Machine Learning Tools in ROOT

L. Moneta, S. Gleyzer, O. Zapata

TMVA

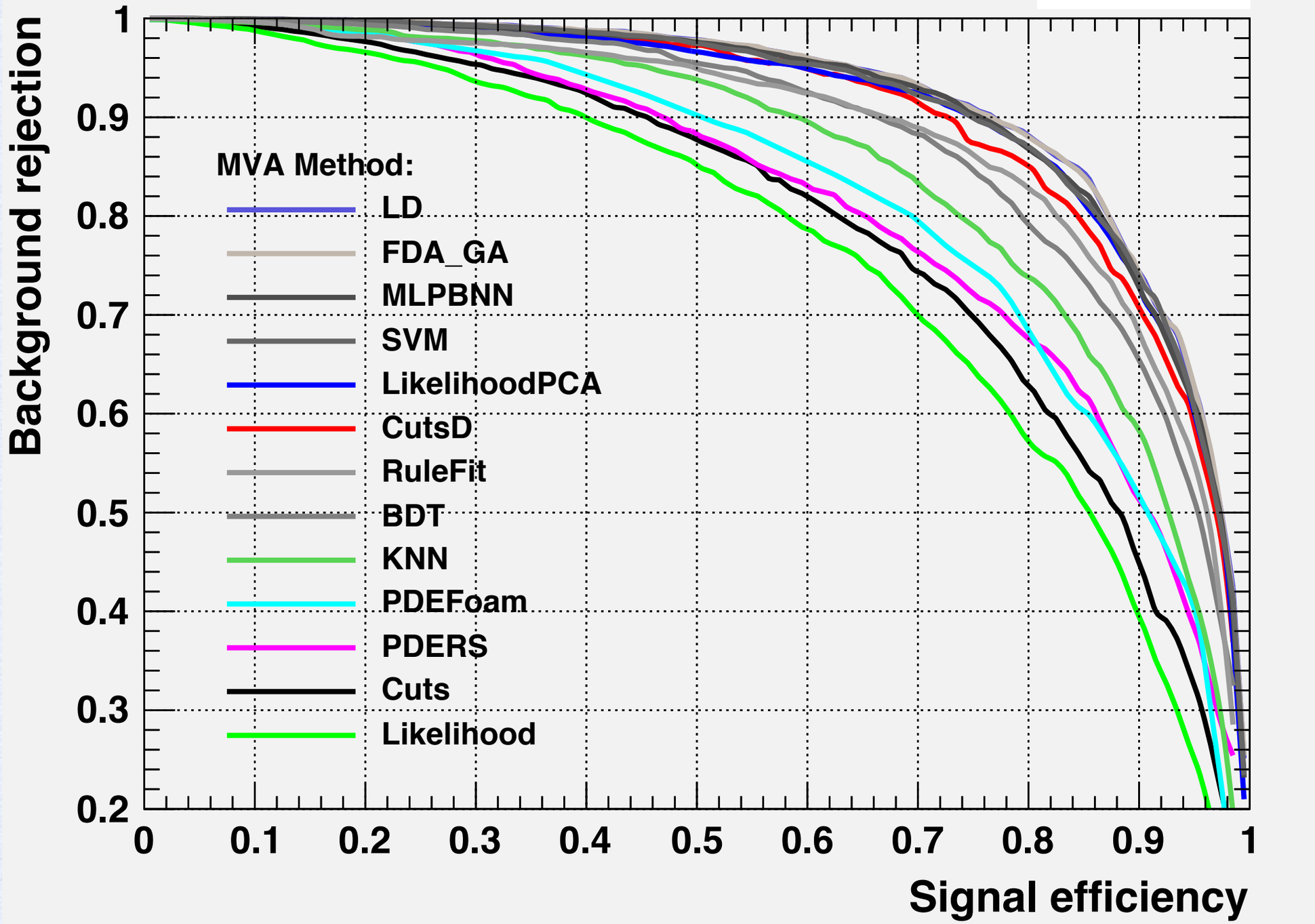
- Toolkit for multi-variate data analysis distributed with ROOT
  - created in 2005 by A. Hoecker, K. Voss and H. Voss
    - common toolkit for MVA tools for HEP
    - integrated in ROOT in 2006
    - it became soon popular for
      - easy to use
      - possible to use and compare several methods in simple way
  - MVA methods started to be very popular in HEP
    - TMVA is used in all major LHC analysis

