{syst}^2)}$. In many Run3 analyses, we will not be dominated by statistical uncertainties anymore!



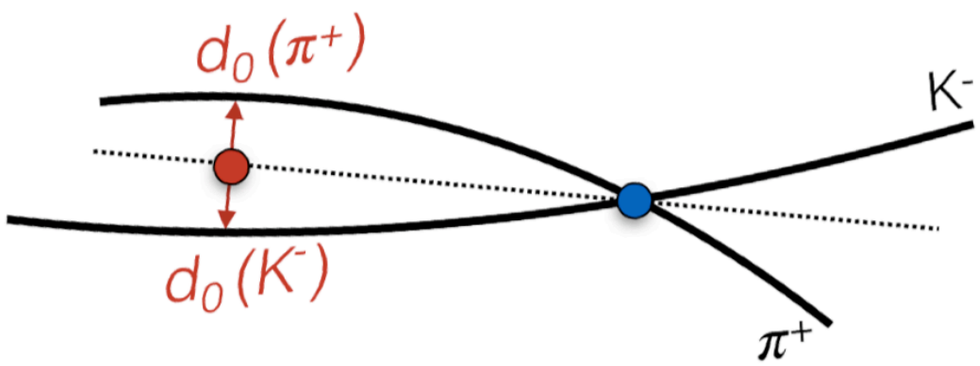
Classification of open HF signals

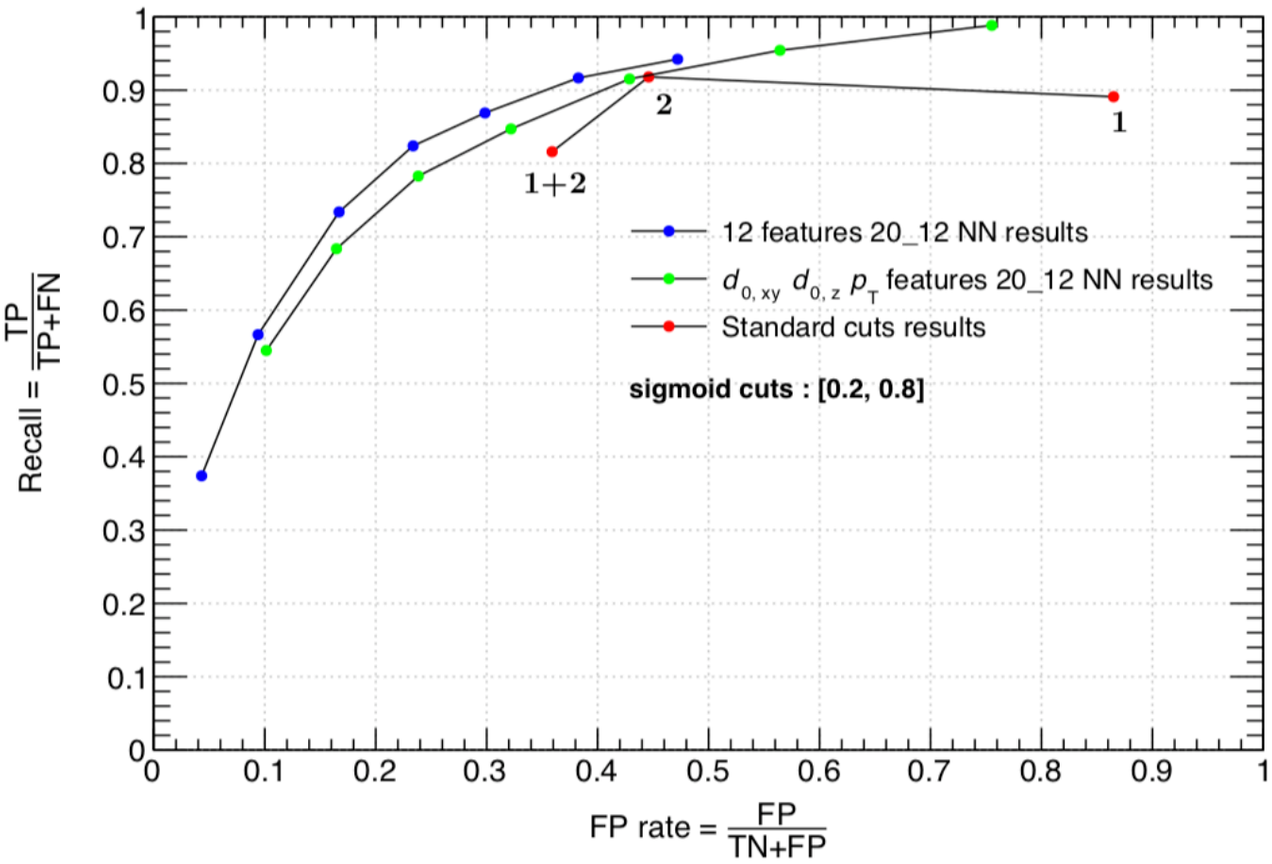
- Several analyses already ongoing….





DNN for HF reconstruction and selection





Deep Neural Networks can also be used to:

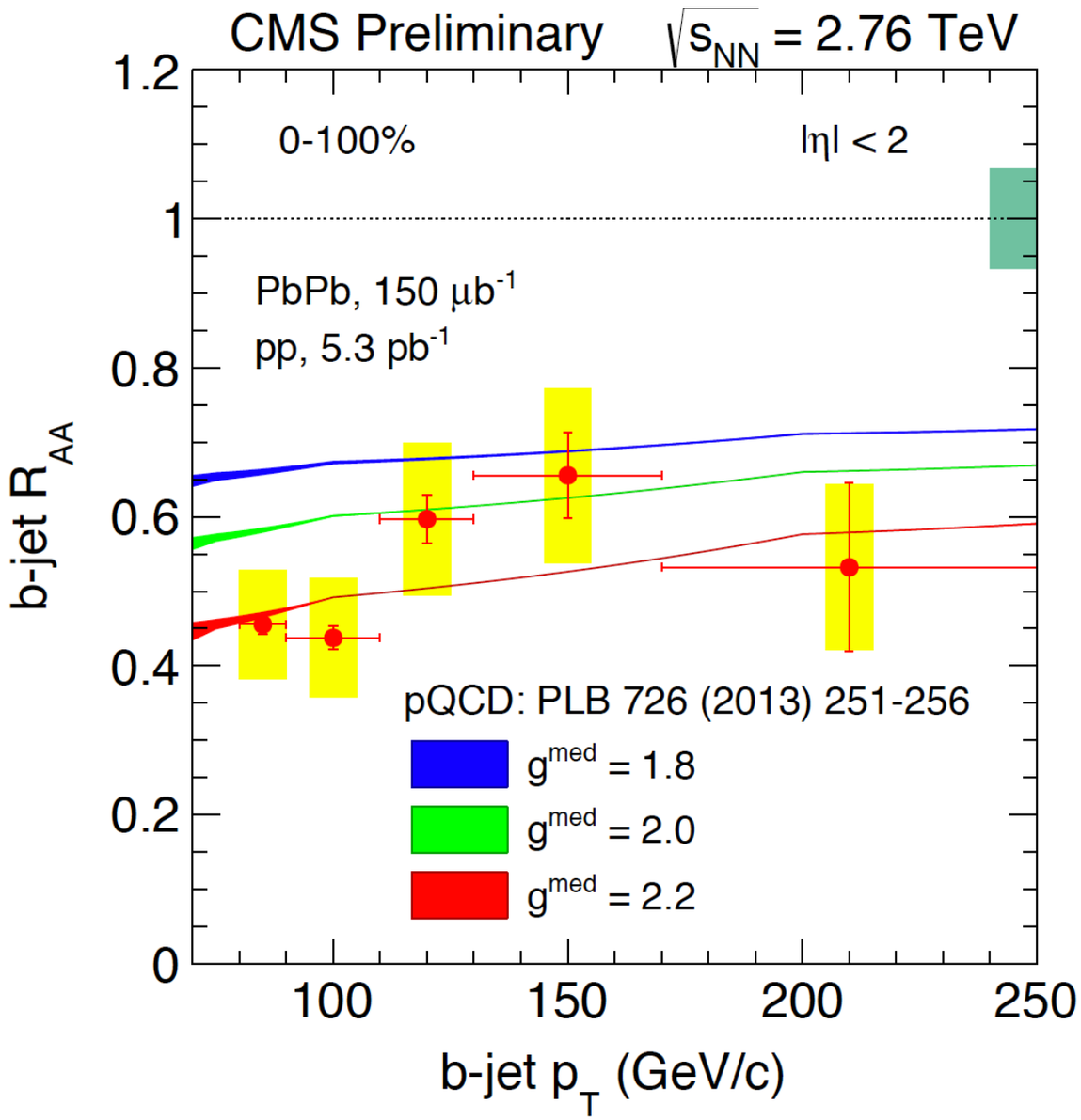
- preselect tracks (with track parameters, PID raw info, etc) to reduce the input size
- even perform secondary vertex reconstruction starting from raw list of track (more exploratory)

