Development of Machine Learning Tools in ROOT

(ACAT 2016 Valparaiso - Chile)

Omar Zapata (Metropolitan Institute Of Technology & University of Antioquia)

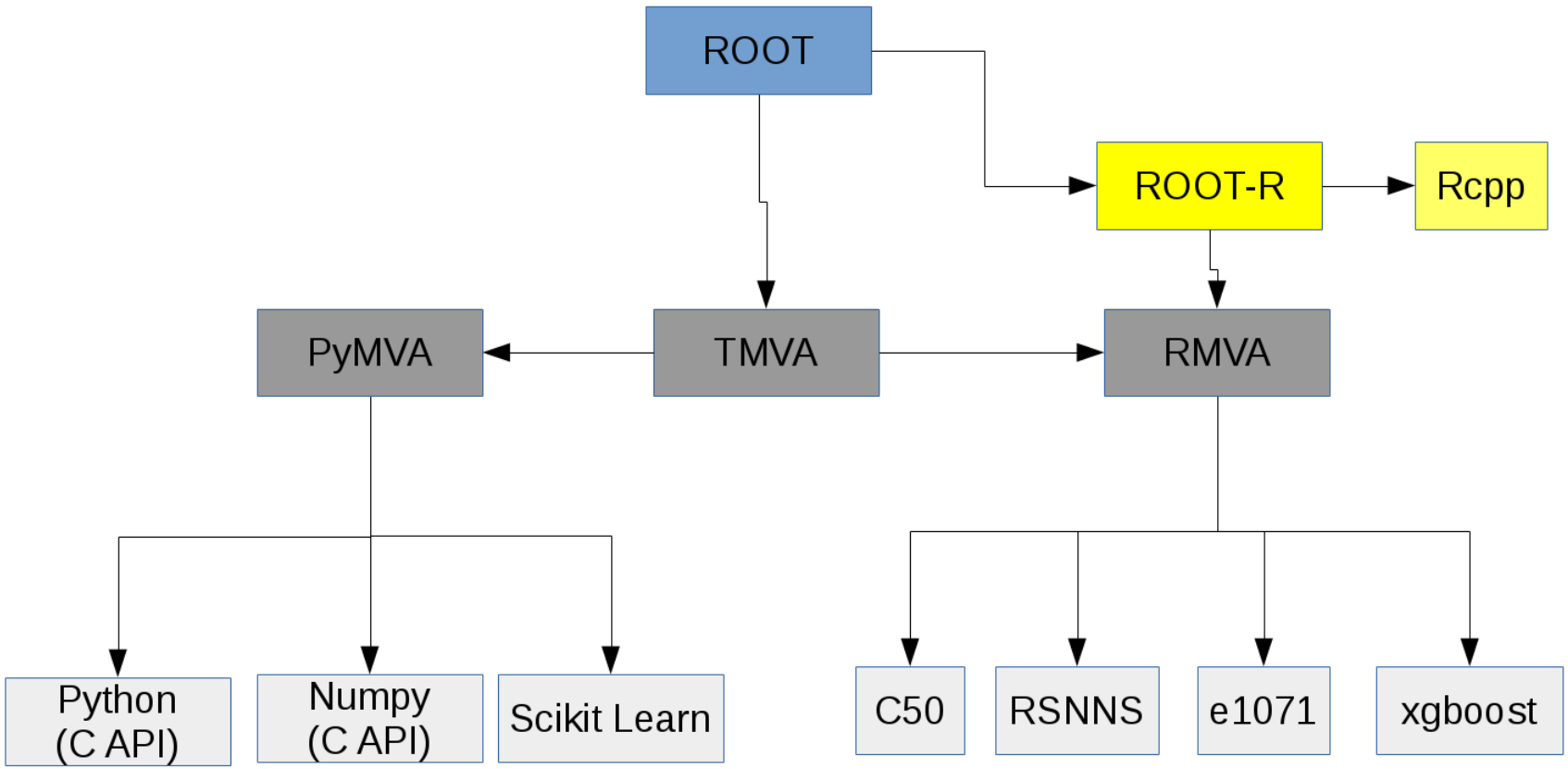
Lorenzo Moneta (CERN)

Sergei Gleyzer (University of Florida & CERN)

Outline

- Machine Learning Tools in ROOT
- New developments
  - ROOT-R
  - TMVA
    - Data Loaders
    - Variable Importance
    - ROOT-R with TMVA
    - Python TMVA (Scikits Learn)
- Incoming developments
- IML (Inter-Experimental LHC Machine Learning Working Group)

Machine Learning Tools in ROOT



What is R?