TMVA Contents

- TMVA provides a collection of several MVA methods for classification or regressions
  - Rectangular cut optimisation
  - Projective likelihood estimation (PDE approach)
  - Multidimensional probability density estimation (PDE - range-search approach and PDE-Foam)
  - Multidimensional k-nearest neighbour method
  - Linear discriminant analysis (H-Matrix, Fisher and linear (LD) discriminants)
  - Function discriminant analysis (FDA)
  - Artificial neural networks (three different MLP implementations)
  - Boosted/Bagged decision trees
  - Predictive learning via rule ensembles (RuleFit)
  - Support Vector Machine (SVM)
- Most used is probably the Boosted Decision Tree

ROC Curve

Background rejection versus Signal efficiency



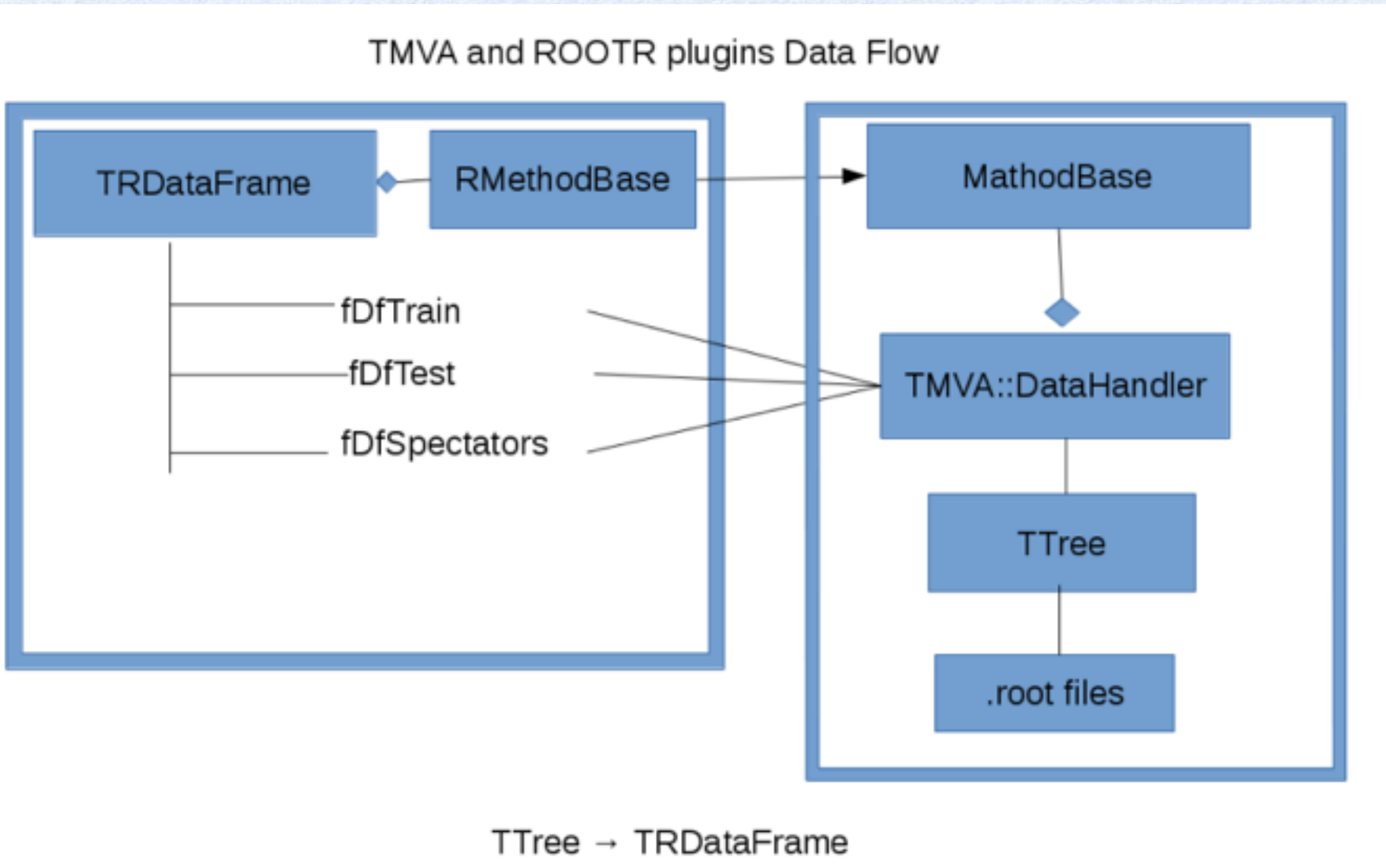
Recent Developments

- Integrate of new ML methods in TMVA
  - advantage of using same interfaces