→ very useful to have in Run3 e.g. tagged dataset of MB event with high HF content to minimize the size of dataset for statistics-eager analysis like HF

Jet tagging

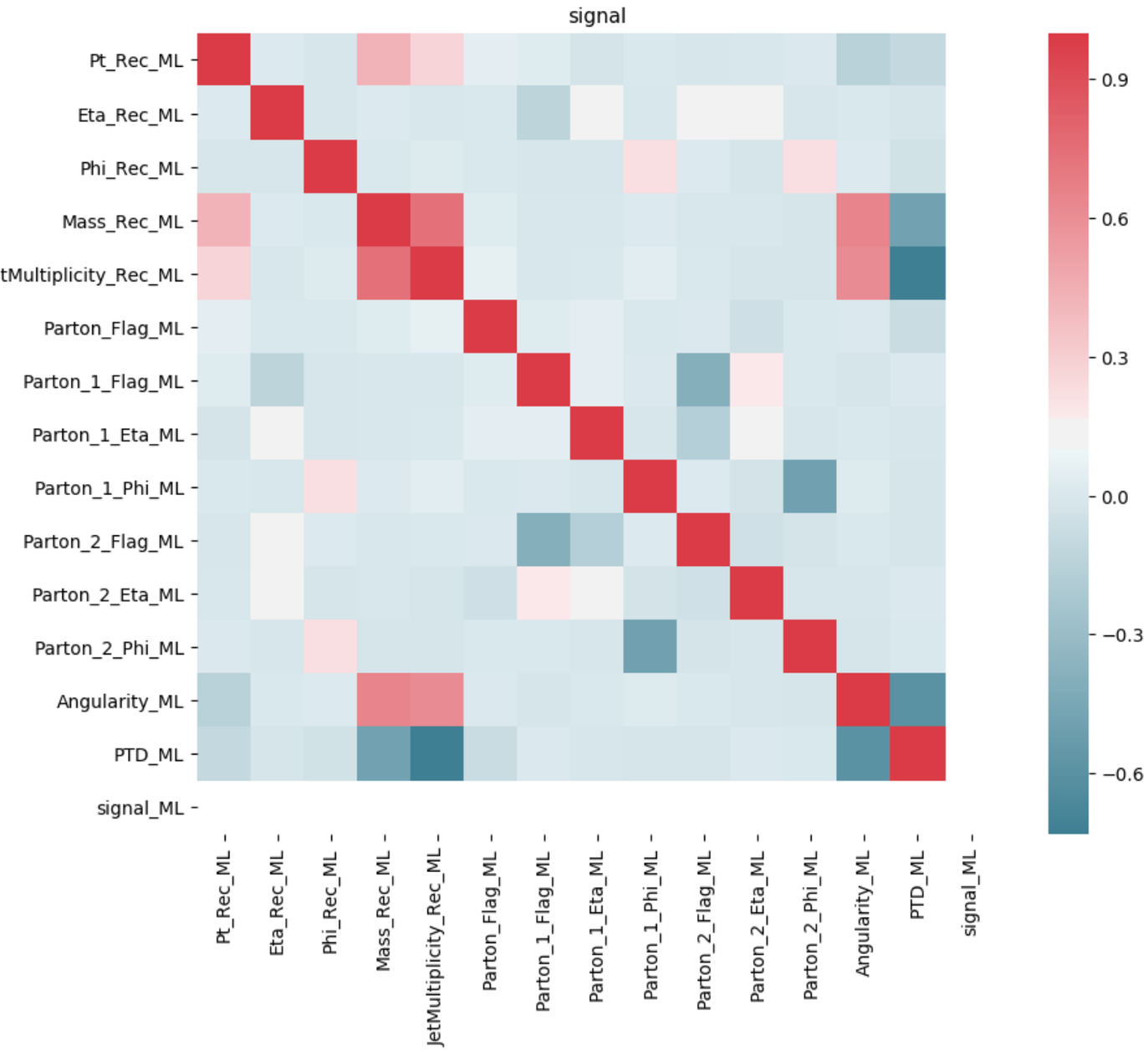
Low p_T b-jets in PbPb and HF jet substructure are of the most interesting probe for dead-cone and E_{loss}.

- p_T <40 GeV is the region where ALICE can be competitive since CMS/ATLAS are limited by trigger thresholds and JES
- New tagging techniques can be developed in parallel with more traditional CSV tagging algorithms
- Similar algorithms can be used for enhancing the fraction of gluon vs quark jet and studying parton dependence of E_{loss}



Quark vs gluon jet tagging

we started investigating the possibility of enhancing the fraction of quark jet by studying jet shape variables.

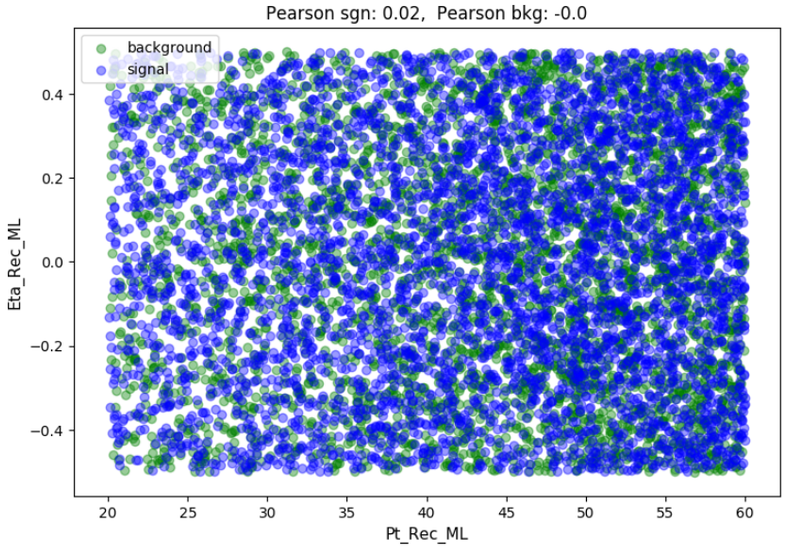


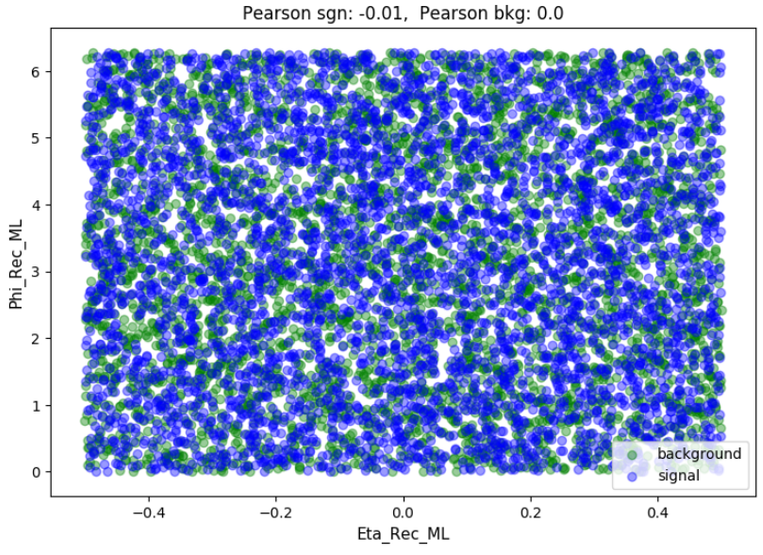
- The idea is to start with standard and DNN algorithm using engineered high-level variables (like angularity etc) and then move to more refined networks that can use track as inputs

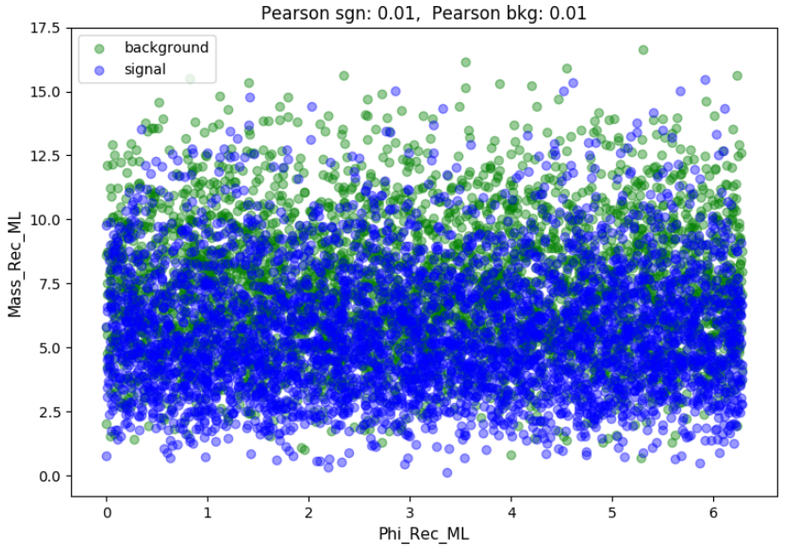
- first step to also develop c/b tagging techniques.

