*"Prejudice meets Reality"*
**Helmholtz workshop, September 20-22, Bonn**

# Multivariate Analysis Methods and Boosted Decision Trees

Eckhard von Toerne
University of Bonn

TMVA

# Contents

- What is a multivariate Method?
- Example: Boosted Decision Trees
- The TMVA Toolkit
- TMVA tutorial

# References

Literature:

T.Hastie, R.Tibshirani, J.Friedman, "*The Elements of Statistical Learning*", Springer 2001
C.M.Bishop, "*Pattern Recognition and Machine Learning*", Springer 2006

Software packages for Multivariate Data Analysis/Classification:

Individual classifier software:

- *e.g.* "JETNET" C.Peterson, T. Rognvaldsson, L.Loennblad,
- NeuroBayes®
- many, many other packages!

"Complete" packages

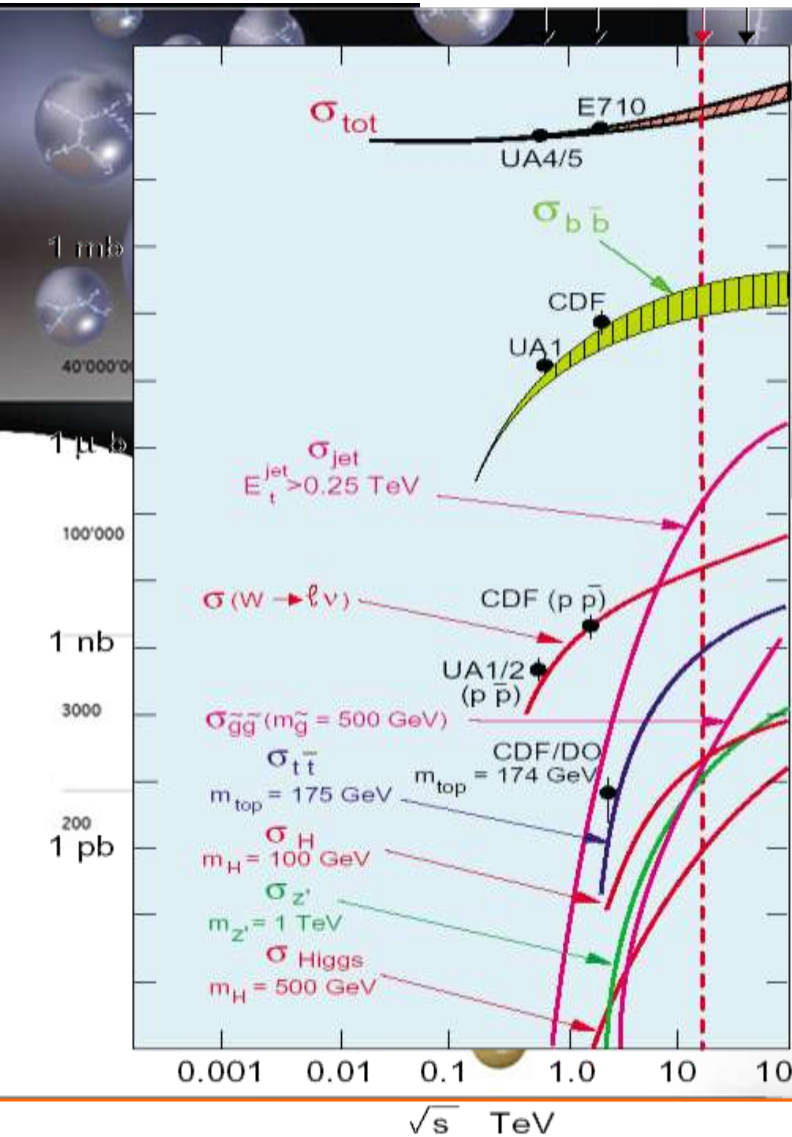
StatPatternRecognition: I.Narsky, *arXiv: physics/0507143*
*http://www.hep.caltech.edu/~narsky/spr.html*

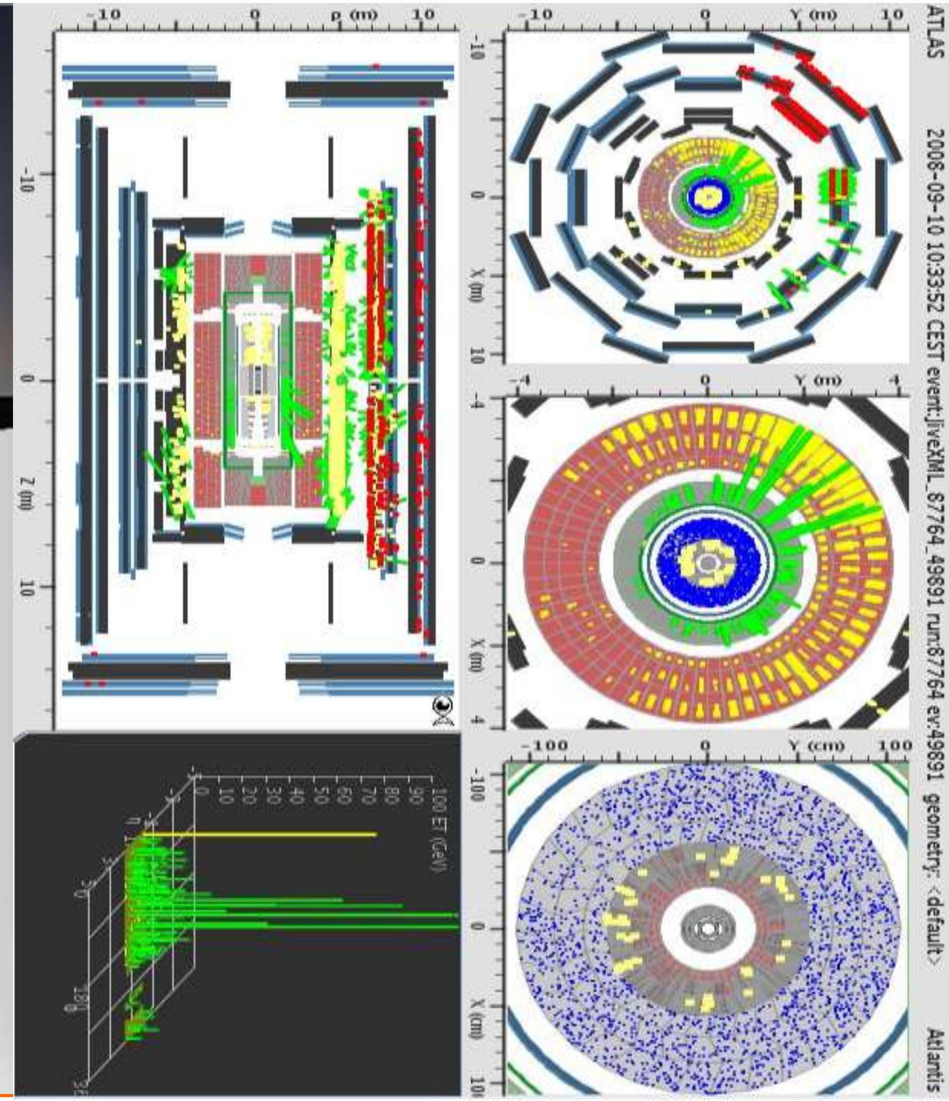
TMVA: Hoecker, Speckmayer, Stelzer, Therhaag, von Toerne, Voss,
*arXiv: physics/0703039, http://tmva.sf.net or every ROOT distribution*

