Machine Learning Techniques for HEP Data Analysis with *T*MVA

Toolkit for Multivariate Analysis

Jörg Stelzer (*) (CERN)

CHEP 2007, Victoria, BC, Canada, September 3rd – 7th

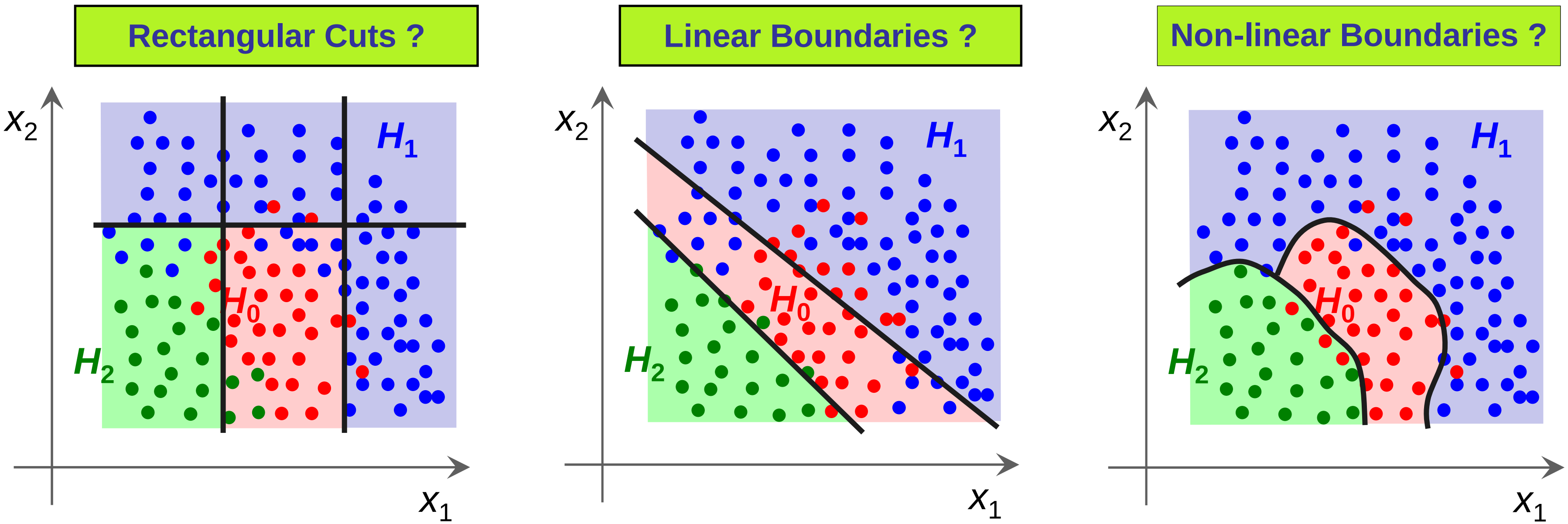
*** On behalf of the developers A. Höcker, P. Speckmayer, J. Stelzer, F. Tegenfeldt, H. Voss, K. Voss**

and the contributors A. Christov, Or Cohan, S. Henrot-Versille, M. Jachowski, A. Krasznahorkay Jr., Y. Mahalalel, R. Ospanov, X. Prudent, M. Wolter, A.Zemla

Special Thanks to Rene Brun and the ROOT team !

The General Classification Problem

- General definition of a classifier f: $R^n \rightarrow N$, $\boldsymbol{x} \rightarrow \{\mathbf{0}, \mathbf{1}, \mathbf{2}, \ldots\}$
  - Sample $\boldsymbol{x}$ (n discriminating input variables) in different categories
  - The problem: How to draw the boundaries between $\mathbf{H_0}$, $\mathbf{H_1}$, and $\mathbf{H_2}$ such that f($\boldsymbol{x}$) returns the true nature of $\boldsymbol{x}$ with maximum correctness



- Which method is best to find the optimal boundary?
  - Large n → Let the machine decide !

Machine Learning

Classification Problems in HEP

- In HEP mostly two class problems – signal (S) and background (B)
  - Event level (Higgs searches, …)
  - Cone level (Tau-vs-jet reconstruction, …)
  - Track level (particle identification, …)
  - Lifetime and flavour tagging (*b*-tagging, …)
  - ...

- Input information
  - Kinematic variables (masses, momenta, decay angles, …)
  - Event properties (jet/lepton multiplicity, sum of charges, …)
  - Event shape (sphericity, Fox-Wolfram moments, …)
  - Detector response (silicon hits, *dE*/*dx*, Cherenkov angle, shower profiles, muon hits, …)
  - …

Conventional Linear Classifiers

Cut Based

- **Widely used because transparent**
- **Machine optimization is challenging:**
  - **MINUIT fails for large *n* due to sparse population of input parameter space**
  - **Alternatives are Monte Carlo Sampling, Genetic Algorithms, Simulated Annealing**

Projective Likelihood Estimator

- **Probability density estimators for each variable combined into one**
- **Much liked in HEP**
  - **Returns the likelihood of a sample belonging to a class**
- **Projection ignores correlation between variables**
  - **Significant performance loss for correlated variables**