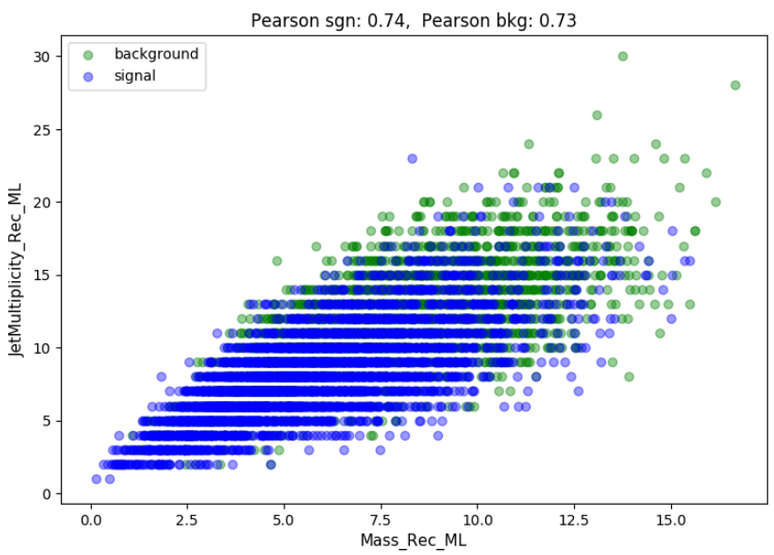
Quark vs gluon jet tagging

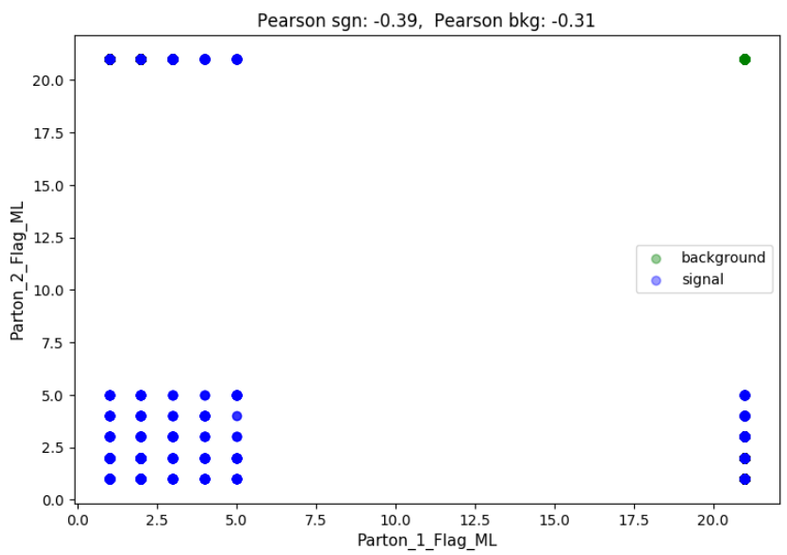
Studying distribution and 2D correlation of the most relevant variables is the starting point

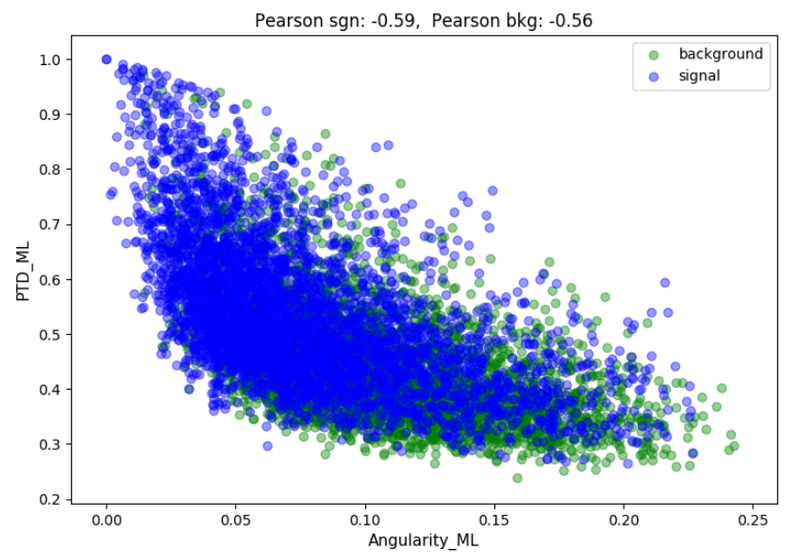










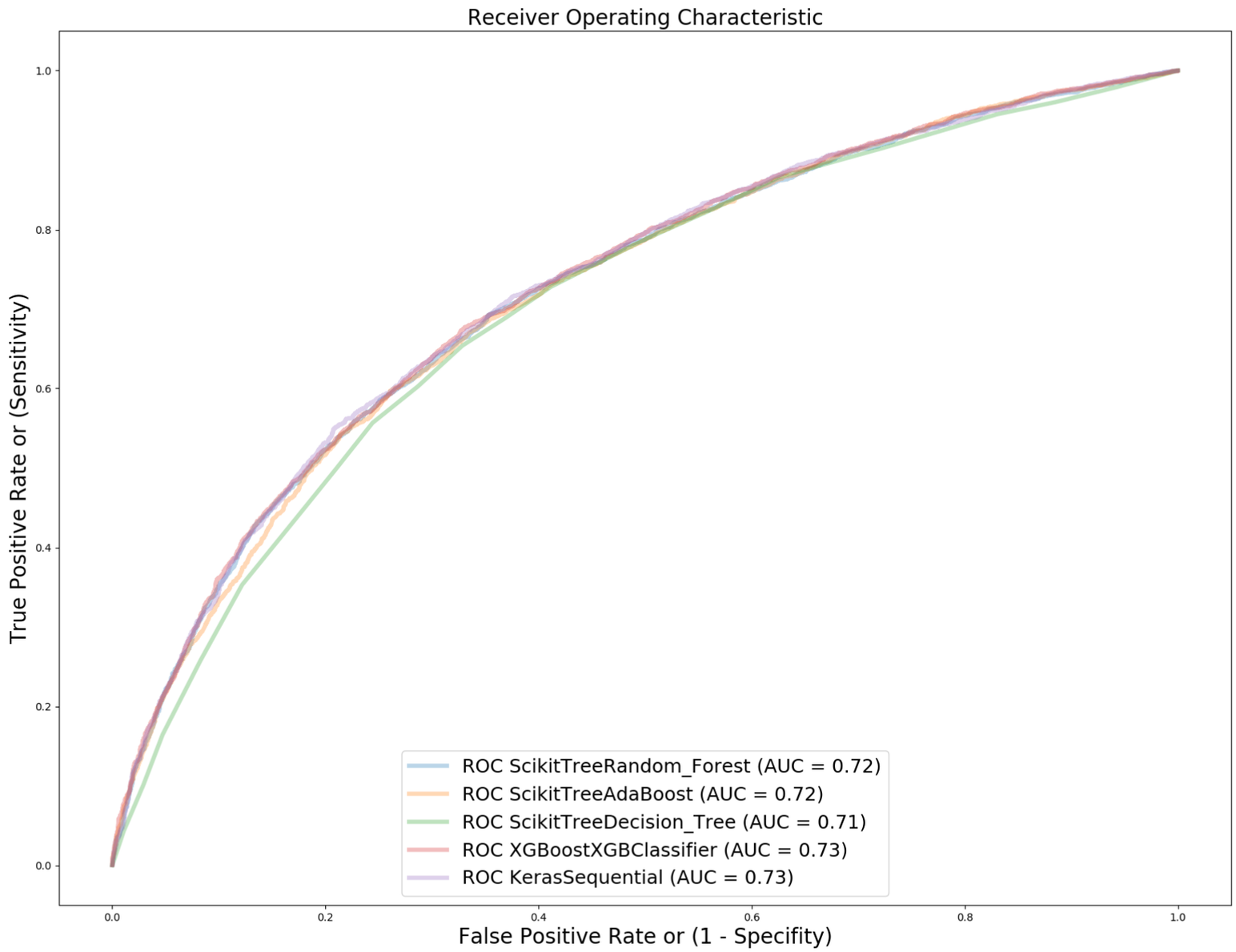


Signal defined as MC jets with parton PDG =1,2,3,4,5 (all quarks u,d,s,c,b,t)