Huge data analysis library available in "R": http://www.r-project.org/

# What is a multivariate method?

# Searches for New Physics



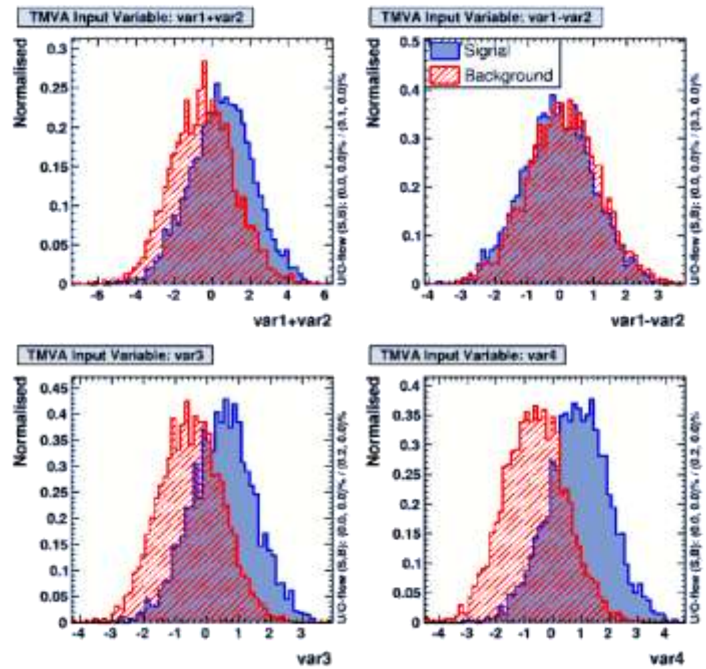




→ **Searching for needles in haystacks**

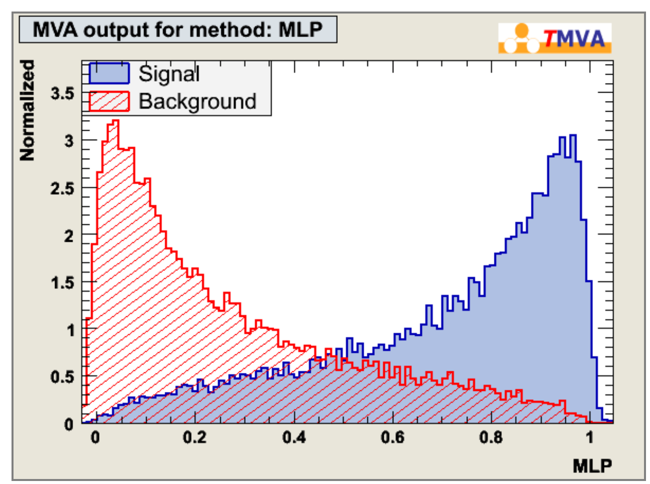
**Can a machine learn that for you? Yes, with multi-variate techniques!**

# What is a multi-variate analysis

- “Combine“ all input variables into one output variable
- Supervised learning means learning by example: the program extracts patterns from training data
- Methods for un-supervised learning → not common in HEP and not covered in this lecture



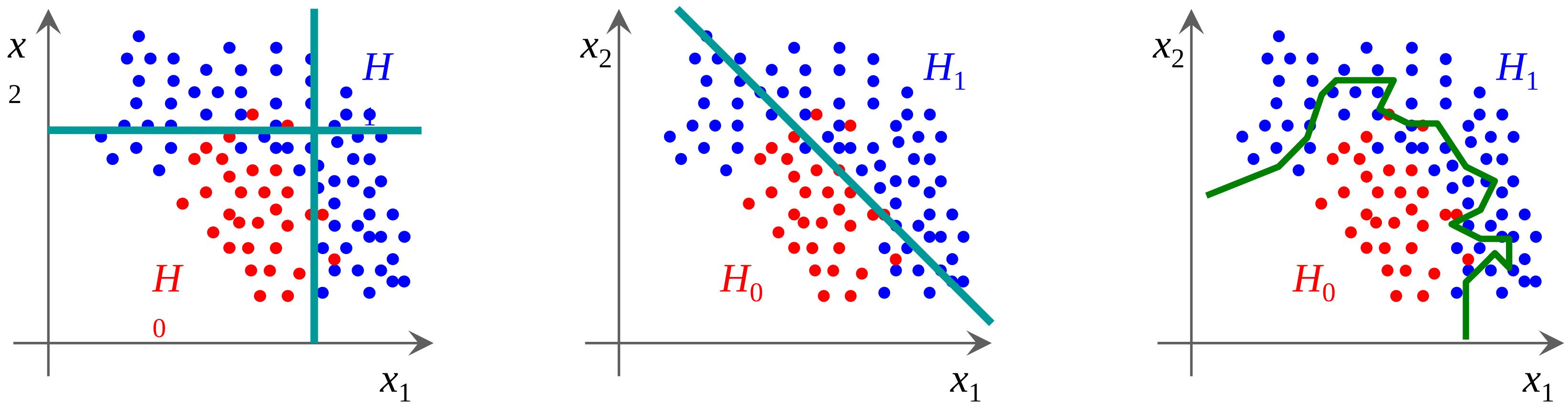

Input Variables



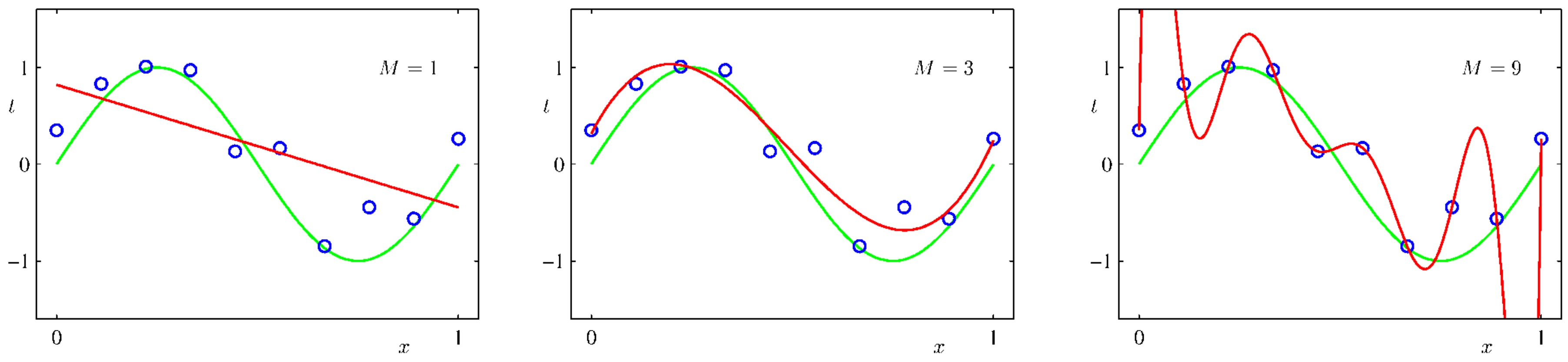

Classifier Output

# Classification/Regression

Classification of signal/background – How to find best decision boundary?

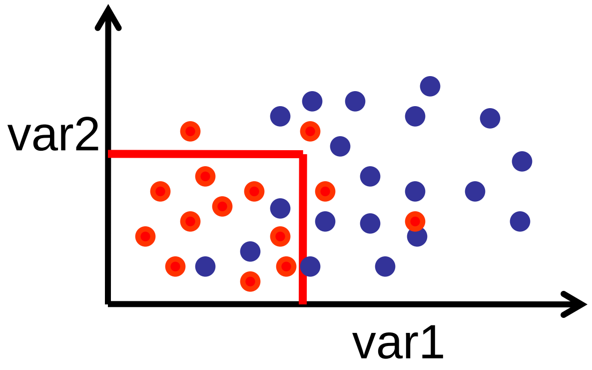



Regression – How to determine the correct model?