R is an open source language and environment for statistical computing and graphics.

- open source implementation of S language developed by J. Chambers
- R popularity and usage increased largely in recent years
- R provides a large variety of statistical tools
  - linear and nonlinear modelling
  - classical statistical tests
  - timeseries analysis
  - classification, clustering, …
- Environment is highly extensible with a large number of existing packages

ROOT-R Interface

ROOT-R is an interface between ROOT and R to call R functions using an R C++ interface (Rcpp, see http://dirk.eddelbuettel.com/code/rcpp.html)

- With ROOT-R you can perform a conversion from ROOT's C++ objects to R objects, transform the returned R objects into ROOT's C++ objects, and the R functionality can be used directly in ROOT.

- This interface opens the possibility to use in ROOT a very large set of mathematical and statistical tools provided by R.
- Presented at CHEP2015 (see https://indico.cern.ch/event/304944/contribution/474)

ROOT-R Example

R code

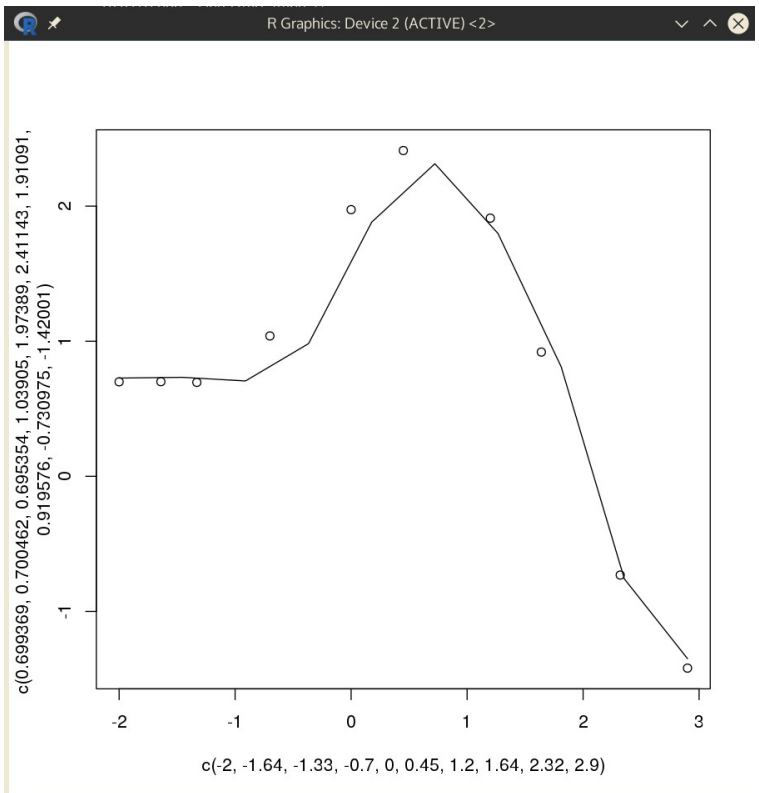
```
//original R code
xdata = c(-2,-1.64,-1.33,-0.7,0,0.45,1.2,1.64,2.32,2.9)
ydata = c(0.6993,0.7004,0.6953,1.039,1.973,2.411,1.910,0.9195,-0.7309,-1.420)
fit = nls(ydata ~ p1*cos(p2*xdata) + p2*sin(p1*xdata), start=list(p1=1,p2=0.2))
summary(fit)
confint(fit)
plot(xdata,ydata)
xgrid=seq(min(xdata),max(xdata),len=10))
lines(xgrid,predict(fit,xgrid))
```