Background defined as MC jets with all other PDGs

Quark vs gluon jet tagging

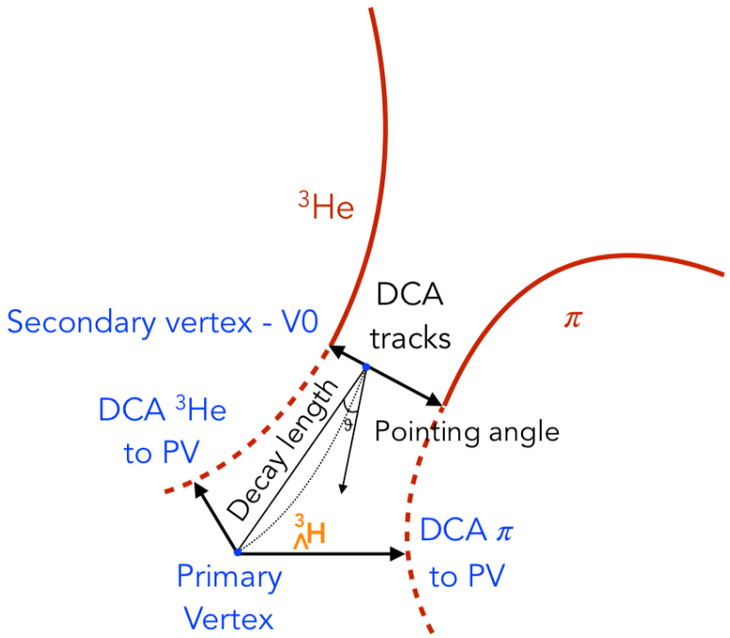
Studying distribution and 2D correlation of the most relevant variables is the starting point

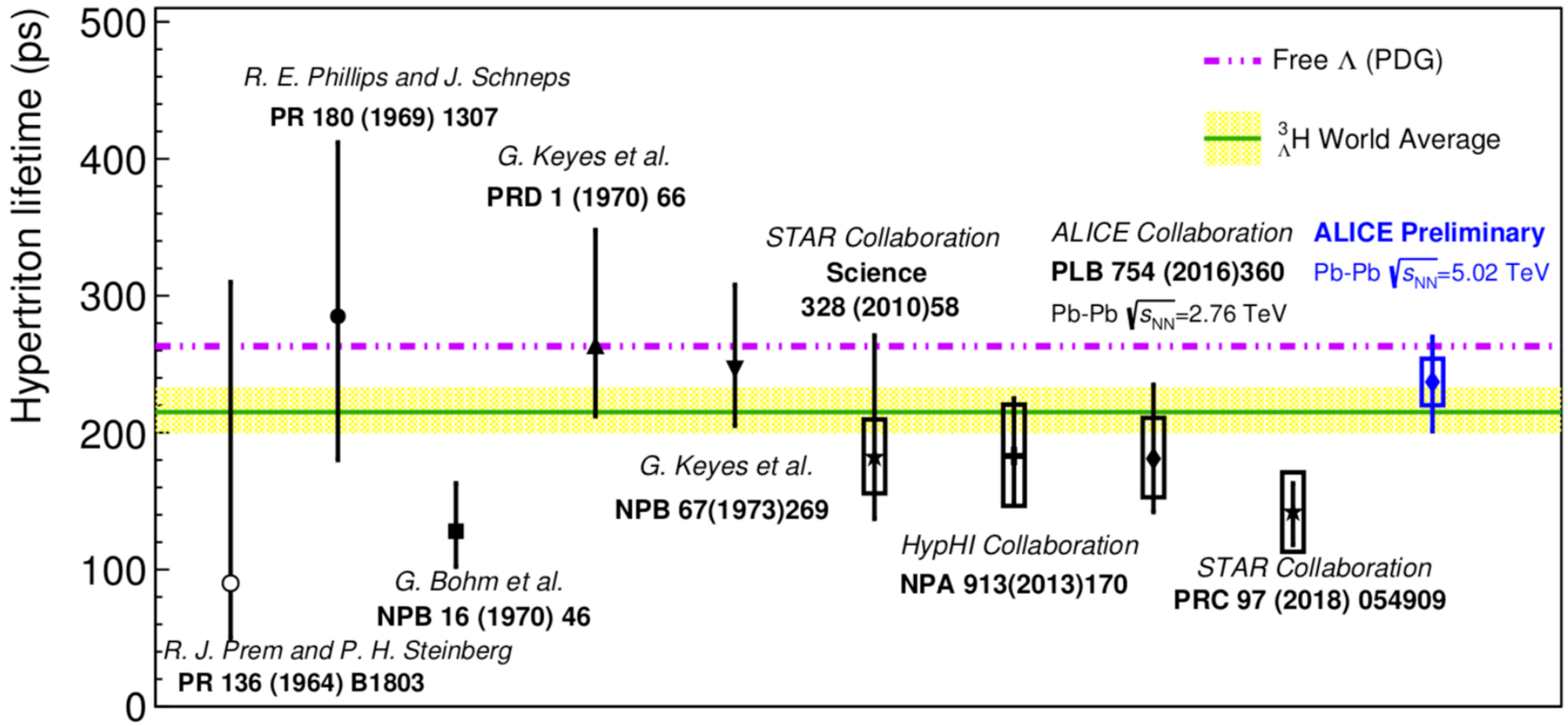


We have already interesting “tagging” results 😀

Nuclei and hypernuclei

- Optimise the selection strategy for (anti-)nuclei and (anti-)hypernuclei
- identify, account and correct for those produced in secondary processes