# Cuts

- Simplest multi-variate classification method: **Cuts**
- Signal region is defined by a series of cuts:
  - Var1 > x1
  - Var2 < x2 ...

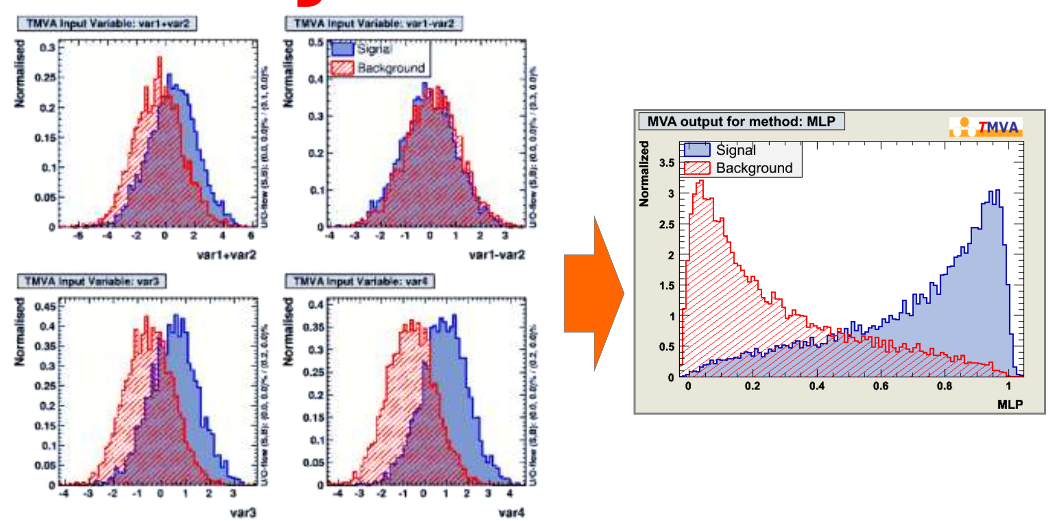



- However not necessarily the easiest approach
  - Cuts have difficulties if variables have low separation power and many variables are involved
- Whenever cut selections are employed, more sophisticated multi-variate methods may also work

# Typical multi-variate analysis steps

- Choice of input variables
- Define preselection
- Choice of MVA method
- Training the MVA method using samples with known signal/background
- Choice of working point

physics input is crucial



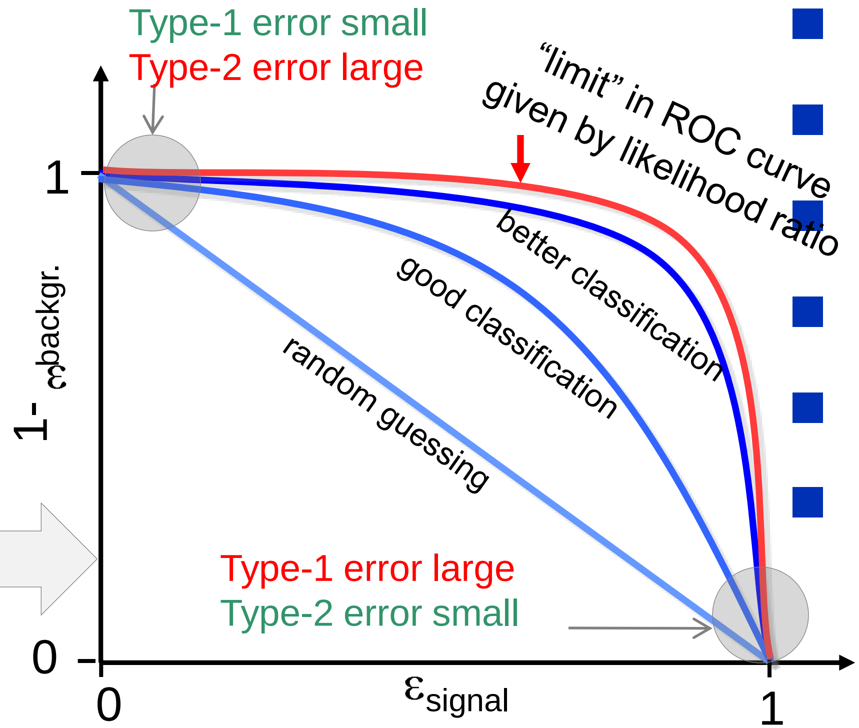
# Neyman-Pearson Lemma

Likelihood Ratio : $y(x) = \frac{P(x|S)}{P(x|B)}$

**Neyman-Pearson:**

The Likelihood ratio used as "selection criterion" y(x) gives for each selection efficiency the best possible background rejection.

*i.e.* it maximizes the area under the "*Receiver Operation Characteristics*" (ROC) curve




Varying y(x)>"cut" moves working point (efficiency and purity) along ROC curve

How to choose "cut"? → need to know prior probabilities (S, B abundances)

- Measurement of signal cross section: maximum of S/√(S+B) or equiv. √(ε·p)
- Discovery of a signal : maximum of S/√(B)
- Precision measurement: high purity (p)
- Trigger selection: high efficiency (ε)

# How to choose a method?

- If you have a training sample with only few events?
  - → Number of „parameters“ must be limited
  - → Use Linear classifier or FDA, small BDT, small MLP
- Variables are uncorrelated (or only linear corrs) → likelihood
- I just want something simple → use Cuts, LD, Fisher
- Methods for complex problems → use BDT, MLP, SVM

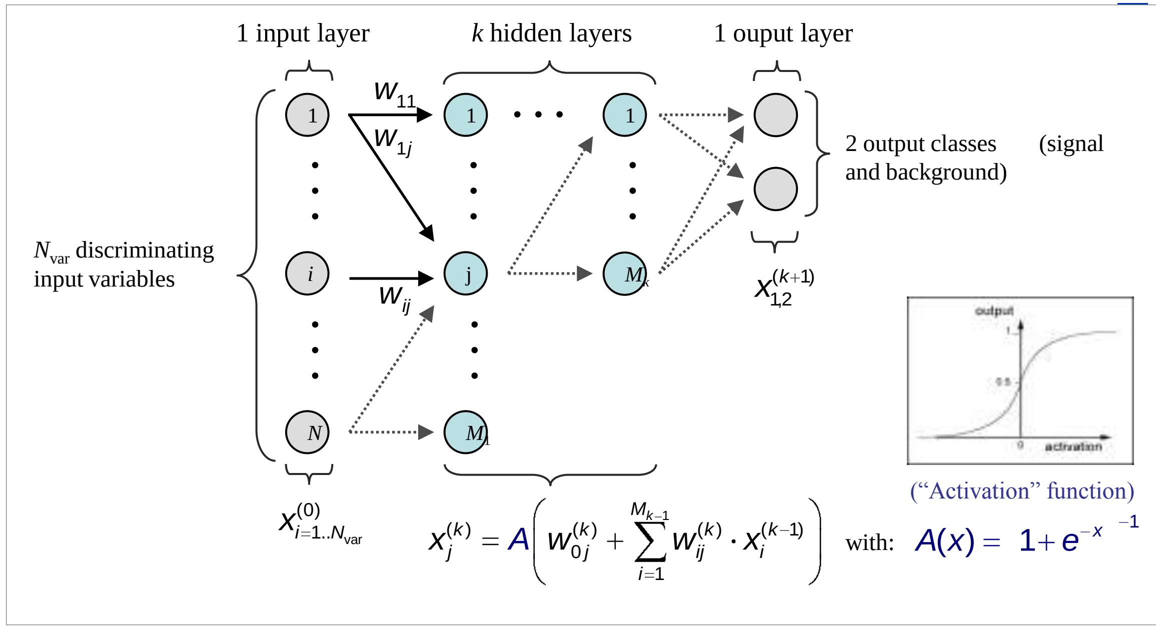
List of acronyms:
BDT = boosted decision tree, see manual page 103
ANN = articifical neural network
MLP = multi-layer perceptron, a specific form of ANN, also the name of our flagship ANN, manual p. 92
FDA = functional discriminant analysis, see manual p. 87
LD = linear discriminant , manual p. 85
SVM = support vector machine, manual p. 98 , SVM currently available only for classification
Cuts = like in “cut selection“, manual p. 56
Fisher = Ronald A. Fisher, classifier similar to LD, manual p. 83