Linear Fisher Discriminant

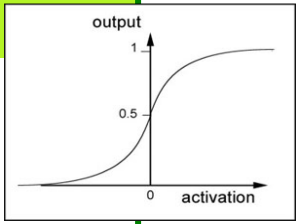
- **Axis in parameter space on which samples are projected, chosen such that signal and background are pushed far away from each other**
  - **Optimal classifier for linearly correlated Gaussian-distributed variables**
  - **Means of signal and background must be different**

*R.A. Fisher, Annals Eugenics **7**, 179 (1936).*

Common Non-linear Classifiers

Neural Network

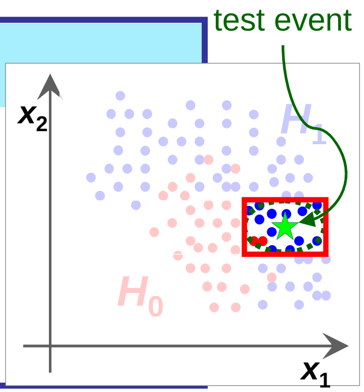
- **Feed forward multilayer perceptron**
  - **Non-linear activation function of each neuron**
  - **<u>Weierstrass theorem</u>: can approximate any continuous functions to arbitrary precision with a single hidden layer and an infinite number of neurons**



("Activation" function)

PDE Range-Search, k Nearest Neighbours

- **n- dimensional signal and background PDF, probability obtained by counting number of signal and background events in vicinity of test event**
  - **Range Search: vicinity is predefined volume**
  - **k nearest neighbor: adaptive (k events in volume)**



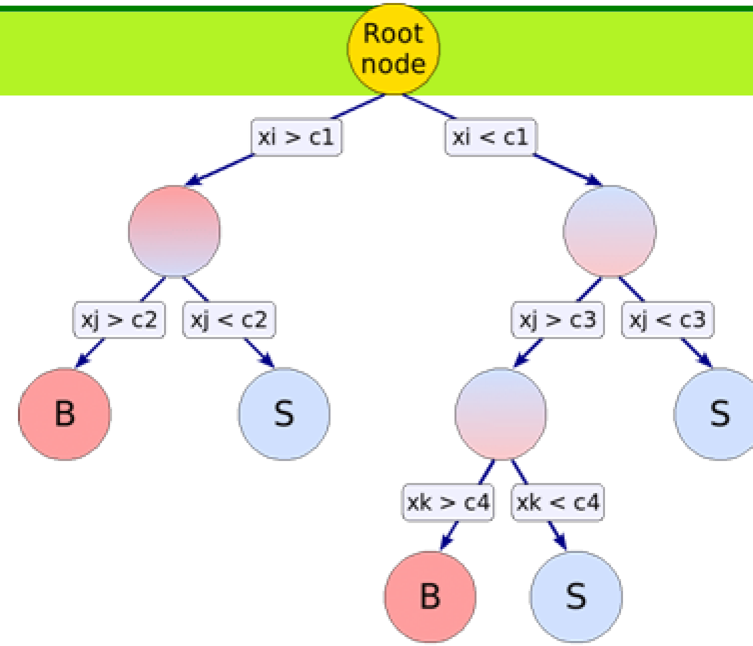
T. Carli and B. Koblitz, Nucl. Instrum. Meth. A501, 576 (2003) [hep-ex/0211019]

Function Discriminant Analysis

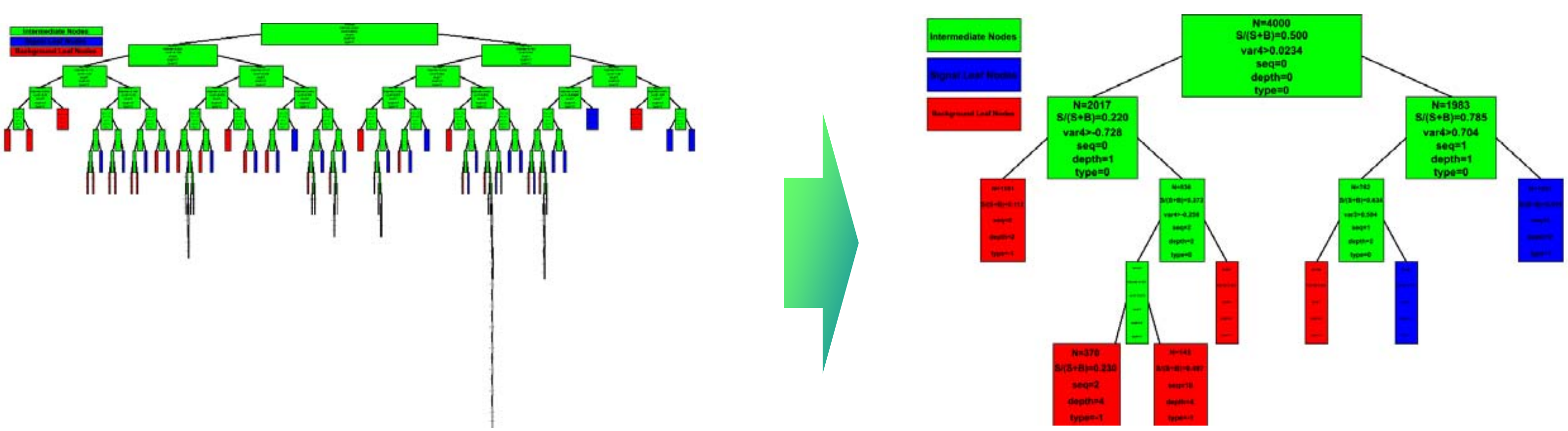
- **User provided separation function fitted to the training data**
  - **Simple, transparent discriminator for non-linear problems**
  - **In-between solution (better then Fisher, but not good for complex examples)**