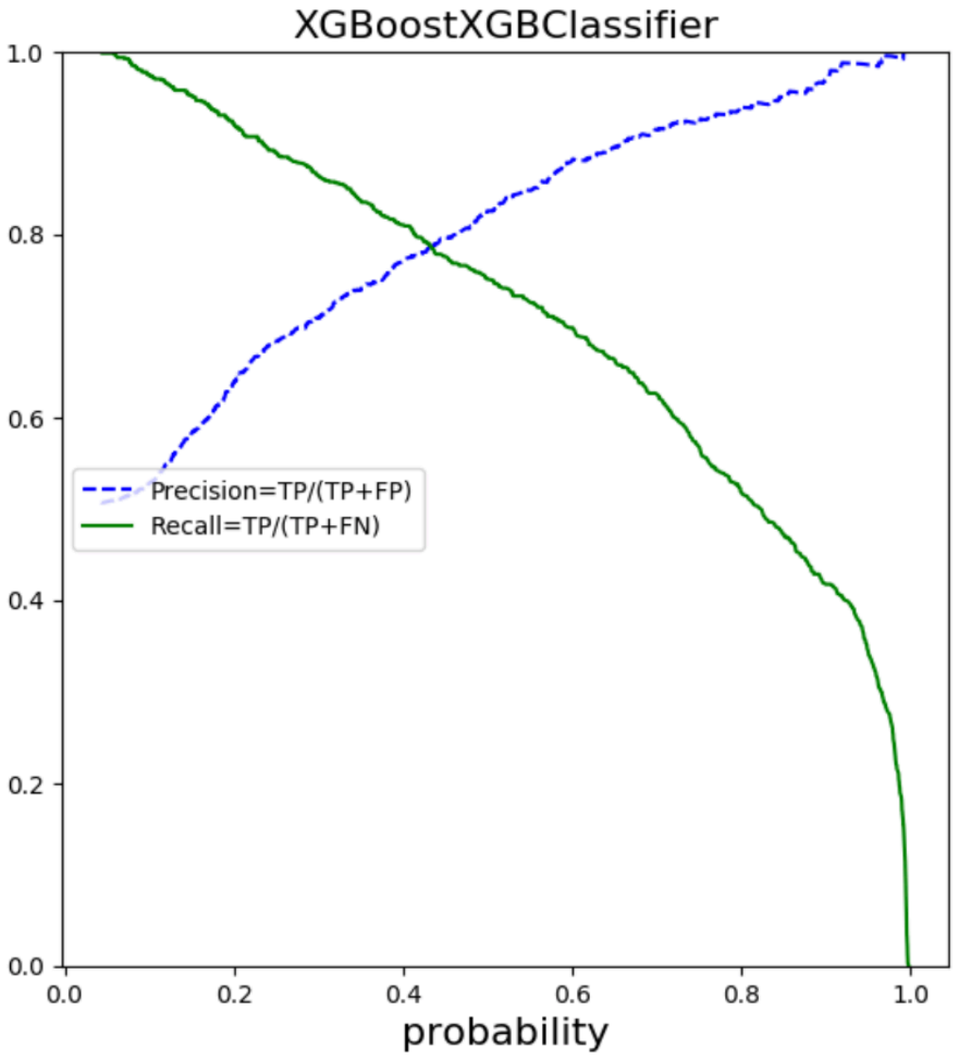


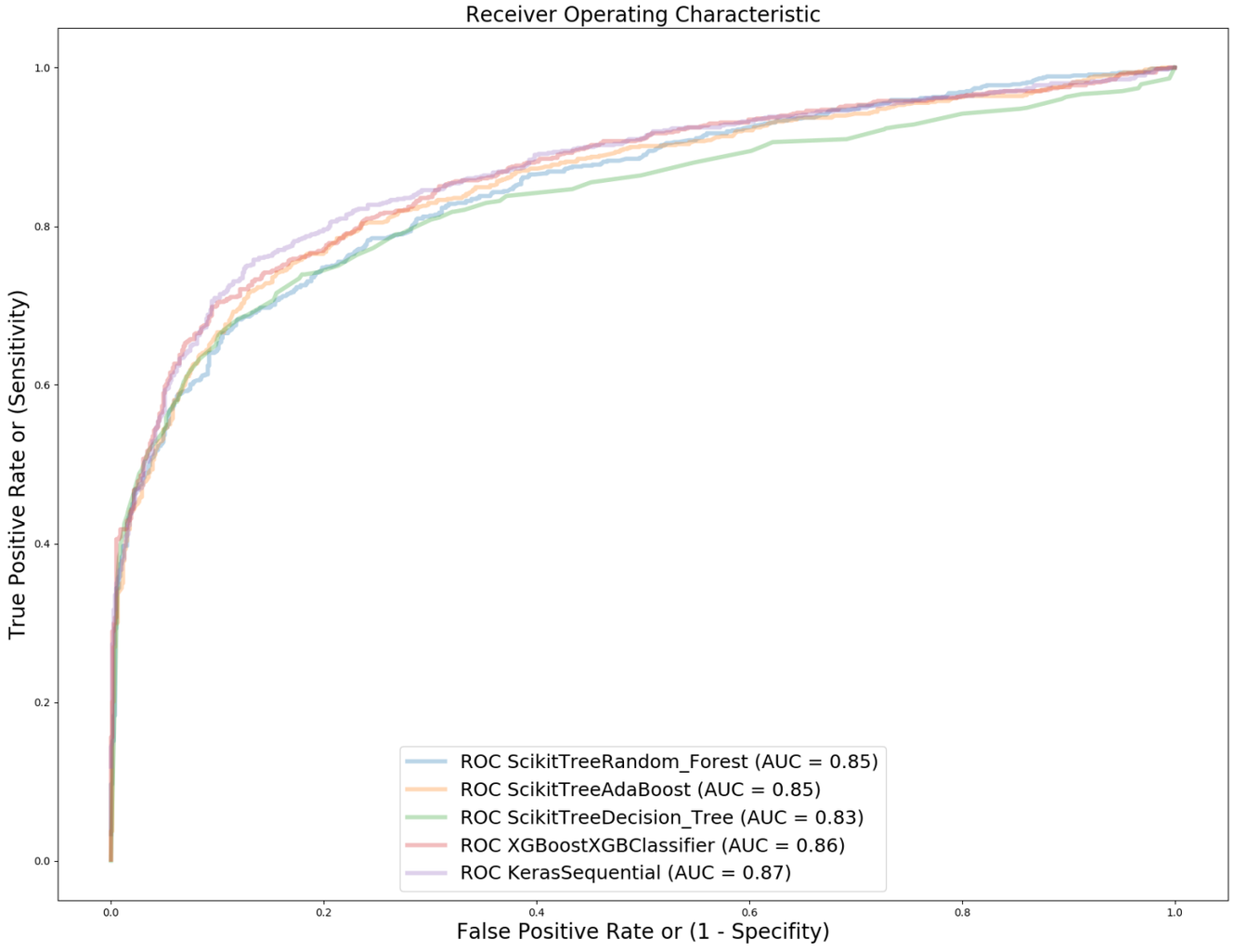


ALI-DER-161043

Hypertritium optimization

Improve the selection strategy of the Hyper-tritium in pp collisions



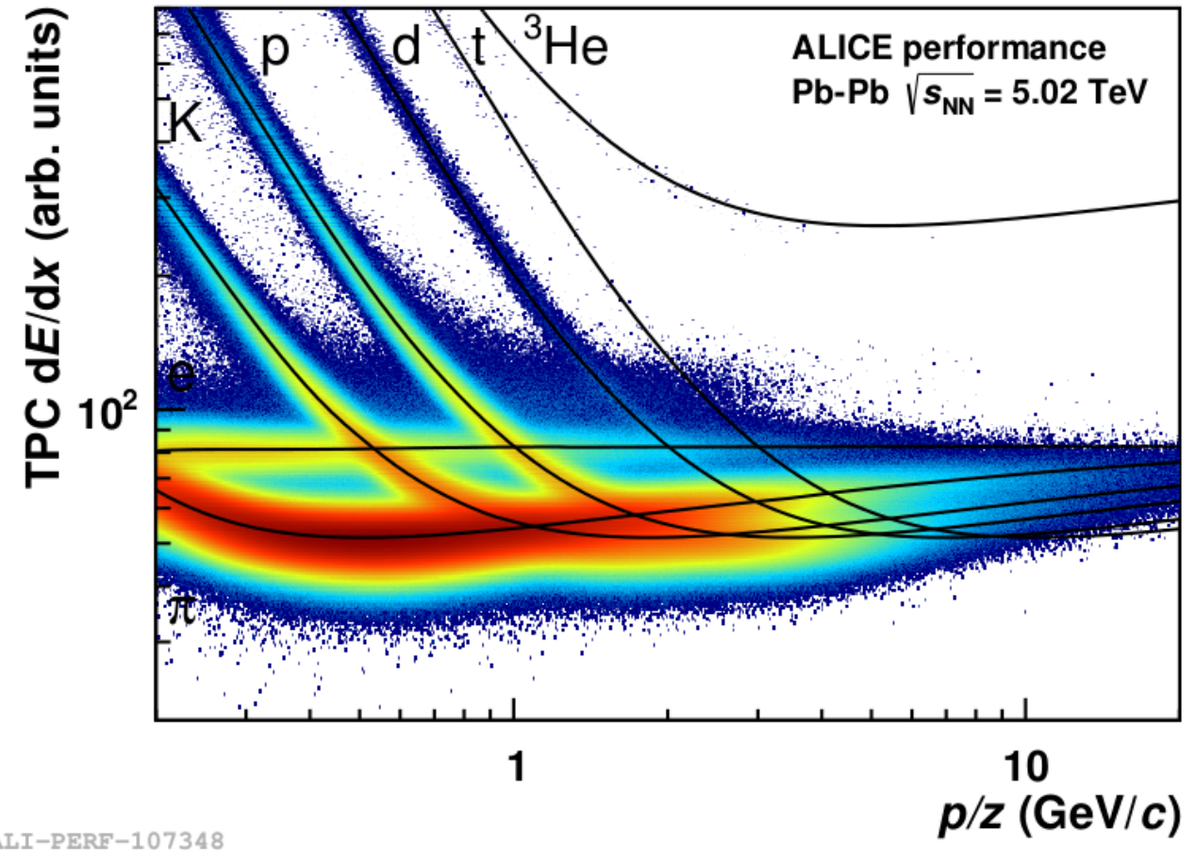


Optimization of the PID selection

PID selection is one of the first areas where ML was applied in HEP

Several classification methods and algorithms can be used. Main ideas:

- exploit correlation between track parameters and PID variables
- use raw PID info instead of engineered ones like nsigma



Optimization of the PID selection

PID selection is one of the first areas where ML was applied in HEP

E.g. list of selection variables:

- TPC, TOF signal
- track dca, sigma dca, chi2, number of ITS hits, TPC hits …

Work ongoing with TPC experts (Jens, Marian) to identify the right selection variables

Optimization of the PID selection

PID selection is one of the first areas where ML was applied in HEP

E.g. list of selection variables:

- TPC, TOF signal
- track dca, sigma dca, chi2, number of ITS hits, TPC hits …

Work ongoing with TPC experts (Jens, Marian) to identify the right selection variables

The elephant in the room….