# Artificial Neural Networks

- Modelling of arbitrary nonlinear functions as a nonlinear combination of simple „neuron activation functions“

- Advantages:
  - very flexible, no assumption about the function necessary
- Disadvantages:
  - „black box“
  - needs tuning
  - seed dependent

Feed-forward Multilayer Perceptron

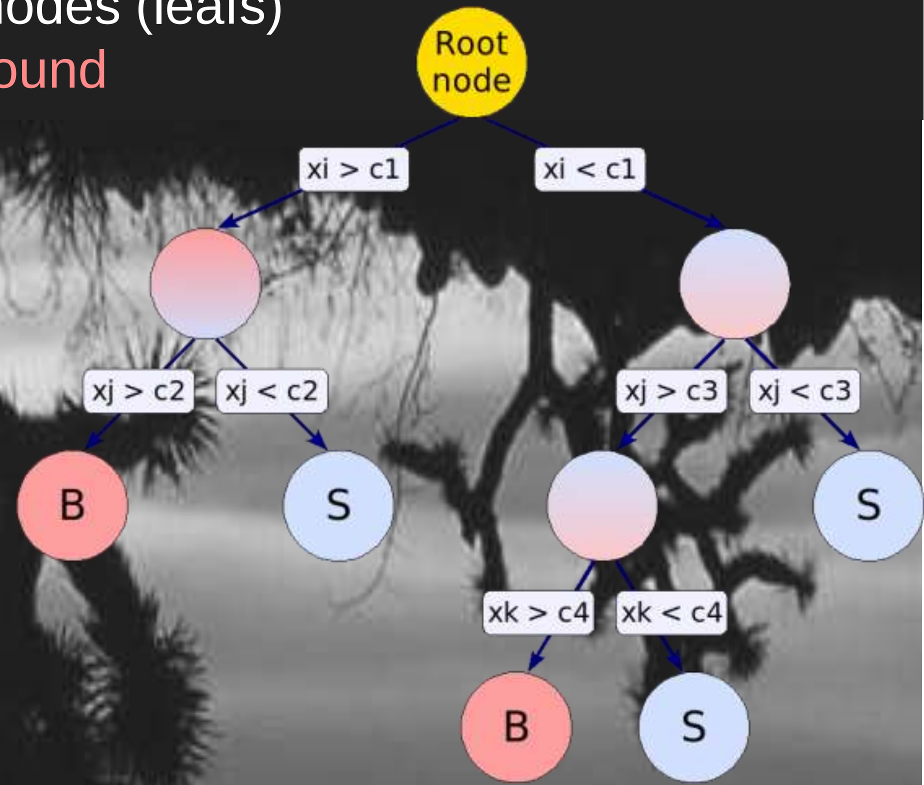



| Performance | | Speed | | Robustness | | Curse of Dim. | Transparency | Regression | |
|---|---|---|---|---|---|---|---|---|---|
| No/linear correlations | Nonlinear correlations | Training | Response | Overtraining | Weak input vars | | | 1D | multi D |
| ☺ | ☺ | 😐 | ☺ | 😐 | 😐 | 😐 | ☹ | ☺ | ☺ |

# Boosted Decision Trees

# Boosted Decision Trees

Decision Tree: Sequential application of cuts splits the data into nodes, where the final nodes (leafs) classify an event as signal or background

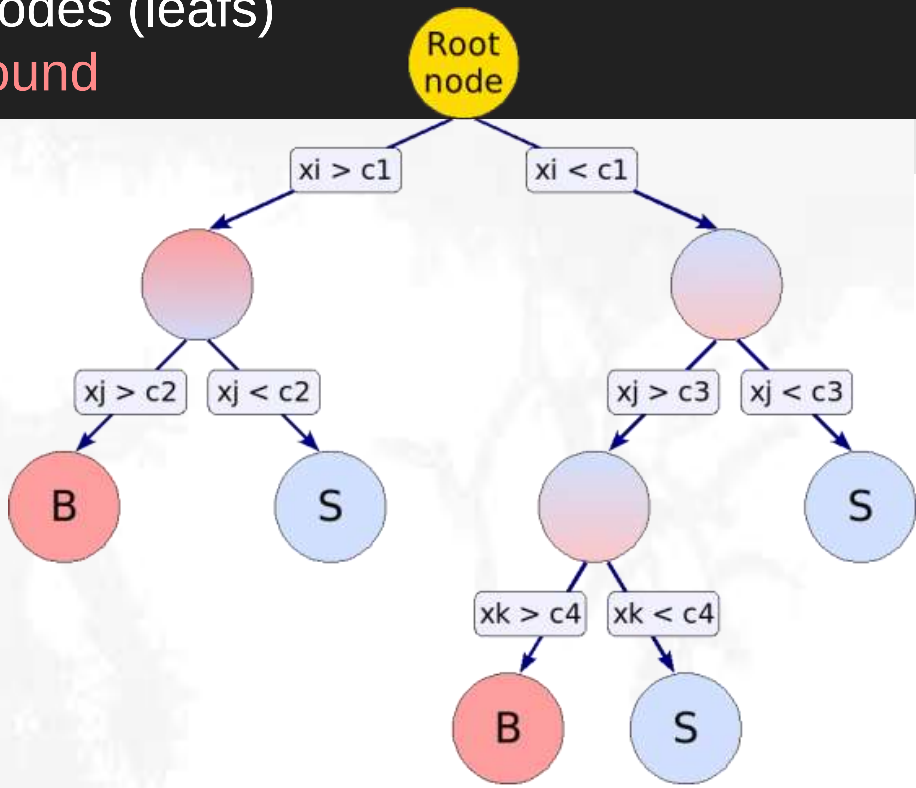


# Boosted Decision Trees

<u>Decision Tree:</u> Sequential application of cuts splits the data into nodes, where the final nodes (leafs) classify an event as signal or background

Single trees were used in general "data-mining" applications, less known in (High Energy) Physics

- similar to "simple Cuts": each leaf node is a set of cuts. → many boxes in phase space attributed either to signal or background.
- independent of monotonous variable transformations, immune against outliers
- weak variables are ignored (and don't (much) deteriorate performance)




<u>Boosted Decision Trees (1996):</u> combine a whole forest of Decision Trees, derived from the same sample, e.g. using different event weights.