Classifiers Recent in HEP

Boosted Decision Trees



- **Decision Tree is a series of cuts that split sample set into ever smaller sets, leafs are assigned either signal or background status**
  - **Each split try to maximizing gain in separation (Gini-index)**
- **Bottom-up pruning of a decision tree**
  - **Protect from overtraining (*) by removing statistically insignificant nodes**
- **DT easy to understand but not powerful**



* Performance on training sample statistically better than on independent test sample

Classifiers Recent in HEP

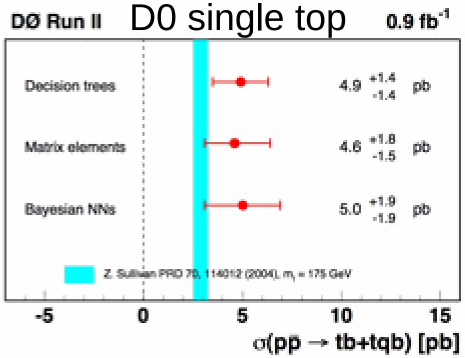
Boosted Decision Trees

- **Decision Tree is a series of cuts that split sample set into ever smaller sets, leafs are assigned either signal or background status**
  - **Each split try to maximizing gain in separation (Gini-index)**
- **Bottom-up pruning of a decision tree**
  - **Protect from overtraining (*) by removing statistically insignificant nodes**
- **DT easy to understand but not powerful**

Boosting

- **Increase the weight of incorrectly identified events and build a new decision tree**
- **Final classifier: ‘forest’ of decision trees linearly combined**
  - **Large coefficient for tree with small misclassification**
  - **Improved performance and stability**

Little tuning required for good performance



* Performance on training sample statistically better than on independent test sample

Classifiers Recent in HEP

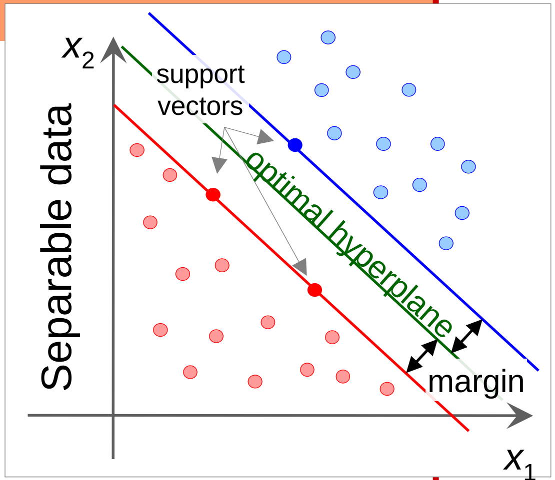
Learning via Rule Ensembles

- **Rule is a set of cuts, defining regions in the input parameter space**
  - **Rules extracted from a forest of Decision Trees (either from BDT, or a random forest generator)**
  - **Linear combinations of rules, coefficients fitted by minimizing risk of misclassification**
- **Good performance**

J. Friedman and B.E. Popescu, "Predictive Learning via Rule Ensembles", Technical Report, Statistics Department, Stanford University, 2004.