ROOT/C++ code

```
//ROOTR C++ code
TRDataFrame data;
data["xdata"] = c(-2,-1.64,-1.33,-0.7,0,0.45,1.2,1.64,2.32,2.9);
data["ydata"] = c(0.6993,0.7004,0.6953,1.039,1.973,2.411,1.910,0.9195,-0.7309,-1.420);
TRObject fit = nls(asformula("ydata ~ p1*cos(p2*xdata) + p2*sin(p1*xdata)"),\
                           Label["data"]=data, \
                           Label["start"]=list(Label["p1"]=1,Label["p2"]=0.2));
summary(fit);
confint(fit);
plot(data["xdata"],data["ydata"]);
TRObject xgrid=seq(min(data["xdata"]),max(data["xdata"]),Label["len"]=10);
lines(xgrid,predict(fit,xgrid));
```

ROOT-R Example

Plot



Console Output

```
[omazapa] [tuxito] [~/ROOT/Workshop]$ root -l example.C
root [0]
Processing example.C...

Formula: ydata ~ p1 * cos(p2 * xdata) + p2 * sin(p1 * x
data)

Parameters:
   Estimate Std. Error t value Pr(>|t|)
p1 1.881851   0.027430   68.61 2.27e-12 ***
p2 0.700230   0.009153   76.51 9.50e-13 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.
1 ' ' 1

Residual standard error: 0.08202 on 8 degrees of freedo
m

Number of iterations to convergence: 7
Achieved convergence tolerance: 2.189e-06

Waiting for profiling to be done...
root [1]
```

(TMVA)Toolkit for Multivariate Analysis

TMVA is ROOT's toolkit for multivariate data analysis that implements machine learning algorithms for classification and regression.

It has a good set of algorithms widely used in HEP:

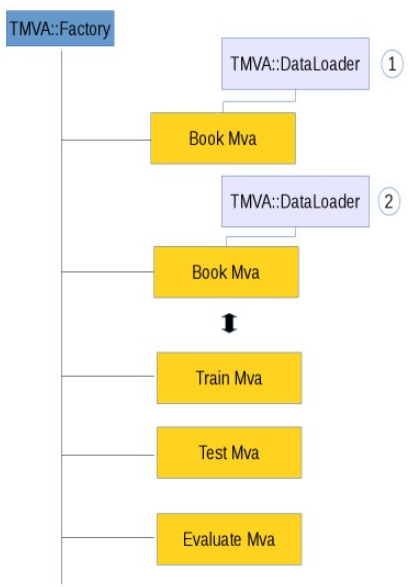
- BDT(Boosted Decision Trees)
- ANN(Artificial Neural Networks)
- SVM(Support Vector Machine)
- k-NN(k-Nearest Neighbour)
- LDA(Linear discriminant analysis)
- And more..

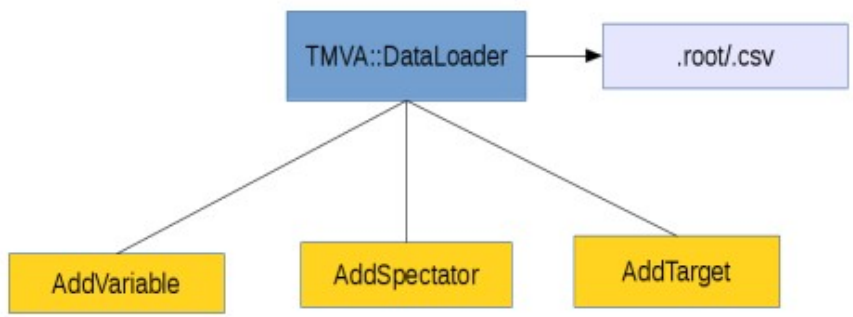
New Developments

- New features implemented in TMVA:
  - DataLoader
  - Variable Importance
  - R TMVA interface
  - Python TMVA interface