- Having sizable Monte Carlo / Data differences is THE problem in having sophisticated PID selection
- the current tune-on-data “destroys” any correlation with track variables

https://indico.cern.ch/event/766087/contributions/3218852/attachments/1754285/2865669/MCdatareweight.pdf

Who is working with us?

- **B^+ in pp/PbPb:** L. Vermunt, L. Van Doremalen (Utrecht)
- **PID**: A. Kelweit, K. Lapidus (CERN), S. Hornung, A. Caliva, M. Ivanov (GSI)
- **D_s:** F. Grosa, F. Catalano (Torino)
- **Jets:** B. Trzeciak (Utrecht), N. Zardoshti (CERN), R. Haake (Yale)
- **Λ_c in pp:** J. Norman (CNRS), C. Hills (Liverpool)
- **Λ_c in PbPb:** C. Zampolli (Bologna, CERN), A. Alici (Bologna)
- **D***: G. Luparello (Trieste)
- **HF Flow** : L. Kreis, A. Dubla (GSI)
- **Electrons/MC-data reweighting**: A. Capon, S. Lehner (SMI)
- **Baryons**: M. Faggin (Padova)
- **Hypernuclei** : K. Lapidus (CERN), S. Hornung, A. Caliva, A. Mastroserio (Bari), S. Bufalino, M. Puccio, M. Masera, F. Mazzaschi, P. Fecchio

Group leaders involved:

A. Dainese, Peter Hristov, S. Masciocchi, A. Mischke, A. Morsch, F. Prino, M. Weber, M. Masera, S. Bufalino

And more joining almost every week…

Summary and outlook

- The effort is moving forward quickly!
- Lot of work was done so far to develop a working framework that is now being tested and improved.

- A new machine will be purchased in the coming days (CERN request already submitted)
  - large RAM
  - 1 NVIDIA TESLA V100
  - high performance CPU +SSD

- **The challenge is now to convert this technical effort into improved physics performances to make the best with the 2018 PbPb runs and with future Run3 data:**

→ new measurements using these techniques will be presented soon in the regular working group and PWG

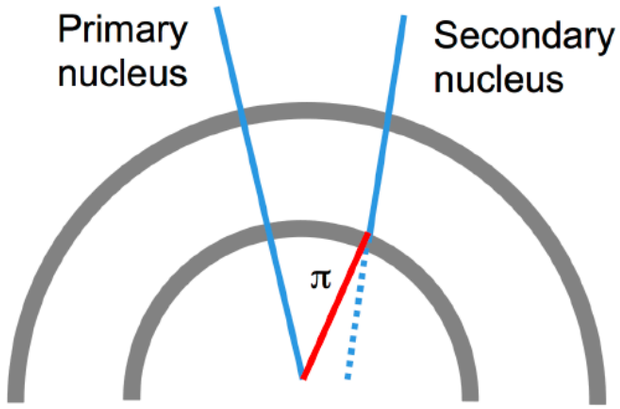
- **Interesting overlap with O2 activities where these techniques could be employed for fast MC productions, TPC distortion corrections, online HF tagging, online calibration…. to be discussed with O2 experts!**

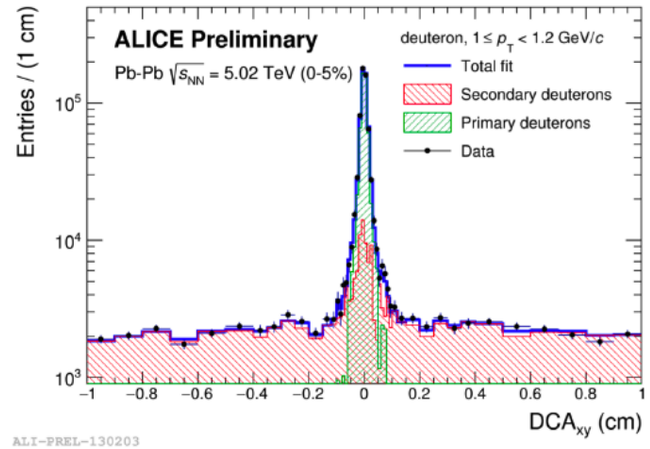
BACKUP

Reconstruction issues

- **Absorption** of anti-matter in detector material

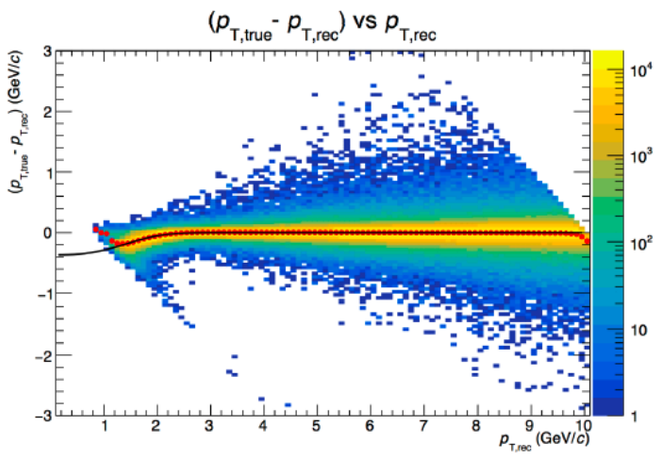
- **Secondary nuclei** emitted by spallation from the detector material
  - Impact parameter





- Considerable **energy loss** of the heavy particles in the detector, and lack of correction for it

  Energy loss corresponds to slowing down of the particle along the trajectory
- Z=2 not properly considered in the energy loss



→ Slide from Silvia Masciocchi, LEAP2018, Paris, March 15, 2018

A new common framework for Machine Learning in ALICE

A new common framework for ML

Proving flexible tool to perform Machine Learning analysis. It includes:

- **Common Ntuplizer for TTree creation that can run on the Grid using LEGO trains**

- **ROOT to Pandas DataFrame conversion:**
  - convert the root TTree of MC and Data into Pandas data frames
  - create training samples mixing MC and data
  - create testing and training samples

- **Training/Testing and common validation routines with Scikit/TensorFlow:**
  - implement most common ML algorithms and Deep Neural network for classification using SciKit and TensorFlow
  - Automatic validation with cross score validation, confusion matrix, learning curves, ROC, etc et

- **Testing on large samples for analysis and new TTree creation:**
  - new decision flag is added to the data frame
  - a new TTree is created including flags and probabilities of all the ML algorithm
  - Possibility of exporting the model in C++ for running testing on the Grid

Advantages

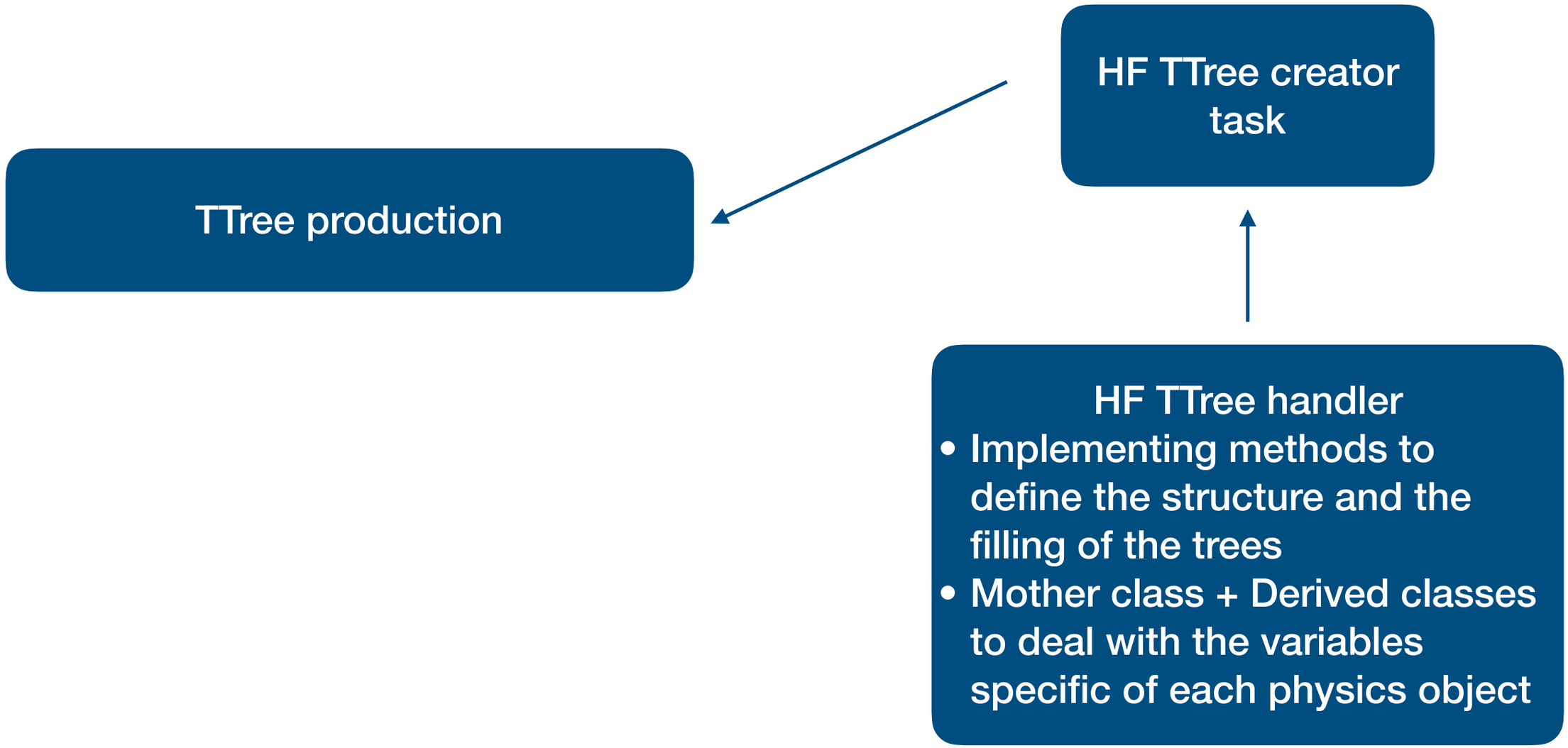
- **Ntuplizer for TTree creation that can run on the Grid using trains**
  - same data format for all the different analyses
  - possibility of producing several ML samples at the same time with grid