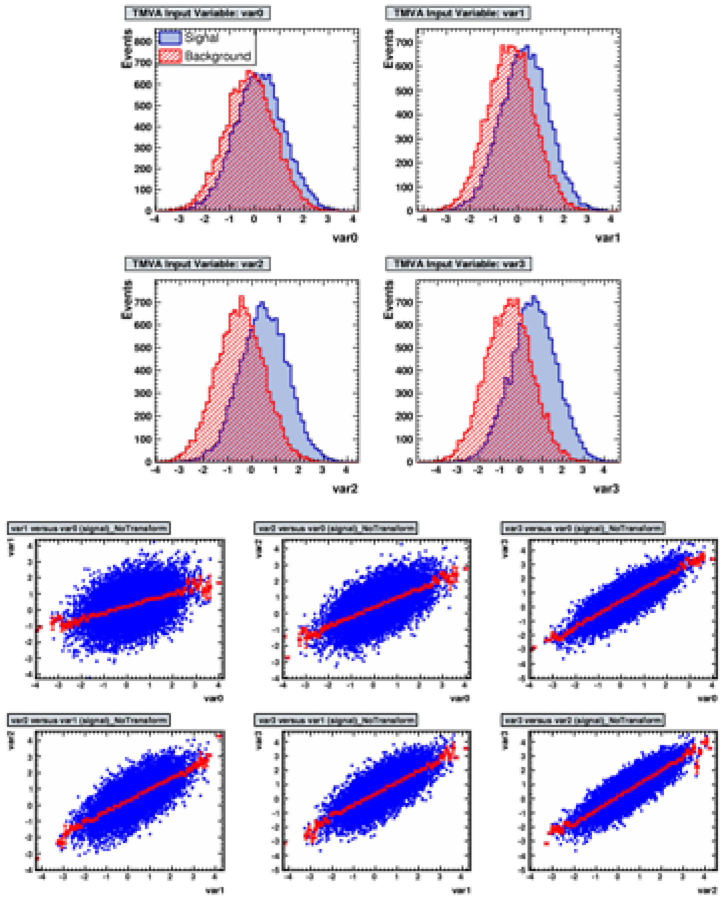
Support Vector Machines

- **Optimal hyperplane between linearly-separable data (1962)**
  - **Wrongly classified events add an extra term to the cost-function which is minimized**
- **Non-separable data becomes linearly separable in higher dimensions Φ: $R^n \rightarrow R^\infty$**
- **Kernel trick (suggested 1964, applied to SVM 1992)**
  - **Cost function depends only on $\Phi(x)\bullet\Phi(y) = K(x,y)$, no explicit knowledge of F required**



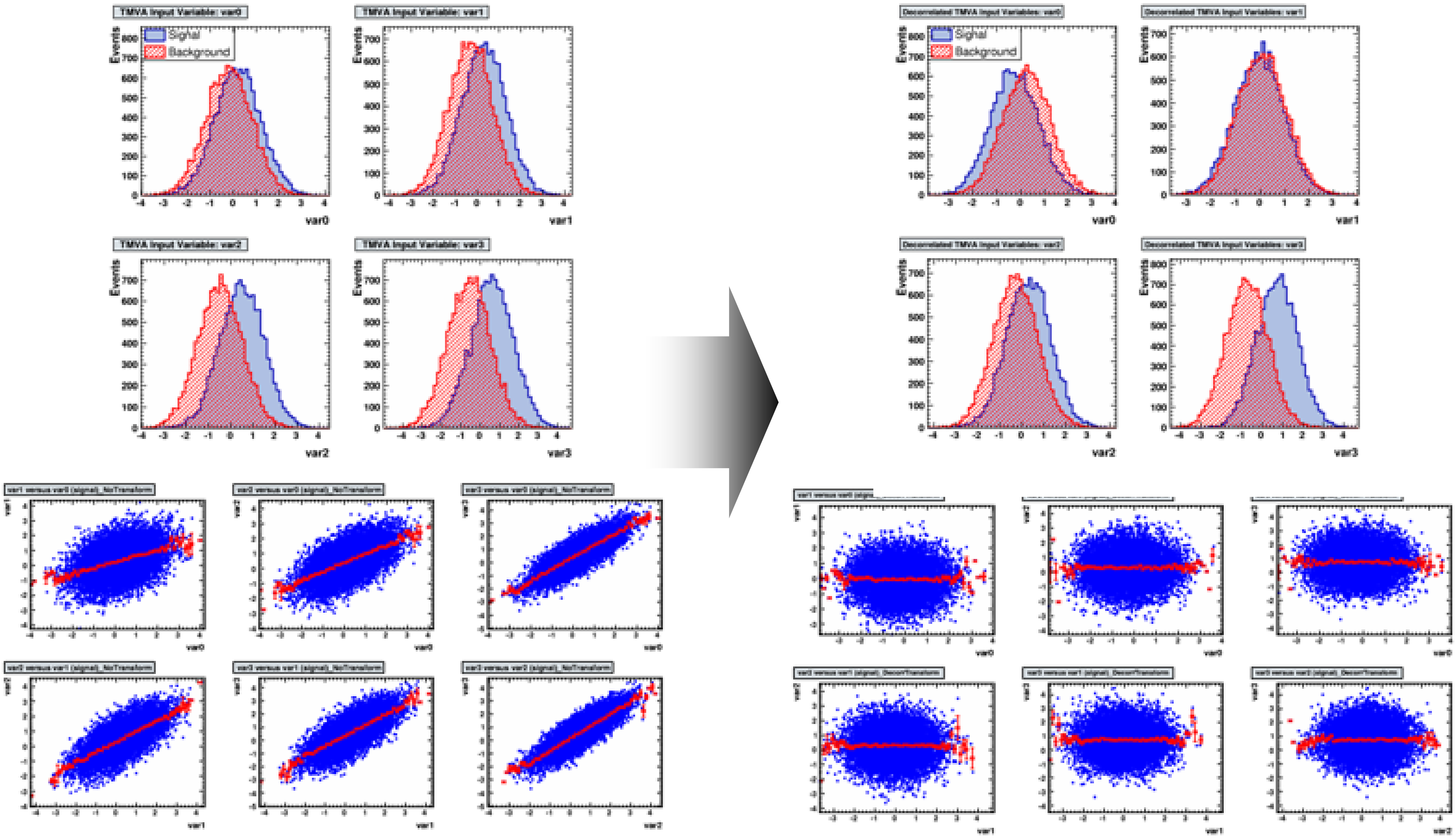
C. Cortes and V. Vapnik, "Support vector networks", Machine Learning, 20, 273 (1995).