DataLoader

- TMVA DataLoader is a class that allows greater modularity and flexibility for training different classifiers with different features.
  - Each classifier can have different variables and/or data






```
factory->BookMethod( DataLoader &,Types::EMVA ,  TString methodTitle, TString theOption);
```

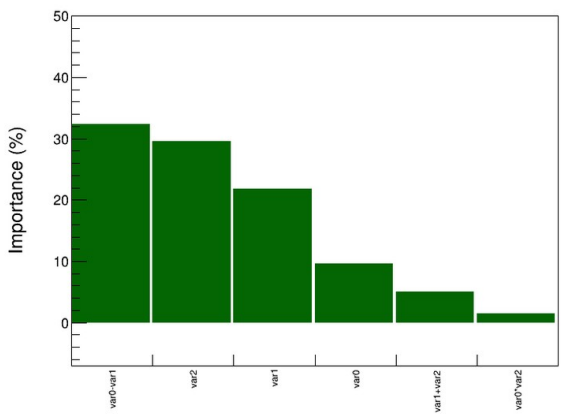
Variable Importance

- Algorithm ranks the importance of a variable in classfication processes.
- The method take a subset of variables and compute weights.

$$FI(X_i) = \sum_{S \subseteq V : X_i \in S} F(S) \times W_{X_i}(S)$$

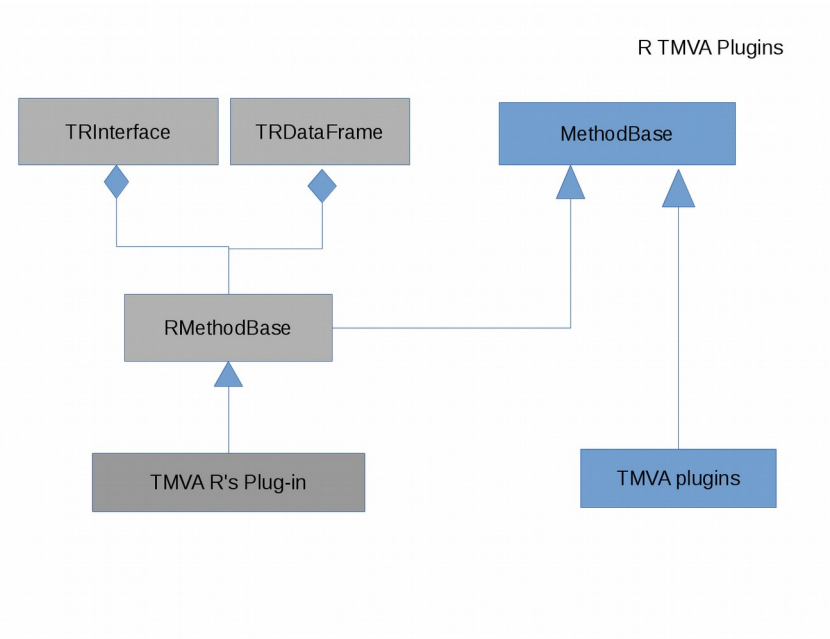
$$W_{X_i}(S) \equiv 1 - \frac{F(S - \{X_i\})}{F(S)}$$

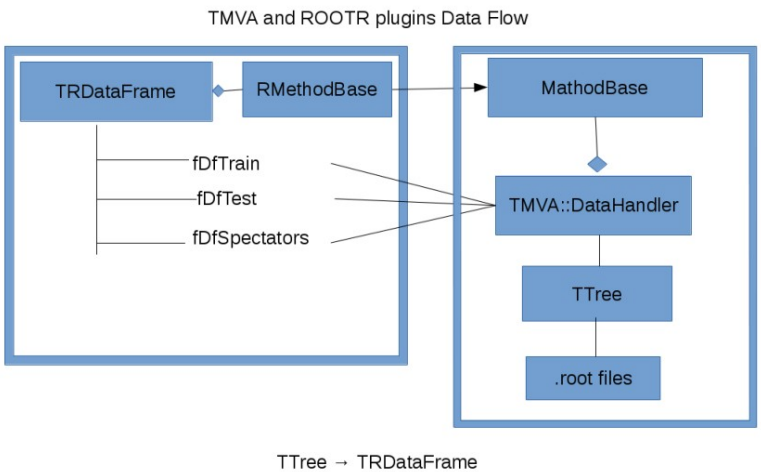
  - Full variable set {V}
  - Variable subset {S}
  - Classifier Performance F(S)
- It is independent of the classifier.
- Is a stochastic algorithm.
- Paper at http://pos.sissa.it/archive/conferences/070/067/ACAT08_067.pdf