    - users do not need to worry of learning on how to use new software tools
  - easy for comparison
- Integrate methods from R software project
  - using the recent ROOT-R code allowing to use R code inside ROOT and C++
- and from SciKit-Learn

R

- R provides several ML packages
  - e.g. for a review of them see
  - https://cran.r-project.org/web/views/MachineLearning.html
  - useful to integrate some of the most interesting for HEP in TMVA
  - use new ROOT-R interface to allow to call R code directly from ROOT
    - automatically input data are converted into the R data classes (data frame) and passed to the tools

R-TMVA



- Map ROOT data in a R data frame (TRDataFrame)
- Implement new R methods as derived class of TMVA::MethodBase

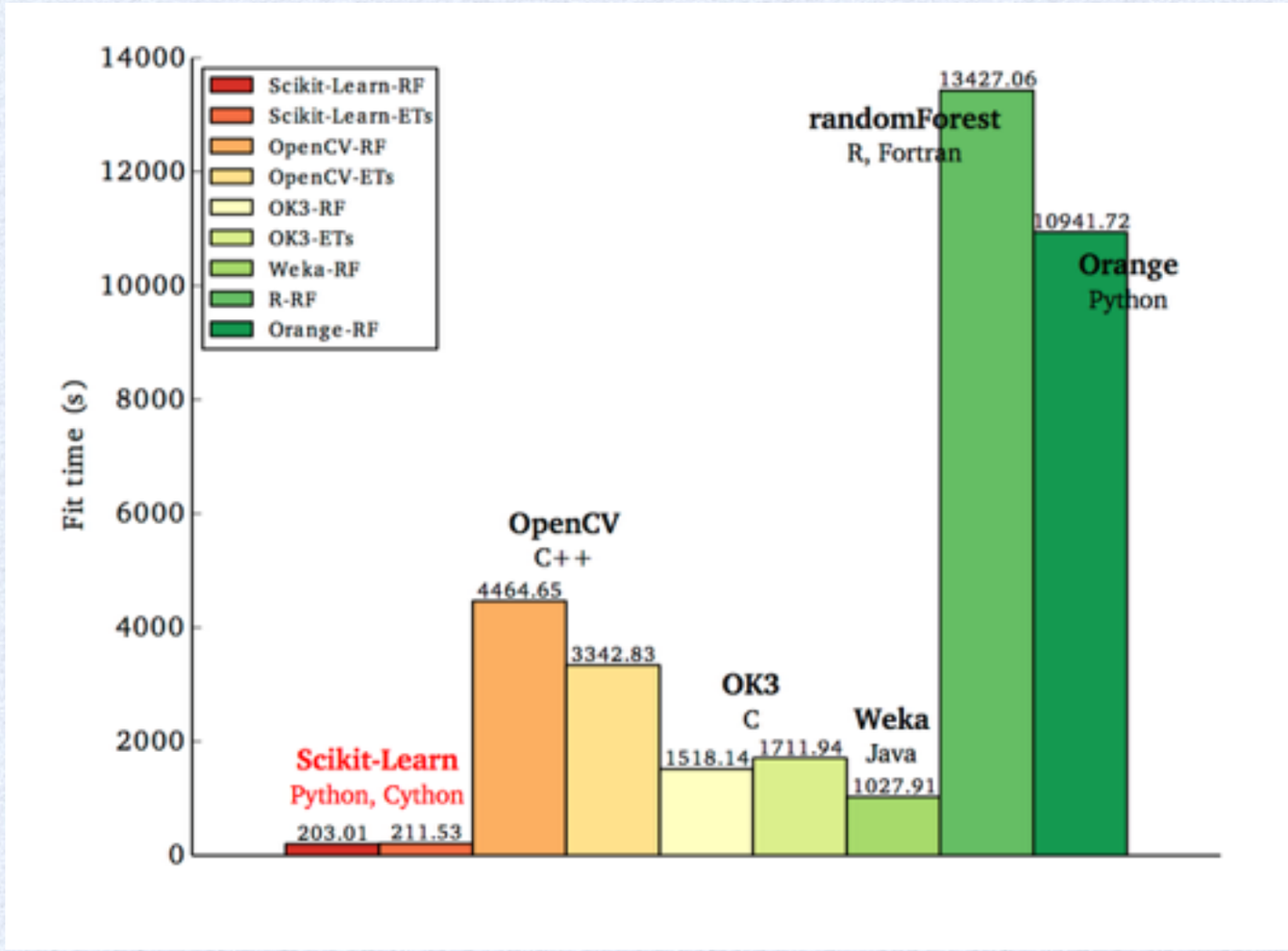
Available now in ROOT since version 6.04.04.
See doc at http://oproject.org/tiki-index.php?page=RMVA