Data Preprocessing: Decorrelation



- Removal of linear correlations by rotating input variables
  - Determine *square-root* C' of covariance matrix C, *i.e.*, $C = C'C'$
  - Transform original (x) into decorrelated variable space (x') by: $x' = C'^{-1}x$

Data Preprocessing: Decorrelation



- Complete decorrelation only possible in case of Gaussian distributions with linear correlations
- Useful for cut- or projective likelihood classifier

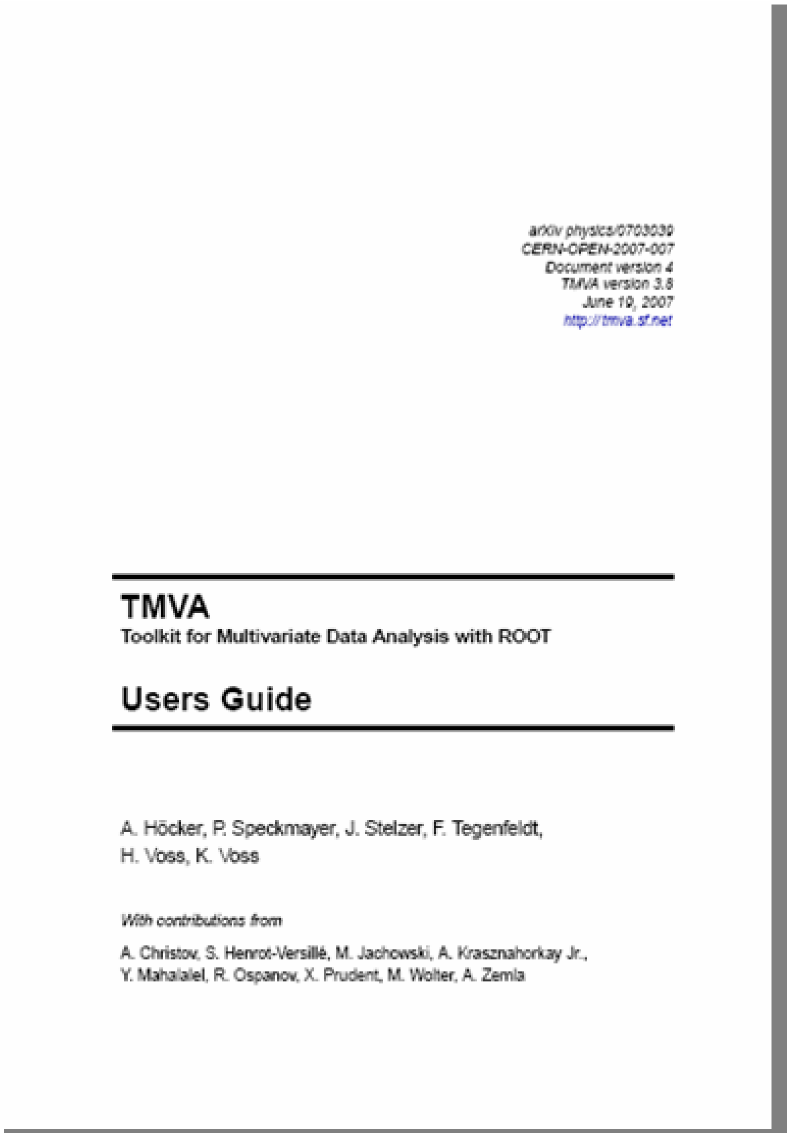
What is *T*MVA

- Motivation: Classifiers perform very different depending on the data, all should be tested on a given problem
  - Situation for many year: usually only a small number of classifiers were investigated by analysts
  - Needed a **Tool** that enables the analyst to **simultaneously evaluate the performance of a large number of classifiers** on his/her dataset
- Design Criteria: Performance and Convenience

  (A good tool does not have to be difficult to use)
  - **Training**, **testing**, and **evaluation** of many classifiers in parallel
  - **Preprocessing** of input data: decorrelation (PCA, Gaussianization)
  - **Illustrative** tools to **compare performance of all classifiers** (ranking of classifiers, ranking of input variable, choice of working point)
  - Actively protect against **overtraining**
  - Straight forward **application** to test data
- Special needs of high energy physics should be addressed
  - Two classes, events weights, familiar terminology

Technical Aspects

- TMVA is open source, written in C++, and based on ROOT
  - Development on SourceForge, there is all the information
  - Bundled with ROOT since 5.11-03
- Training requires ROOT-environment, resulting classifiers also available as standalone C++ code (except two)
- Six core developers, many contributors
  - > 1400 downloads since Mar 2006 (not counting ROOT users)
  - Mailing list for reporting problems



arXiv physics/0703039
CERN-OPEN-2007-007
Document version 4
TMVA version 3.8
June 19, 2007
http://tmva.sf.net