→ became popular in HEP since MiniBooNE, B.Roe et.a., NIM 543(2005)

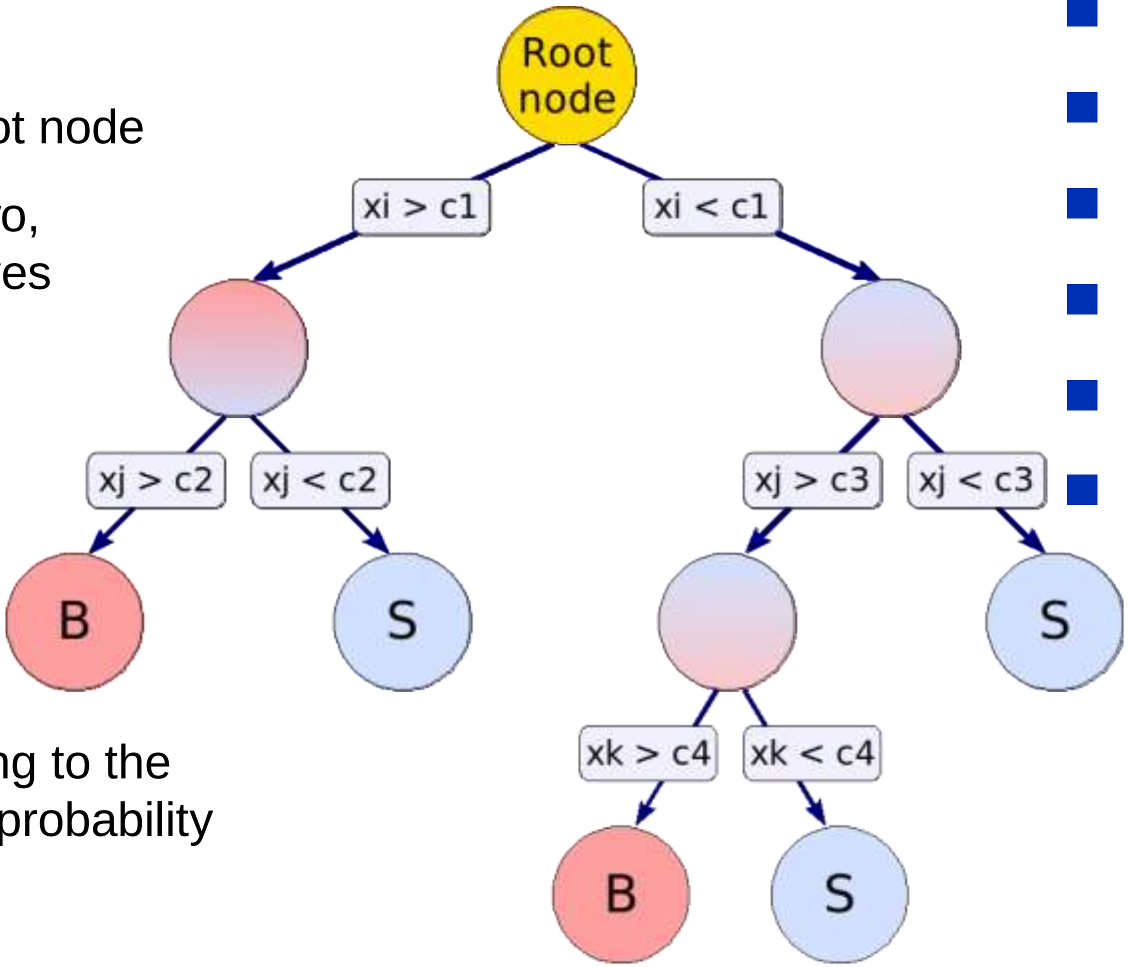
# Growing a Decision Tree

start with training sample at the root node

split training sample at node into two, using a cut in the variable that gives best separation gain

continue splitting until:
- minimal #events per node
- maximum number of nodes
- maximum depth specified

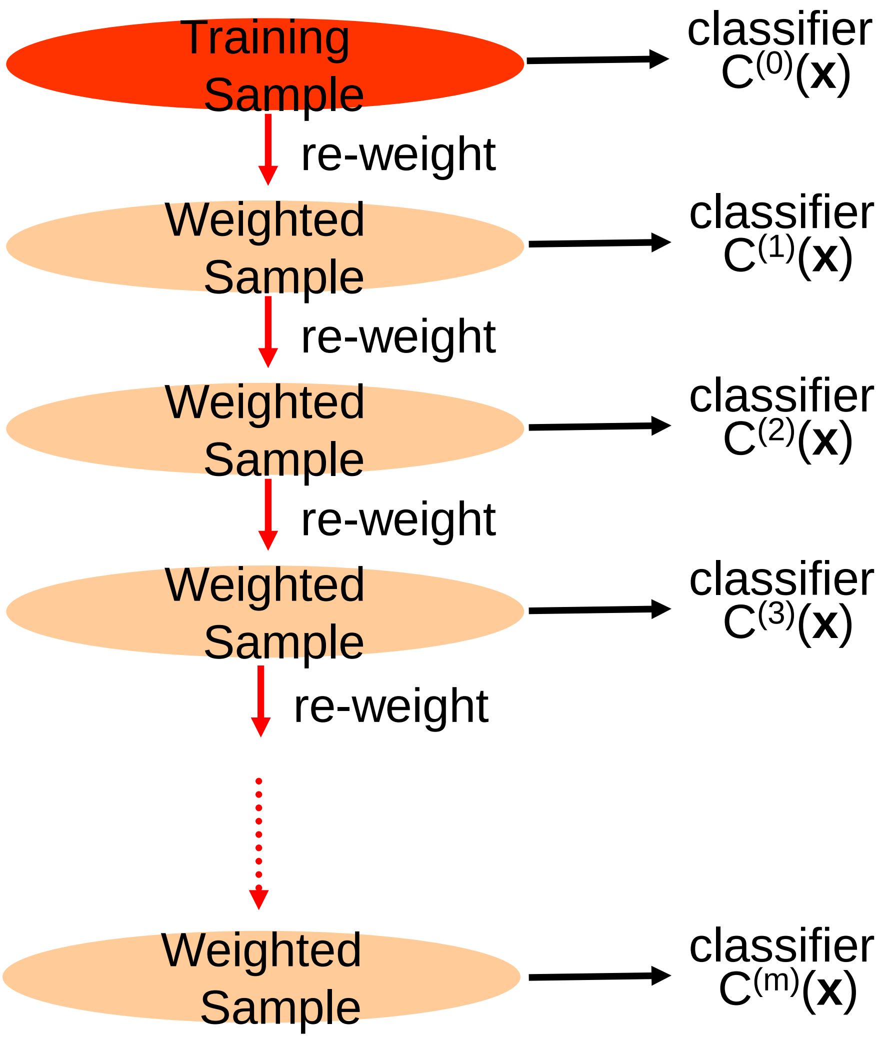
leaf-nodes classify as S,B according to the majority of events or give a S/B probability




Why no multiple branches (splits) per node ?

→ Fragments data too quickly; also: multiple splits per node = series of binary node splits

# Adaptive Boosting (AdaBoost)




AdaBoost re-weights events misclassified by previous classifier by:

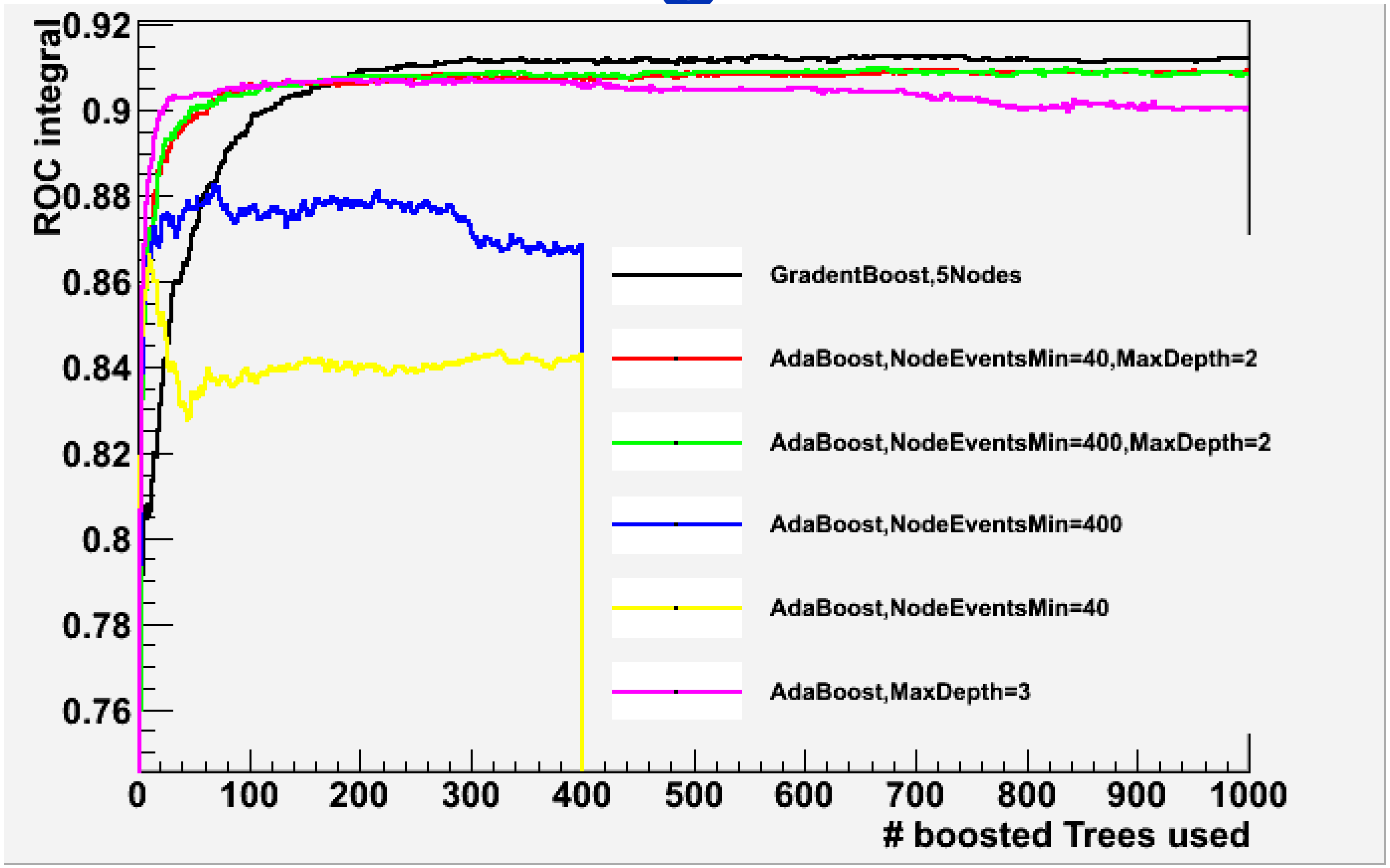
$$\frac{1-f_{err}}{f_{err}} \quad \text{with:}$$

$$f_{err} = \frac{\text{misclassified events}}{\text{all events}}$$

AdaBoost weights the classifiers also using the error rate of the individual classifier according to:

$$y(x) = \sum_{i}^{N_{Classifier}} \log\left(\frac{1-f_{err}^{(i)}}{f_{err}^{(i)}}\right) C^{(i)}(x)$$

# Boosting at Work

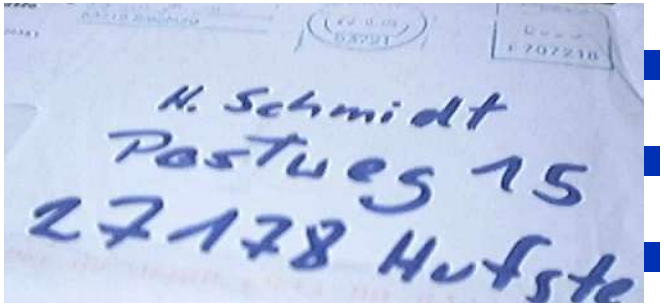



Boosting seems to work best on “weak” classifiers (i.e. small, dum trees)
Tuning (tree building) parameter settings are important
For good out of the box performance: Large numbers of very small trees

# A non-HEP example for a multivariate classification task

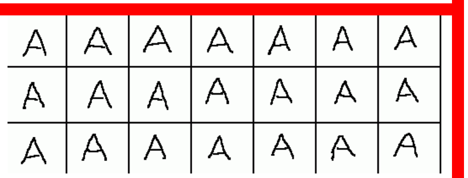
# Pattern Recognition of handwritten digits

- Automatic reading of handwritten digits for Zip-code processing in mail services (Postleitzahlerkennung)

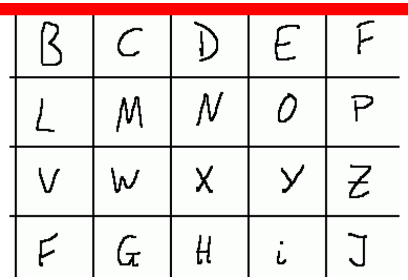



- One MVA methods for each digit: one digit is Signal, everything else is Background

Signal Sample




Background Sample



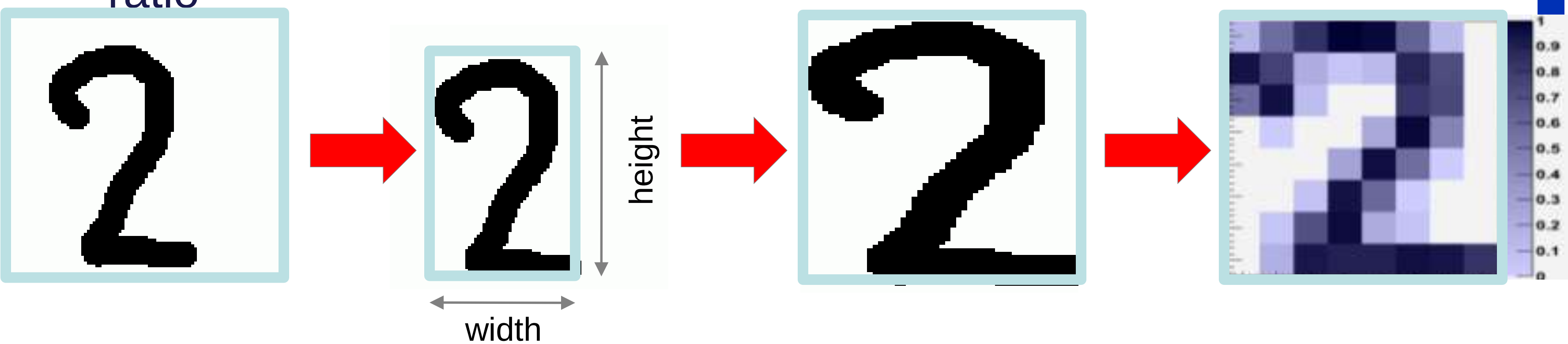

- Input values: brightness of each individual pixel

  → need to reduce number of pixels

  → preprocessing necessary

# Handwritten digits/letters

- Preprocessing:
  - [step 1] Find frame around digit, determine aspect ratio
  - [step 2] Transform to aspect ratio=1
  - [step 3] Merge pixels into 8x8 array
  - Input to multivariate analysis: 64 pixels plus the original aspect ratio




Original digit
~100 dpi

Step (1)