RMVA (R TMVA)

RMVA is a set of plugins for TMVA package based on ROOTR that allows new methods of classification and regression calling R's packages.

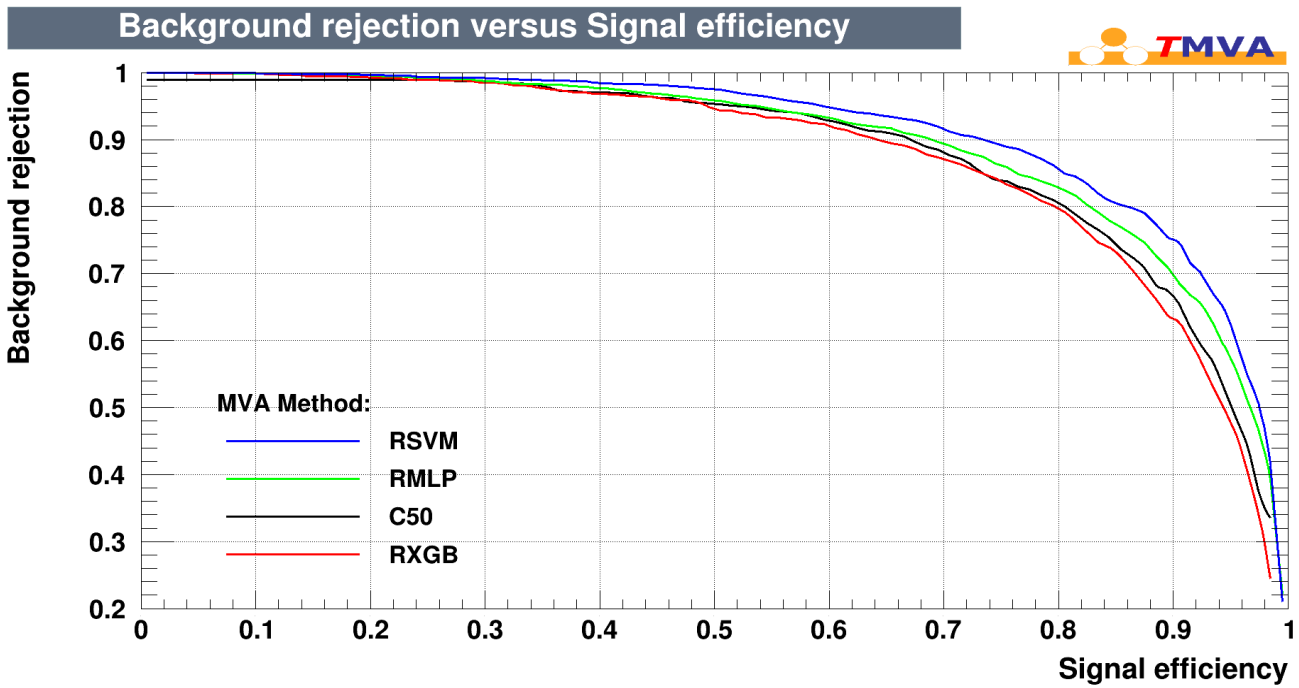




RMVA (R TMVA)

- **(C50) C5.0** decision trees and rule-based models for pattern recognition.
- **(RSNNS)**Neural Networks in R using the Stuttgart Neural Network Simulator (SNNS)
- **(e1071)** Support Vector Machine can be used to carry out general regression and classification (of nu and epsilon-type), as well as density-estimation. A formula interface is provided.
- **eXtreme Gradient Boost** (R package xgboost) An optimized general purpose gradient boosting library.
  - It implements machine learning algorithms under the Gradient Boosting framework, including Generalized Linear Model (GLM) and Gradient Boosted Decision Trees (GBDT).