TMVA
Toolkit for Multivariate Data Analysis with ROOT

Users Guide

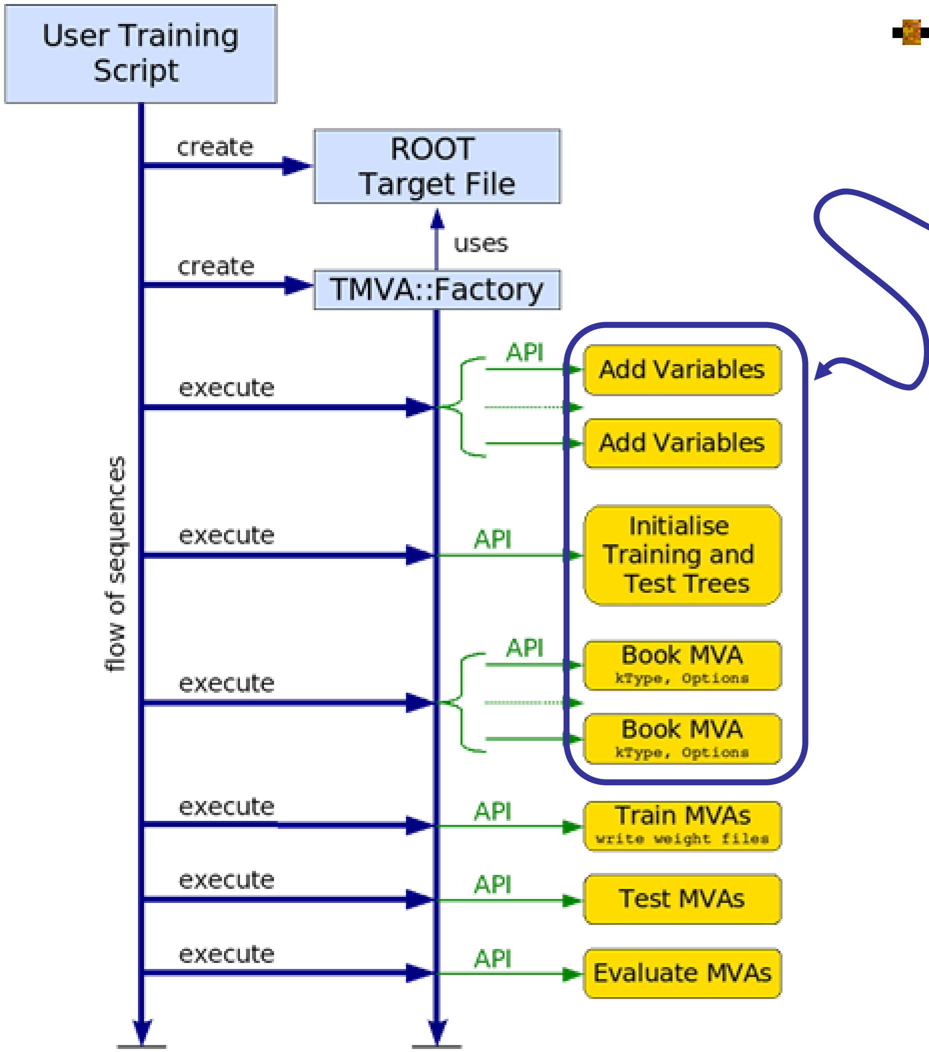
A. Höcker, P. Speckmayer, J. Stelzer, F. Tegenfeldt, H. Voss, K. Voss

With contributions from

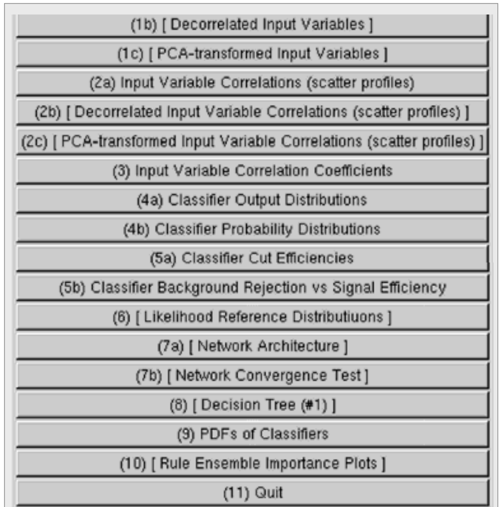
A. Christov, S. Henrot-Versillé, M. Jachowski, A. Krasznahorkay Jr., Y. Mahalalel, R. Ospanov, X. Prudent, M. Wolter, A. Zemla

Users Guide at http://sf.tmva.net:
97p., classifier descriptions, code examples
arXiv physics/0703039

Using TMVA



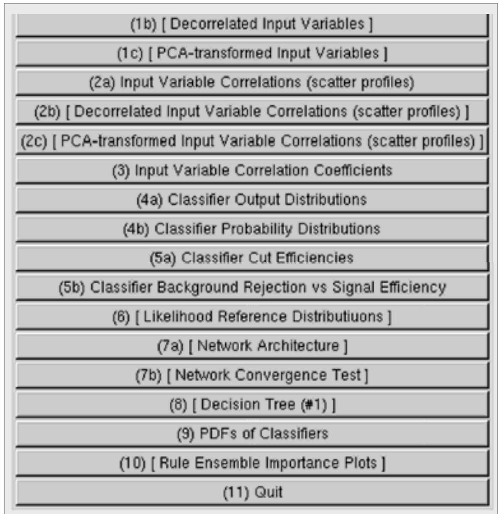
- User usually starts with template TMVAnalysis.C
  - Choose training variables
  - Choose input data
  - Select classifiers