Step (2)
Aspect ratio=1

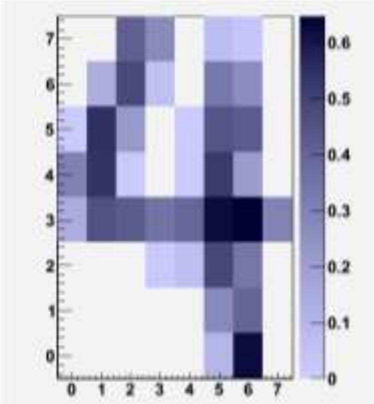
Step (3)
8x8 pixel array

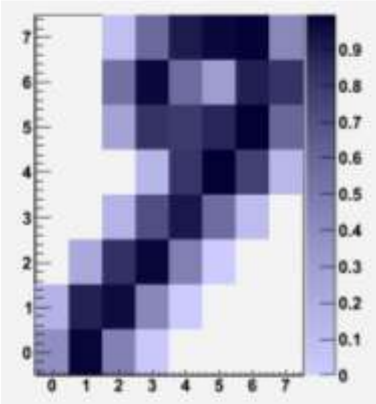
$$\text{Aspect ratio} = \frac{\text{height}}{\text{width}}$$

# Pattern Recognition of handwritten digits

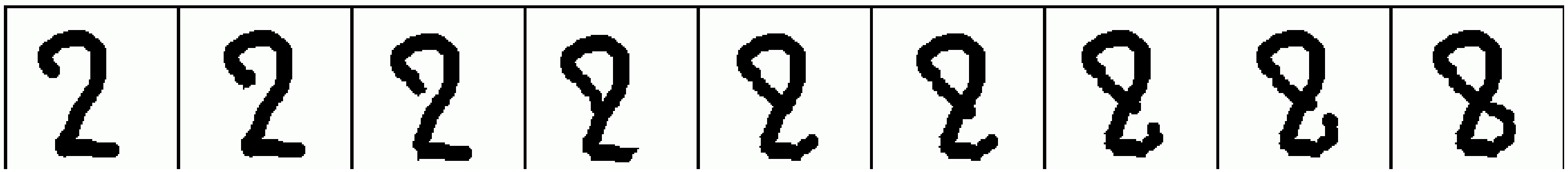
- Boosted Decision Trees with gradient boost (3000 trees)

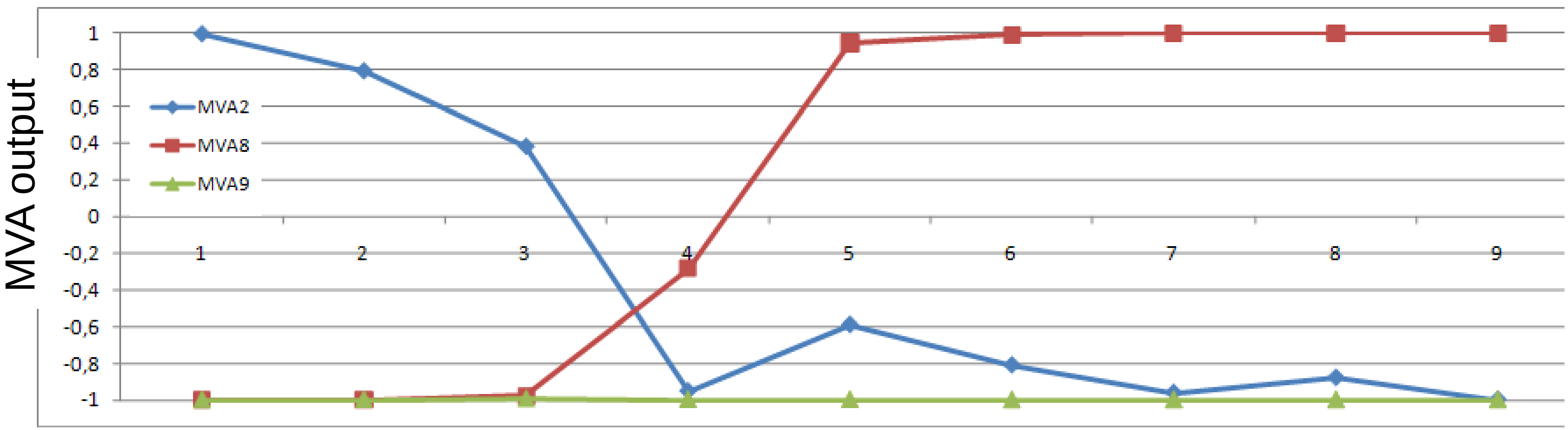
- Training: One digit is signal, all others are background
- Data sample:
  - MNIST database: 60k training digits, 10k test
  - (http://yann.lecun.com/exdb/mnist/)
  - Strict separation of test and training sample
  - persons contributing to training sample do NOT contribute to test sample (and vice versa).



# Example: stepwise morphing of “2“ into “8“






Selected as

“2“ “2“ “2“ “8“ “8“ “8“ “8“ “8“ “8“

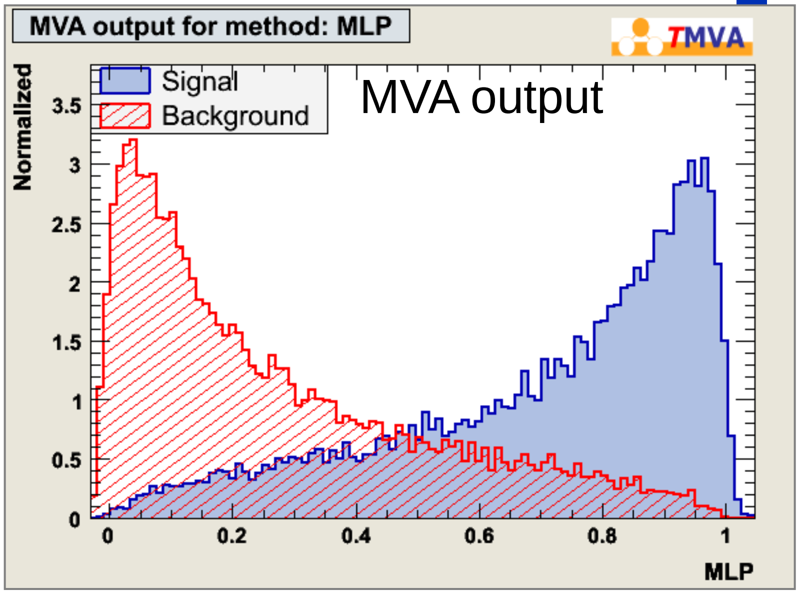
Output digit determined by MVA with largest output value

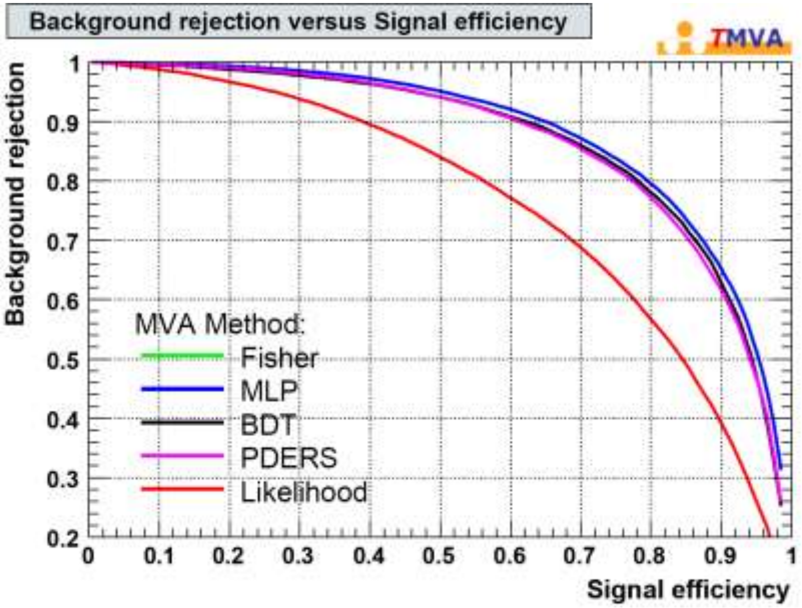
# TMVA

# **T**oolkit for **M**ulti**V**ariate **A**nalysis

# What is TMVA

- Supervised learning
- Classification and Regression tasks
- Easy to train, evaluate and compare various MVA methods
- Various preprocessing methods (Decorr.,PCA, Gauss...)
- Integrated in ROOT