RMVA (R TMVA)

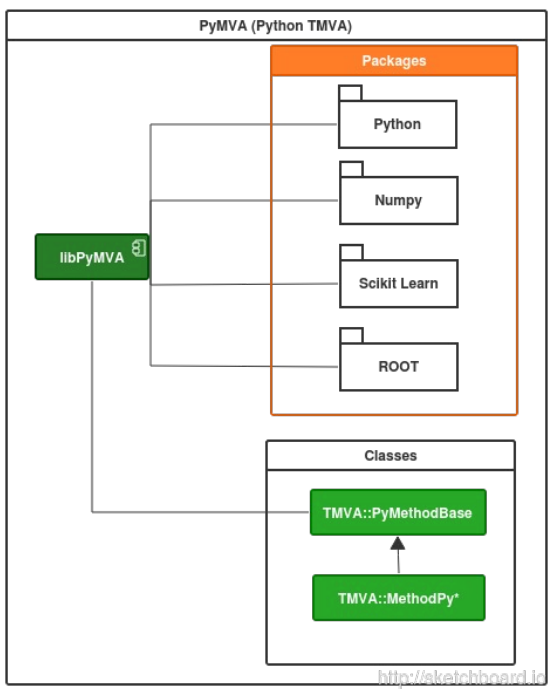


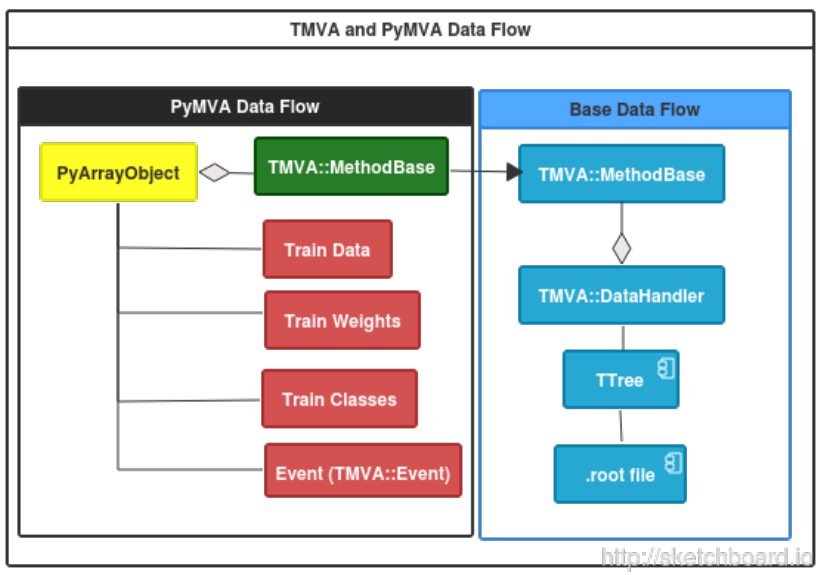
Evaluation results ranked by best signal efficiency and purity (area)

MVA Method:	Signal efficiency at bkg eff.(error): @B=0.01	@B=0.10	@B=0.30	ROC-integ.	Separation:	Significance:
RSVM	0.328(08)	0.735(08)	0.924(04)	0.913	0.526	1.355
RMLP	0.286(08)	0.689(08)	0.899(05)	0.897	0.481	1.310
C50	0.000(00)	0.671(08)	0.878(05)	0.881	0.462	1.253
RXGB	0.233(07)	0.643(08)	0.867(06)	0.875	0.434	1.194

PyMVA (Python TMVA)

PyMVA is a set of TMVA plugins based on Python API that allows new methods of classification and regression calling Python's packages.