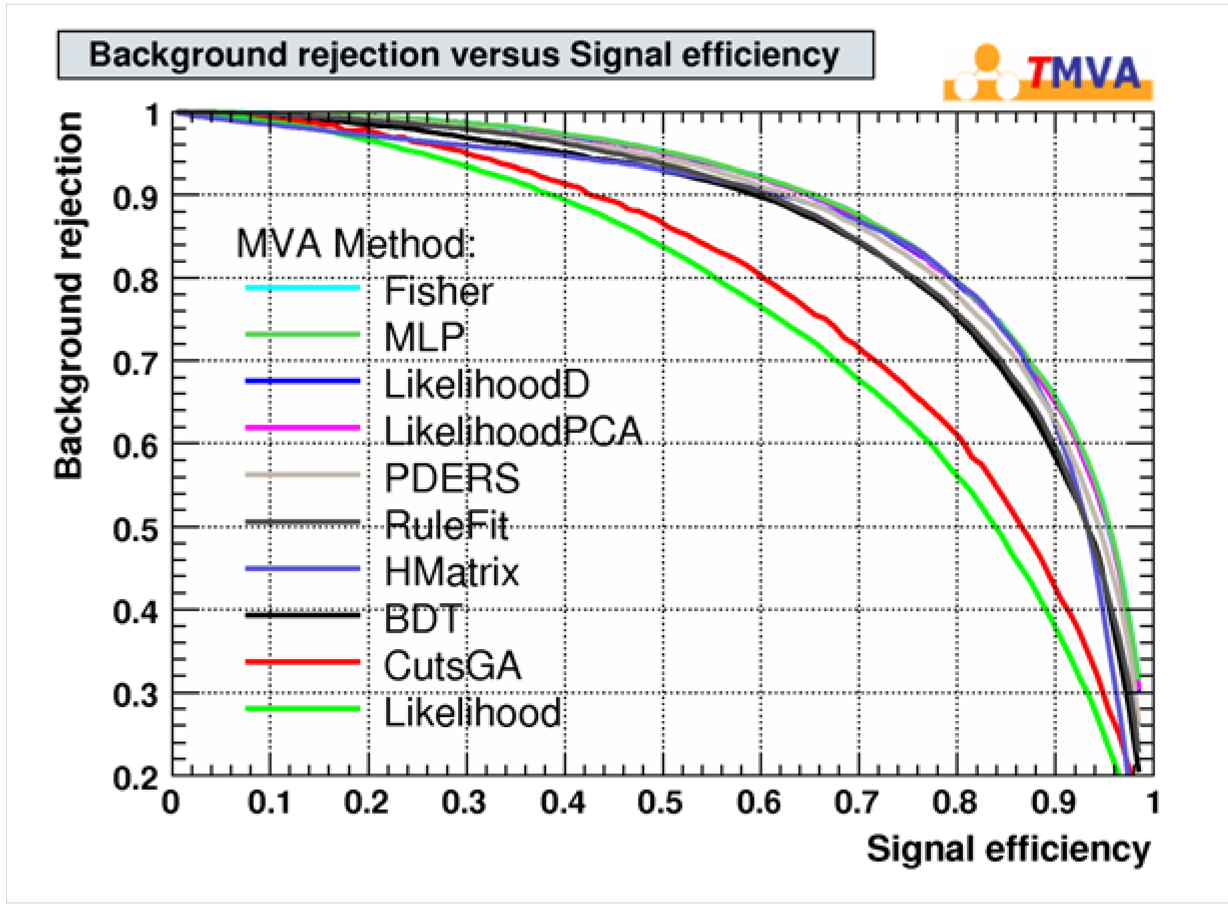
TMVA GUI

Template TMVAnalysis.C available at $TMVA/macros/ and $ROOTSYS/tmva/test/

Using TMVA



TMVA GUI



More Evaluation Output


```
   Evaluation results ranked by best signal efficiency and purity (area)
-----------------------------------------------------------------------------
 MVA               Signal efficiency at bkg eff. (error):  |  Sepa-    Signifi-
 Methods:          @B=0.01    @B=0.10    @B=0.30    Area   |  ration:  cance:
-----------------------------------------------------------------------------
 Fisher          : 0.268(03)  0.653(03)  0.873(02)  0.882  |  0.444    1.189
 MLP             : 0.266(03)  0.656(03)  0.873(02)  0.882  |  0.444    1.260
 LikelihoodD     : 0.259(03)  0.649(03)  0.871(02)  0.880  |  0.441    1.251
 PDERS           : 0.223(03)  0.628(03)  0.861(02)  0.870  |  0.417    1.192
 RuleFit         : 0.196(03)  0.607(03)  0.845(02)  0.859  |  0.390    1.092
 HMatrix         : 0.058(01)  0.622(03)  0.868(02)  0.855  |  0.410    1.093
 BDT             : 0.154(02)  0.594(04)  0.838(03)  0.852  |  0.380    1.099
 CutsGA          : 0.109(02)  1.000(00)  0.717(03)  0.784  |  0.000    0.000
 Likelihood      : 0.086(02)  0.387(03)  0.677(03)  0.757  |  0.199    0.682
-----------------------------------------------------------------------------
  Testing efficiency compared to training efficiency (overtraining check)
-----------------------------------------------------------------------------
       MVA           Signal efficiency: from test sample (from traing sample)
       Methods:         @B=0.01              @B=0.10              @B=0.30
-----------------------------------------------------------------------------
    Fisher         : 0.268 (0.275)       0.653 (0.658)       0.873 (0.873)
    MLP            : 0.266 (0.278)       0.656 (0.658)       0.873 (0.873)
    LikelihoodD    : 0.259 (0.273)       0.649 (0.657)       0.871 (0.872)
    PDERS          : 0.223 (0.389)       0.628 (0.691)       0.861 (0.881)
    RuleFit        : 0.196 (0.198)       0.607 (0.616)       0.845 (0.848)
    HMatrix        : 0.058 (0.060)       0.622 (0.623)       0.868 (0.868)
    BDT            : 0.154 (0.268)       0.594 (0.736)       0.838 (0.911)
    CutsGA         : 0.109 (0.123)       1.000 (0.424)       0.717 (0.715)
    Likelihood     : 0.086 (0.092)       0.387 (0.379)       0.677 (0.677)
-----------------------------------------------------------------------------
```

More Evaluation Output

Variable Ranking

Better variable

```
--- Fisher       : Ranking result (top variable is best ranked)
--- Fisher       : ----------------------------------------------------------------
--- Fisher       : Rank : Variable  : Discr. power
--- Fisher       : ----------------------------------------------------------------
--- Fisher       :    1 : var4      : 2.175e-01
--- Fisher       :    2 : var3      : 1.718e-01