- **Training/Testing and common validation routines:**
  - same ML core software for all classifications analyses
  - automatic tools for validating the results

- **Testing on large samples for analysis and new TTree creation:**
  - all the analyses can still be performed with standard ROOT software
  - possibility of performing testing on large scale with Grid

Common TTree production

Effort led by Andrea F.

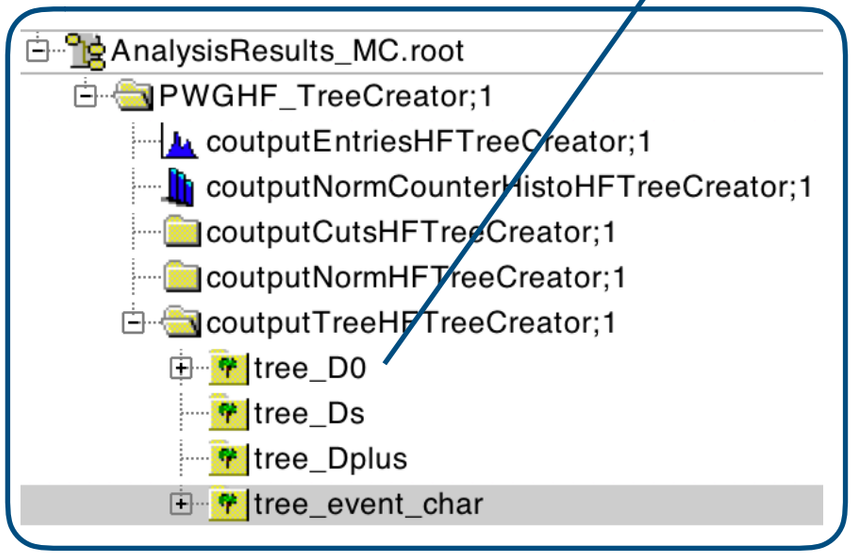


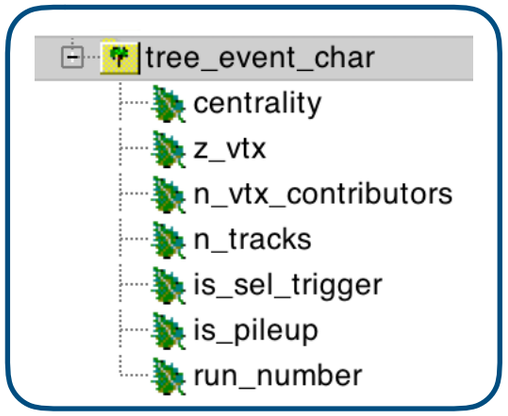
Common TTree production

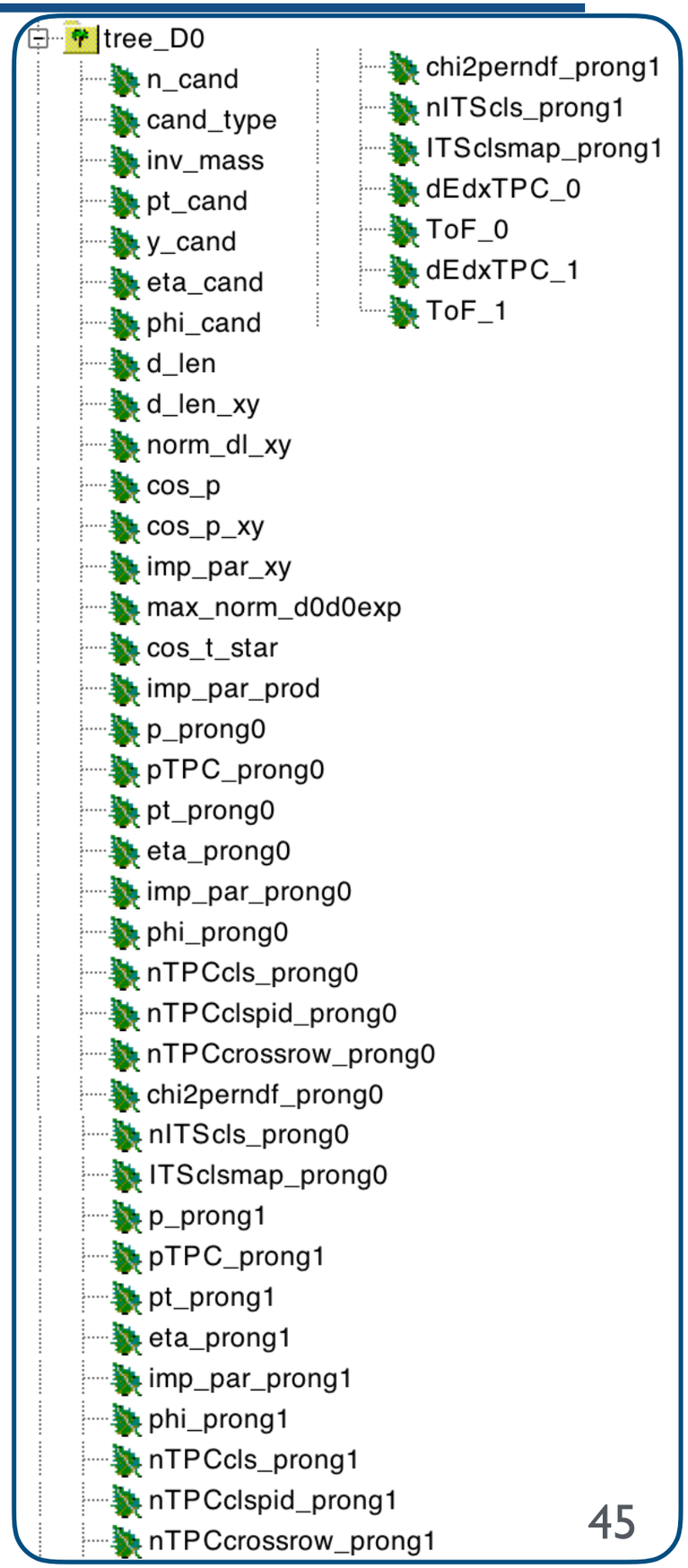
TTrees

- Tree for every physics object, e.g. D^0, filled event-by-event, containing
  - Single track variables
  - Object-related variables
  - Common structure among different objects

- Tree for event characterisation
  - Centrality
  - Z vertex
  - N. of vtx. contributors
  - Trigger selection
  - Pileup
  - Run number