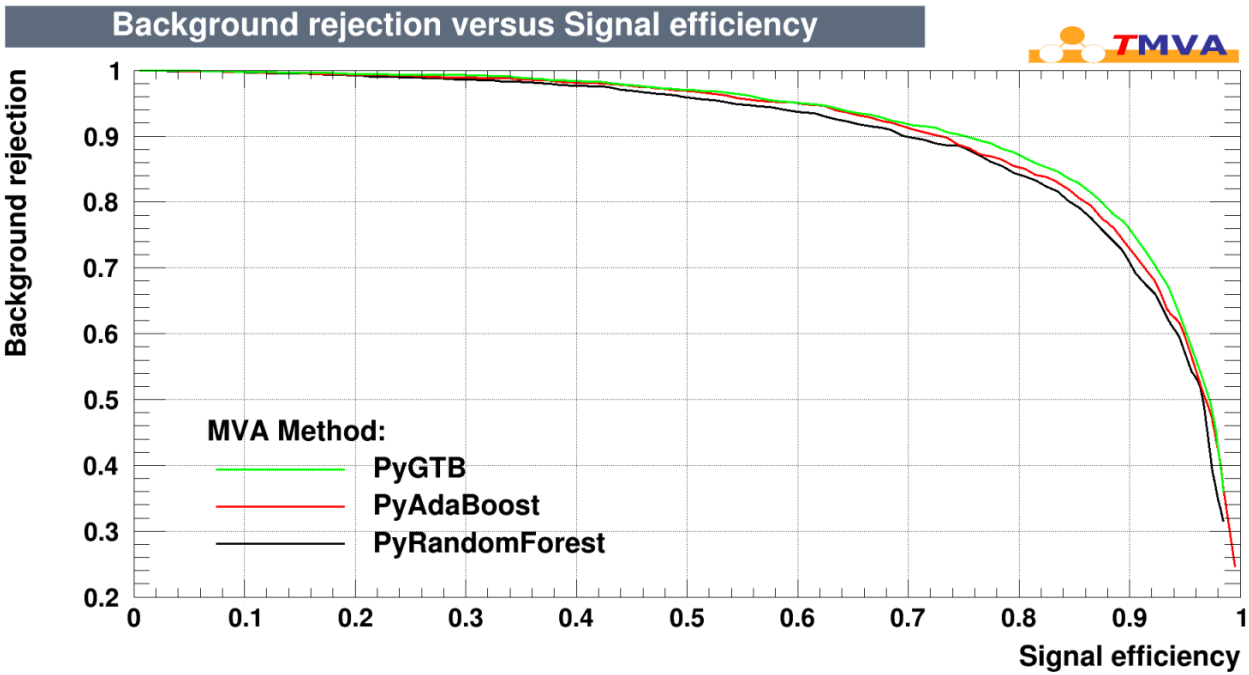


PyMVA (Python TMVA)

Implemented Methods Based on Scikit-learn package

- **PyRandomForest:** in random forests each tree in the ensemble is built from a sample drawn with replacement (i.e., a bootstrap sample) from the training set.
- **PyGTB (Gradient Trees Boosted) :** Gradient Tree Boosting or Gradient Boosted Regression Trees (GBRT) is a generalization of boosting to arbitrary differentiable loss functions. GBRT is an accurate and effective off-the-shelf procedure that can be used for both regression and classification problems.
- **PyAdaBoost (Adaptative Boosting):**The core principle of AdaBoost is to fit a sequence of weak learners (i.e., models that are only slightly better than random guessing, such as small decision trees) on repeatedly modified versions of the data.