R-TMVA

- Implemented plugin's for these R methods:
  - C50: C5.0 decision trees and rule-based models
  - 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
  - xgboost: Extreme gradient boosted
    - algorithm used by one of the winner by the Higgs ML Challenge

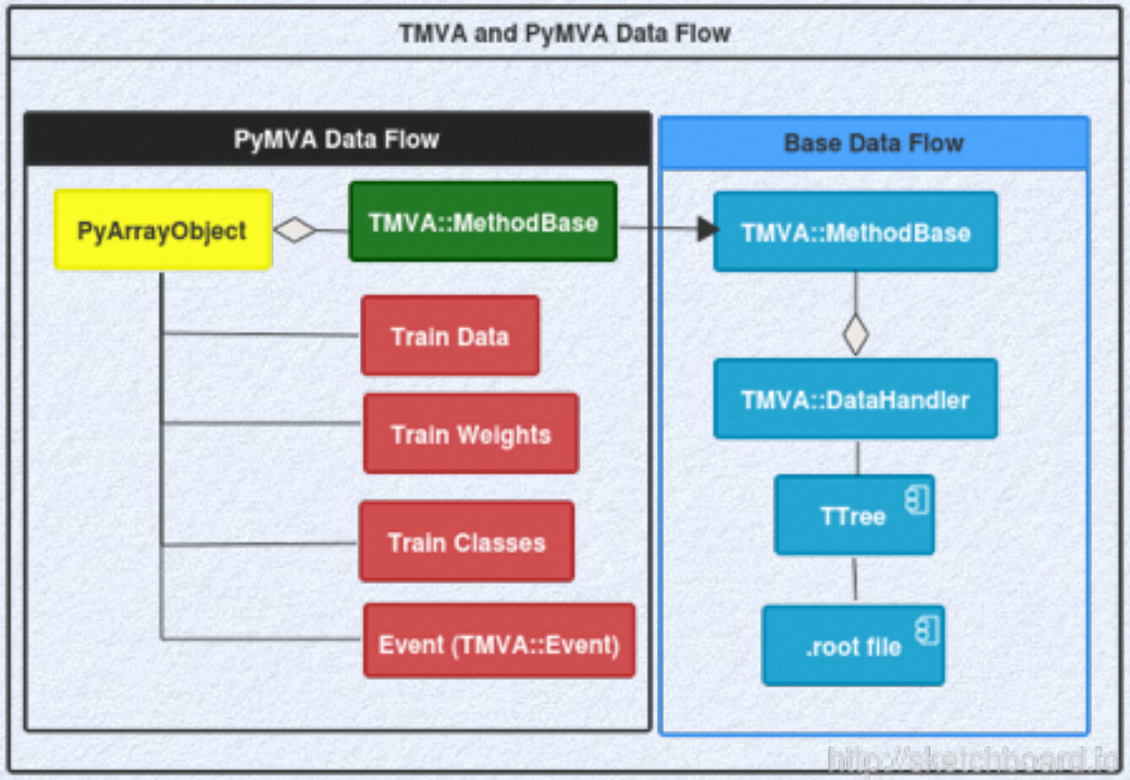
scikit-learn

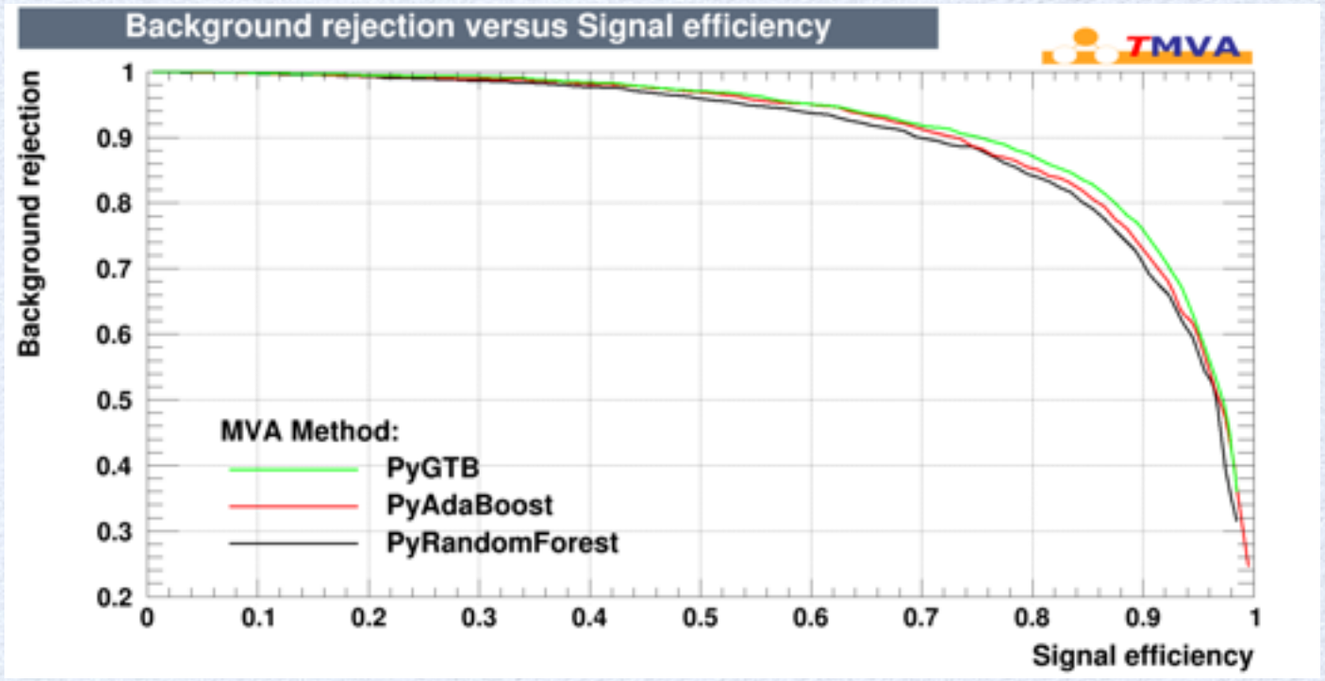
- Most popular Python ML package
  - build on NumPy and SciPy
- Several interesting algorithms for regression, classifications and clustering (e.g. random forest, gradient boosting, etc..)
- Very efficient implementations



PyMVA

- New interface to use Python ML tools from TMVA
  - Use Scikit-Learn methods
    - Random Forest, Gradient Tree Boost, Ada Boost
  - Convert ROOT data in PyArrayObjects (C interface to numpy)
  - Use directly Python from C++ using its C interface