--- Fisher       :    3 : var1      : 9.549e-02
--- Fisher       :    4 : var2      : 2.841e-02
--- Fisher       : ----------------------------------------------------------------
```

- how useful is a variable?

Classifier correlation

```
--- Factory      : Inter-MVA overlap matrix (signal):
--- Factory      : ------------------------------
--- Factory      :             Likelihood  Fisher
--- Factory      : Likelihood:     +1.000  +0.667
--- Factory      :     Fisher:     +0.667  +1.000
--- Factory      : ------------------------------
```

- do classifiers perform the same separation into signal and background?

Some General Remarks on MVA

No black boxes

- Cuts and Likelihood are transparent, so if they perform (rarely the case) use them
- In <u>presence of correlations</u> other classifiers are better
  - Correlations are difficult to understand at any rate
- Multivariate classifiers are <u>no black boxes</u>, we just need to understand them

Differences MC and Data

Training data (MC) might not describe detector data well. This is not good, but <u>not necessarily a large problem</u>:

- performance on real data will be worse than training results (bad training)
- general rule: unless verified with control sample don't use MC efficiencies in data analysis → bias
- optimized cuts are in general <u>not</u> less vulnerable to systematics

Systematics

There is <u>no principle difference</u> in systematics evaluation between single discriminating variables and MV classifiers

- Control sample for classifier output (not necessarily for each input variable)
- If variable with large uncertainty → shift/smear that variable and retrain
  - see if variable gets less emphasized

A Summary of What is – and What's to Come

TMVA provides easy access to a large number of multivariate classifiers and helps the user to utilize these for an optimized signal selection as part of the data analysis

- Besides TMVA a couple more packages for parallelized MV training and evaluation in HEP
  - Pioneer: Cornelius package (BABAR)
  - Frequently used: StatPatterRecognition (some overlap with TMVA)
  - Many individual implementations

Current developments

- Applying the general boosting procedure to all classifiers
  - More robust classifiers with better performance
- Generalized classifier → goal: optimal performance
  - Combine *any* classifiers using *any* set of input variables in *any* phase space region

http://sf.net/projects/tmva http://root.cern.ch/

No Single Best !

Criteria		Classifiers								
		Cuts	Likeli-hood	PDERS/ k-NN	H-Matrix	Fisher	MLP	BDT	RuleFit	SVM
Perfor-mance	no / linear correlations	😐	☺	☺	😐	☺	☺	😐	☺	☺
	nonlinear correlations	😐	☹	☺	☹	☹	☺	☺	😐	☺
Speed	Training	☹	☺	☺	☺	☺	😐	☹	😐	☹
	Response	☺	☺	☹/😐	☺	☺	☺	😐	😐	😐
Robust -ness	Overtraining	☺	😐	😐	☺	☺	☹	☹	😐	😐
	Weak input variables	☺	☺	☹	☺	☺	😐	😐	😐	😐
Curse of dimensionality		☹	☺	☹	☺	☺	😐	😐	😐	😐
Clarity		☺	☺	😐	☺	☺	☹	☹	☹	☹