Currently running on LEGO trains

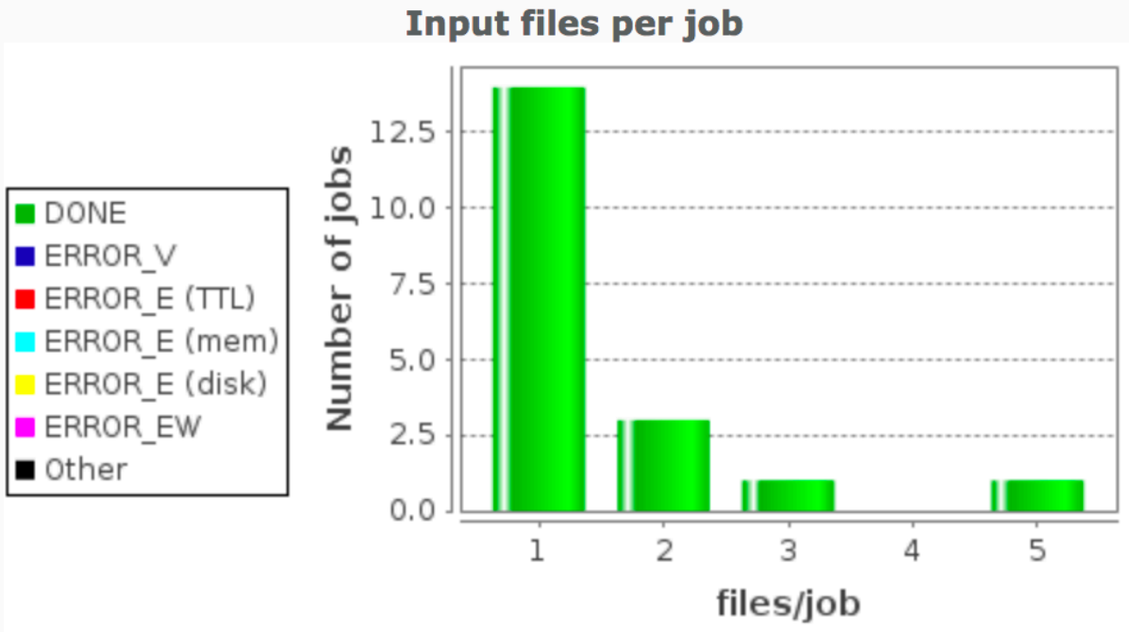
With Markus, Andrea and Jan-Fiete, we managed to run the ntuplisers with LEGO trains → no quota problems, larger storage available compared to private jobs

Train Run (PWG train overview)

Status	Running triggered on 08 Oct 2018 18:19 (23d 20:47 ago) **All jobs submitted, masterjobs submitted: 1, last run: 245148**
Files	Files copied to the Grid successfully | file copying log | train files in FC
Processing	processing progress **19 total, 19 done, 0 error, 0 active, 0 waiting** stack trace analysis
Merging	merging progress: **1 total, 0 done, 1 error, 0 active, 0 waiting** intermediate merging: stage1 (**0/0/0/0/0**) stage2 (**0/0/0/0/0**) stage3 (**0/0/0/0/0**) stage4 (**0/0/0/0/0**)
Final Merging	
Statistics	**Totals:** running time: 3:36 | output size: 752.2 MB **Files/job** (for done jobs): min: 1, max: 5, average: 1.5 , standard deviation: 1 **Running time/job** (for done jobs): min: 46s, max: 1:13, average: 11m 25s, standard deviation: 16m 28s, 95% done after 1:13

Job Overview

State	Jobs		Files		Input size	Files/job		
	#	%	#	%		min	max	avg
DONE	19	100.0%	28	100.0%	8.451 GB	1	5	1.5
ERROR_V	0	0.0%	0	0.0%	0 B	0	0	0
ERROR_E (TTL)	0	0.0%	0	0.0%	0 B	0	0	0
ERROR_E (mem)	0	0.0%	0	0.0%	0 B	0	0	0
ERROR_E (disk)	0	0.0%	0	0.0%	0 B	0	0	0
ERROR_EW	0	0.0%	0	0.0%	0 B	0	0	0
Other	0	0.0%	0	0.0%	0 B	0	0	0



What is included?

Several algorithms are already available:

- SVM
- Boosted Decision Trees and AdaBoost
- Random Forests
- Logistic regression
- Keras Deep Neural Networks
  - Sequential Network
  - Multi-Layer-Perceptron

Ready to perform:

1) Signal/Background classification for:

- D_s, D^0, Λ_c , B^+, Hypertritium..
- TPC/TOF PID
- jet tagging with high-level variables

2) Linear and non linear regression

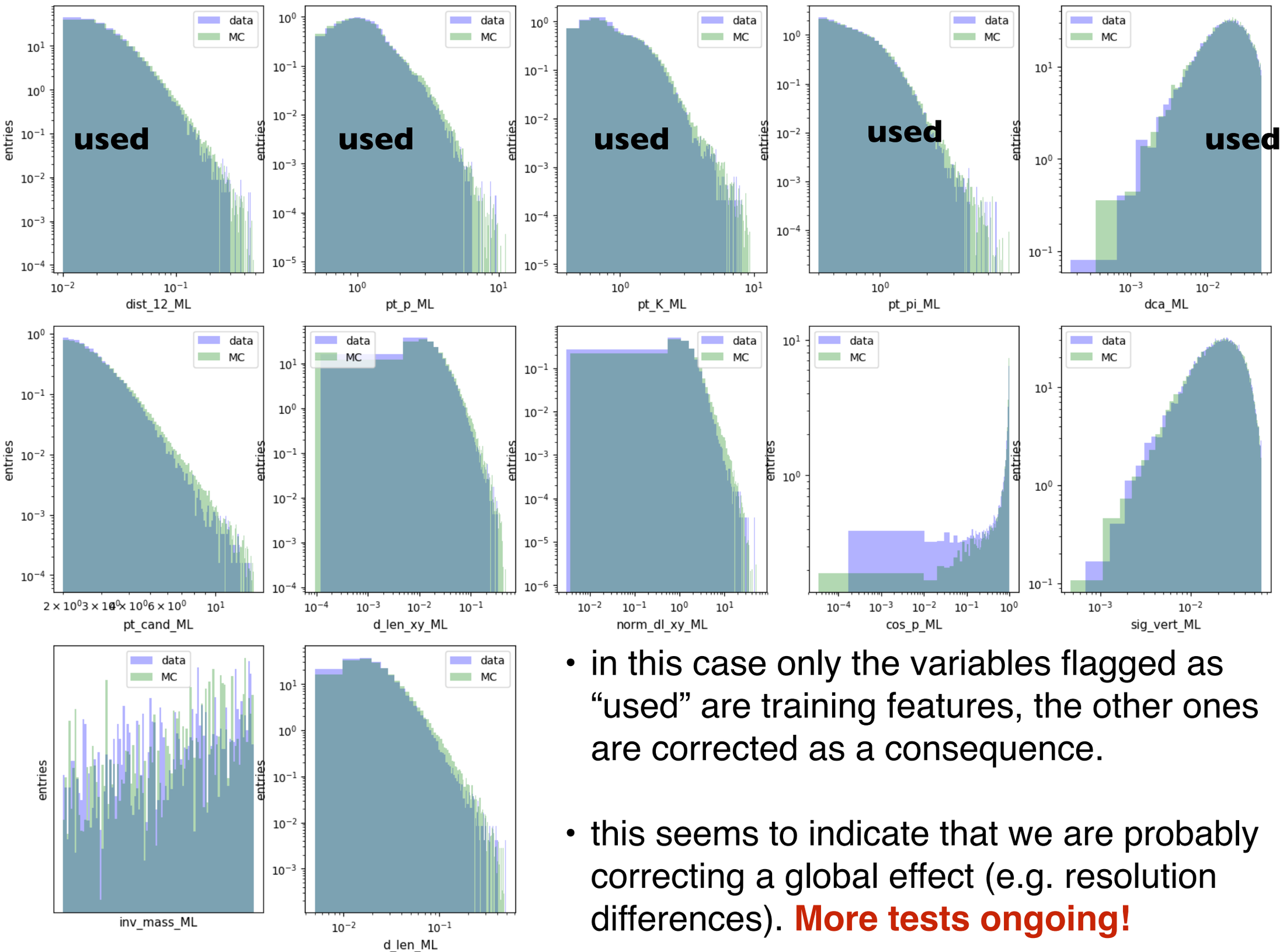
3) A new tool for MC-data reweighing (discussed later)

Algorithms available

Several algorithms are already available:

- SVM
- Boosted Decision Trees and AdaBoost
- Random Forests
- Logistic regression
- Keras Deep Neural Networks
  - Sequential Network
  - Multi-Layer-Perceptron

Can this become a more general tool?



- in this case only the variables flagged as “used” are training features, the other ones are corrected as a consequence.

- this seems to indicate that we are probably correcting a global effect (e.g. resolution differences). **More tests ongoing!**