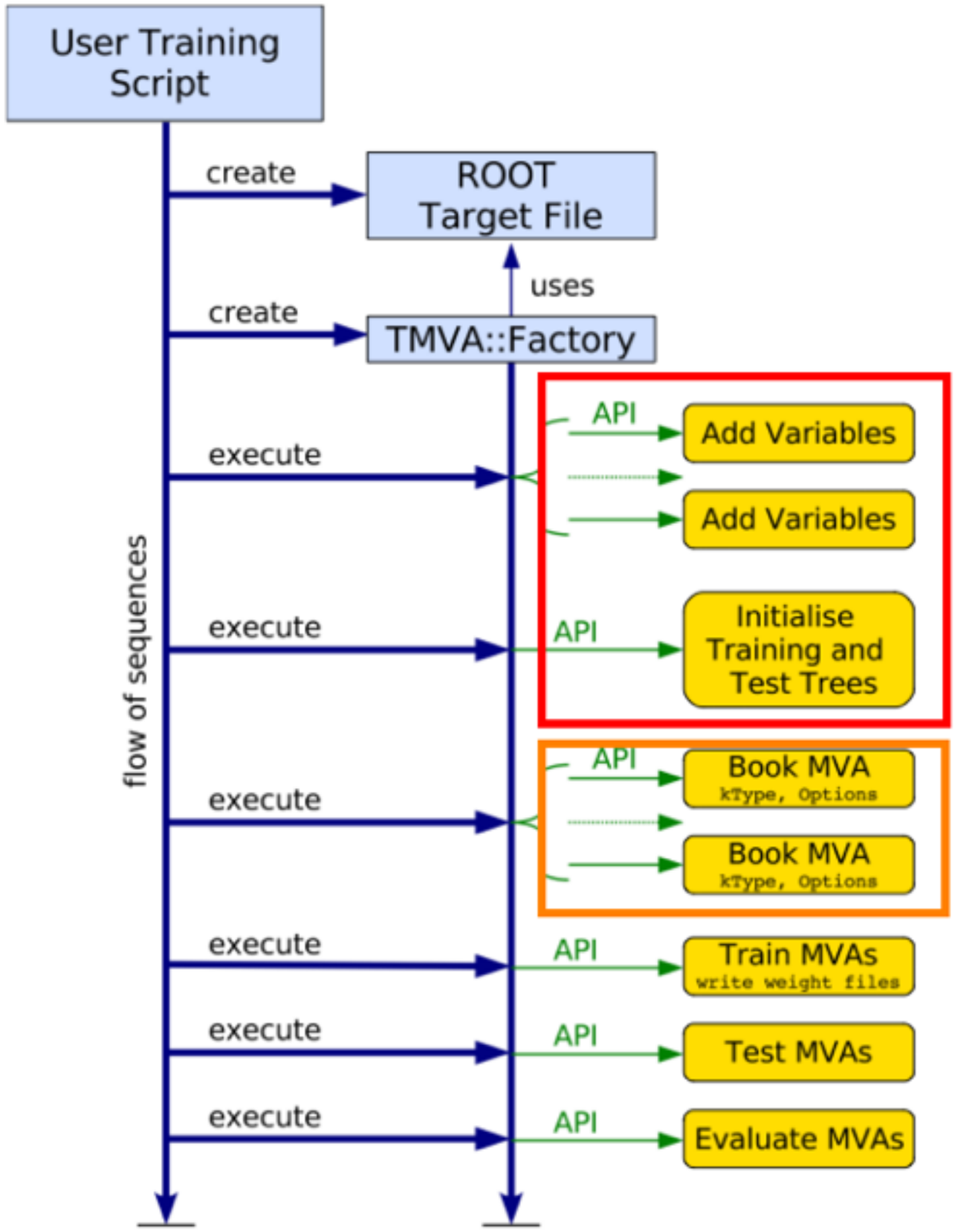




see http://oproject.org/tiki-index.php?page=PyMVA

available in ROOT 6.04.04

TMVA designed for comparison of different Machine Learning (ML) methods



Users operate via the Factory class

- Add variables
- Load datasets
- Book MVA methods
- Train
- Test
- Evaluate

Factory treats equally all booked methods

- Facilitate comparison
- Common pre-processing

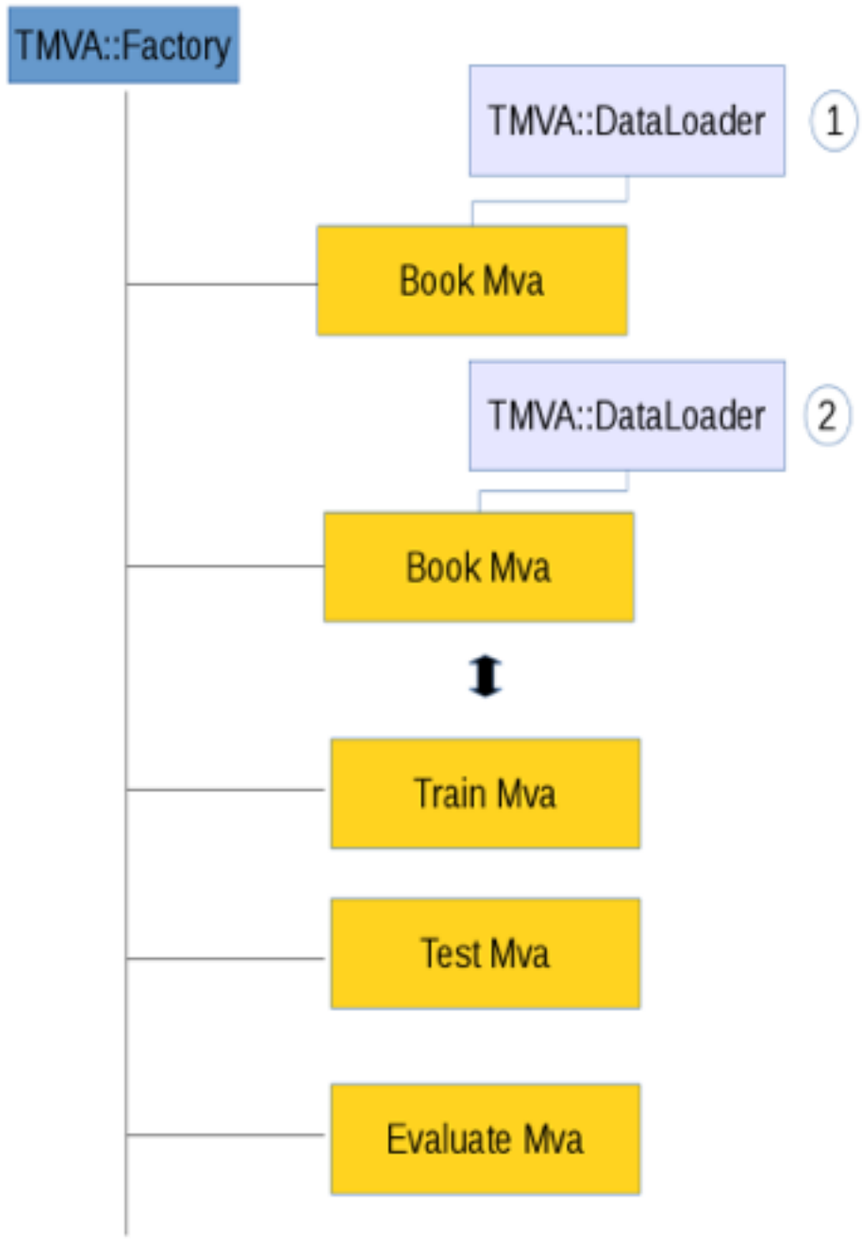
Issues with current TMVA

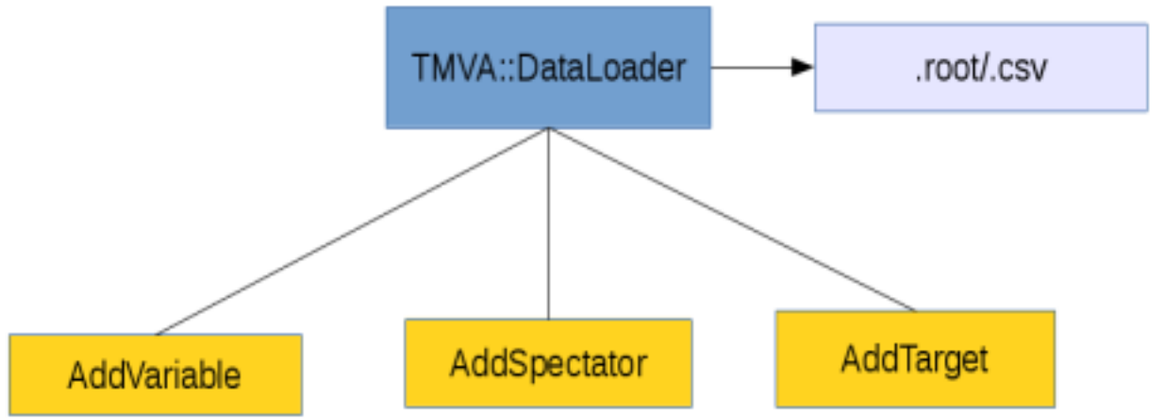
- All MVA methods see same data for training, test and evaluation. This is a limitation:
  - cannot use different datasets for different methods
  - cannot change input variables for the methods
- Difficult to integrate new methods for cross validation, variable importance,etc..
- Large use of static variables
  - cannot run more than one Factory in the same job