PyMVA (Python TMVA)




```
: Evaluation results ranked by best signal efficiency and purity (area)
: ------------------------------------------------------------------------------
: MVA              Signal efficiency at bkg eff.(error):       | Sepa-    Signifi-
: Method:          @B=0.01    @B=0.10    @B=0.30    ROC-integ. | ration:  cance:
: ------------------------------------------------------------------------------
: PyGTB          : 0.343(08)  0.751(07)  0.924(04)    0.914    | 0.539    1.514
: PyAdaBoost     : 0.331(08)  0.741(07)  0.918(05)    0.911    | 0.761    0.943
: PyRandomForest : 0.245(07)  0.702(08)  0.905(05)    0.898    | 0.497    1.375
: ------------------------------------------------------------------------------
:
```

Incoming Developments

- New TMVA::Factory design that allows more flexiblility to integrate new techniques for:
  - Cross validation
  - Data storage and manipulation (HDF5, CSV, JSON, Custom Serialization FIles and SQL)
- Improved design that will allow to use threads and new parallelization technologies like OpenMP, MPI, cuda and TBB.
- New deep learning NN (from Peter Speckmayer)
- Improved Support Vector Machine (from Thomas Stevenson)
  - see next presentation
- Additional deep learning plugins(deep belief nets and restricted boltzmann machines)
- Gaussian Processes

IML

- The Inter-experimental LHC Machine Learning (IML) Working Group is focused on the development of modern state-of-the art machine learning methods,
- Provide software solutions and training beneficial for all LHC experiments
- Forum where on-going work and relevant issues are discussed by the community.

IML

Website: http://iml.cern.ch

Email: lhc-machinelearning-wg@NOSPAMcern.ch

Group:https://groups.cern.ch/group/lhc-machinelearning-wg/default.aspx

Jupyter Notebook

- ROOT notebook example using DataLoader and R/ Python methods in TMVA

jupyter TMVA Last Checkpoint: Last Sunday at 10:14 AM (autosaved)

Control Panel Logout

File Edit View Insert Cell Kernel Help ROOT C++

Markdown Cell Toolbar: None

ROOT
Data Analysis Framework

TMVA

Required header files

In [1]:
```
#include "TFile.h"
#include "TTree.h"
#include "TString.h"

#include "TMVA/Factory.h"
#include "TMVA/Tools.h"
#include "TMVA/DataLoader.h"

#include "TMVA/MethodRXGB.h"
#include "TMVA/MethodC50.h"

#include "TMVA/MethodPyAdaBoost.h"
#include "TMVA/MethodPyGTB.h"
#include "TMVA/MethodPyRandomForest.h"
```

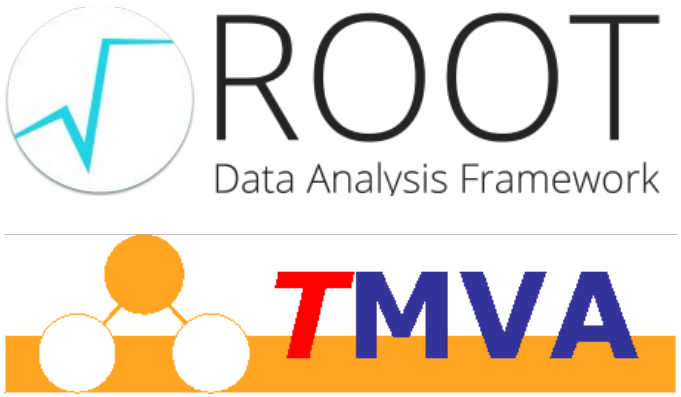
Declare Factory and DataLoaders

In [2]:
```
TMVA::Tools::Instance();

TString outfileName( "TMVAOutputDL.root" );
TFile* outputFile = TFile::Open( outfileName, "RECREATE" );

TMVA::Factory *factory = new TMVA::Factory("TMVAClassification", outputFile,
                                           "!V:ROC:!Correlations:Silent:Color:!DrawProgressBar:AnalysisType=Classification" );

TMVA::DataLoader *loader1=new TMVA::DataLoader("mymc-config1");
```

Summary

- New Machine Learning methods are available in latest ROOT release
  - Integrated methods from R and Python inside TMVA
  - New flexible design (DataLoader) and new tools (Variable Importance)
- Continue developments by improving TMVA (e.g. use parallelization) and by integrating new methods (deep learning, new SVM, etc..)
- Work in collaboration with IML working group

More Information



Websites

http://root.cern.ch

http://oproject.org

http://iml.cern.ch

Thanks

ROOT

Data Analysis Framework