  - problem for concurrency
- Several issues raised by users in the ROOT questionnaire
  - performances, missing modern ML methods (e.g. deep neural networks)
- All these issues currently addressed by new IML working group

Example: TMVA::DataLoader

- DataLoader is a new interface to
  - load the datasets (root files, .csv files, ….)
  - add variables
- A specific MVA method is used together with a DataLoader instance
- This gives the desired flexibility in method customisation:
  - allows easy integration of variable importance and cross validation tools

TMVA::DataLoader






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

see http://oproject.org/tiki-index.php?page=TMVA

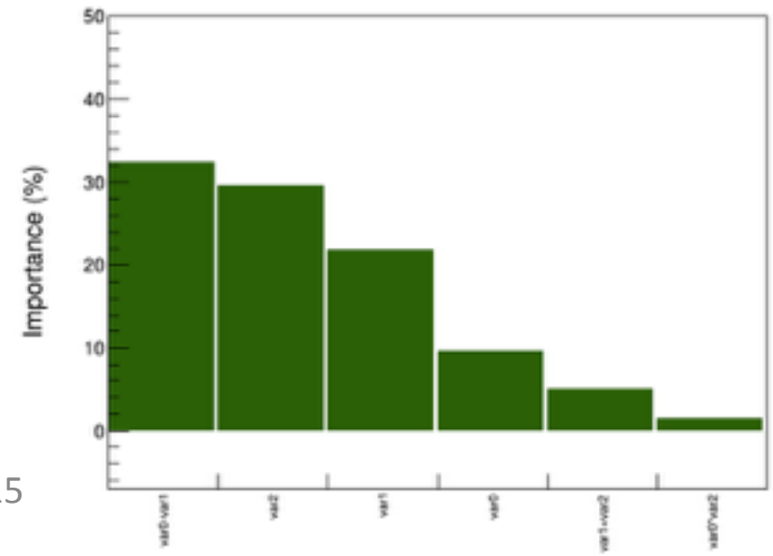
Feature Selection

New variable selection in TMVA (long-asked for feature) in classification context

- Variable importance $\longrightarrow$ proportional to classifier performance in which variable participates

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

  - Variable set V
  - Variable subsets S
  - Classifier performance $F(S)$



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

Amount of classifier loss (or gain) if variable X_i is removed

PoS(ACAT08) 207

Future Improvements in TMVA

- Persistency of methods
  - use general ROOT I/O (and not be limited to XML) for output of training
- Parallelisation (needed to exploit better new hardware)
  - limit usage of static variables for multi-threads
  - porting eventually to GPU when possible
- Code improvements and optimisation, exploit vectorisation and new C++ features.
- Improvements in memory usage. Better usage of ROOT I/O to allow for not requiring all data to be in memory.