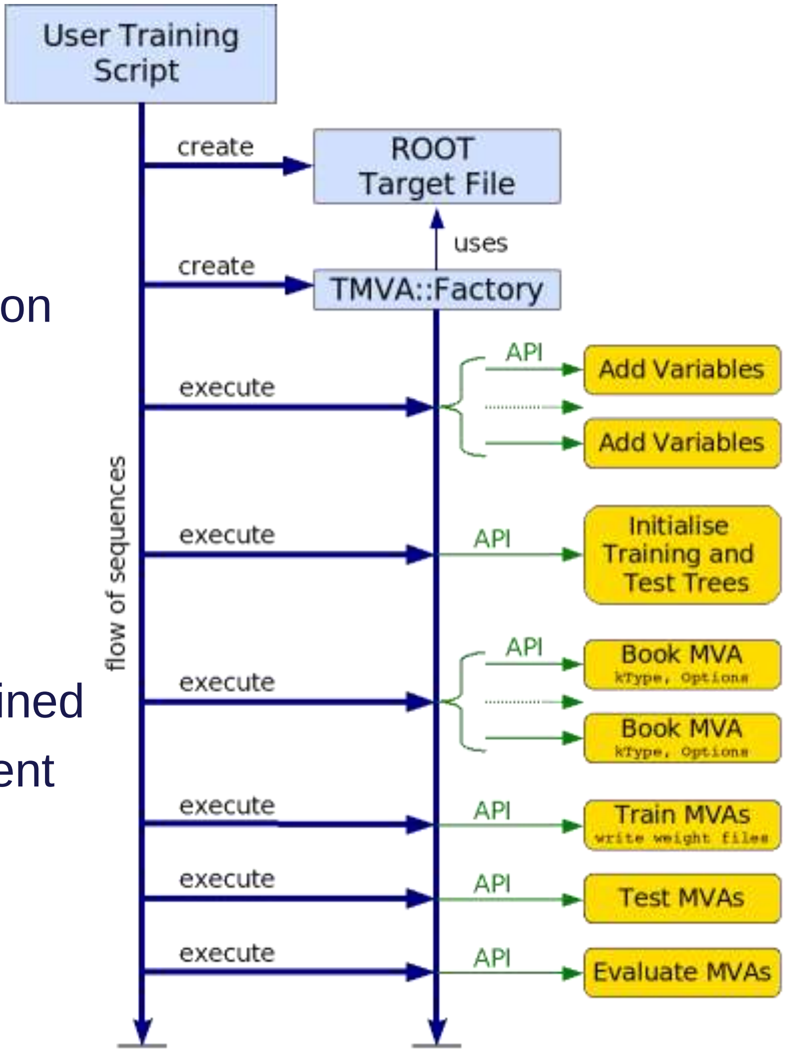
# TMVA references

- Web-Site: http://tmva.sourceforge.net/
- See also: "*TMVA - Toolkit for Multivariate Data Analysis , A. Hoecker, P. Speckmayer, J. Stelzer, J. Therhaag, E.v.Toerne, H. Voss et al.,* *arXiv:physics/0703039v5 [physics.data-an]*

- Extensively used by both ATLAS and CMS

# TMVA workflow

- Training:
  - Classification:
    Learn the features of the different event classes from a sample with known signal/background composition
  - Regression:
    Learn the functional dependence between input variables and targets
- Testing:
  - Evaluate the performance of the trained classifier/regressor on an independent test sample
  - Compare different methods
- Application:
  - Apply the classifier/regressor to real data



# Customizing the method via the option string

- Method booking

factory->BookMethod(
TMVA::Types::kBDT, “myBDT",
“**BoostType=Grad:SeparationType=GiniIndex:Ntrees=500**“);

- Read description of method in the manual.
- Choose the number of defining parameters according to data size and number of variables.

BDT option table (from manual)

8.12 Boosted Decision and Regression Trees 107

| Option | Array | Default | Predefined Values | Description |
|---|---|---|---|---|
| NTrees | – | 200 | – | Number of trees in the forest |
| BoostType | – | AdaBoost | AdaBoost, Bagging, RegBoost, AdaBoostR2, Grad | Boosting type for the trees in the forest |
| AdaBoostR2Loss | – | Quadratic | Linear, Quadratic, Exponential | Loss type used in AdaBoostR2 |
| UseBaggedGrad | – | False | – | Use only a random subsample of all events for growing the trees in each iteration. (Only valid for GradBoost) |
| GradBaggingFraction | – | 0.6 | – | Defines the fraction of events to be used in each iteration when UseBaggedGrad=kTRUE. |
| Shrinkage | – | 1 | – | Learning rate for GradBoost algorithm |
| AdaBoostBeta | – | 1 | – | Parameter for AdaBoost algorithm |
| UseRandomisedTrees | – | False | – | Choose at each node splitting a random set of variables |
| UseNvars | – | 4 | – | Number of variables used if randomised tree option is chosen |
| UseNTrainEvent | – | N | – | Number of Training events used in each tree building if randomised tree option is chosen |
| UseWeightedTrees | – | True | – | Use weighted trees or simple average in classification from the forest |
| UseYesNoLeaf | – | True | – | Use Sig or Bkg categories, or the purity=S/(S+B) as classification of the leaf node |
| NodePurityLimit | – | 0.5 | – | In boosting/pruning, nodes with purity > NodePurityLimit are signal; background otherwise. |
| SeparationType | – | GiniIndex | CrossEntropy, GiniIndex, GiniIndexWithLaplace, MisClassificationError, SDivSqrtSPlusB, RegressionVariance | Separation criterion for node splitting |

Option Table 21: Configuration options reference for MVA method: *BDT*. Values given are defaults. If predefined categories exist, the default category is marked by a '⋆'. The options in Option Table 9 on page 59 can also be configured. The table is continued in Option Table 22.

# A complete TMVA training/testing session

```
void TMVAnalysis( )
{
  TFile* outputFile = TFile::Open( "TMVA.root", "RECREATE" );

  TMVA::Factory *factory = new TMVA::Factory( "MVAnalysis", outputFile,"!V");

  TFile *input = TFile::Open("tmva_example.root");

  factory->AddVariable("var1+var2", 'F');
  factory->AddVariable("var1-var2", 'F');   //factory->AddTarget("tarval", 'F');

  TTree* dataTree = (TTree*) input->Get("TreeS");
  double coeffA = 1.0, coeffB = 0.34 coeffC = …; //set coefficients
  factory->AddTree (dataTree, “Signal”,           1.,        “m> signalLow && m<signalHigh”);  // Region A
  factory->AddTree (dataTree, “Background”, weightB, “m> bg1Low && m<bg1High”); // Region B
  factory->AddTree (dataTree, “Background”, weightC, “m> bg2Low && m<bg2High”); // Region C

  factory->PrepareTrainingAndTestTree( "", "",  "NormMode=None");

  factory->BookMethod( TMVA::Types::kMLP, "MLP",
  "!V:NCycles=200:HiddenLayers=N+1,N:TestRate=5" );

  factory->TrainAllMethods();
  factory->TestAllMethods();
  factory->EvaluateAllMethods();
  outputFile->Close();
  delete factory;
}
```

Create Factory

Add variables/ targets

Initialize Trees

Book MVA methods

Train, test and evaluate

# How to obtain signal and background samples for training

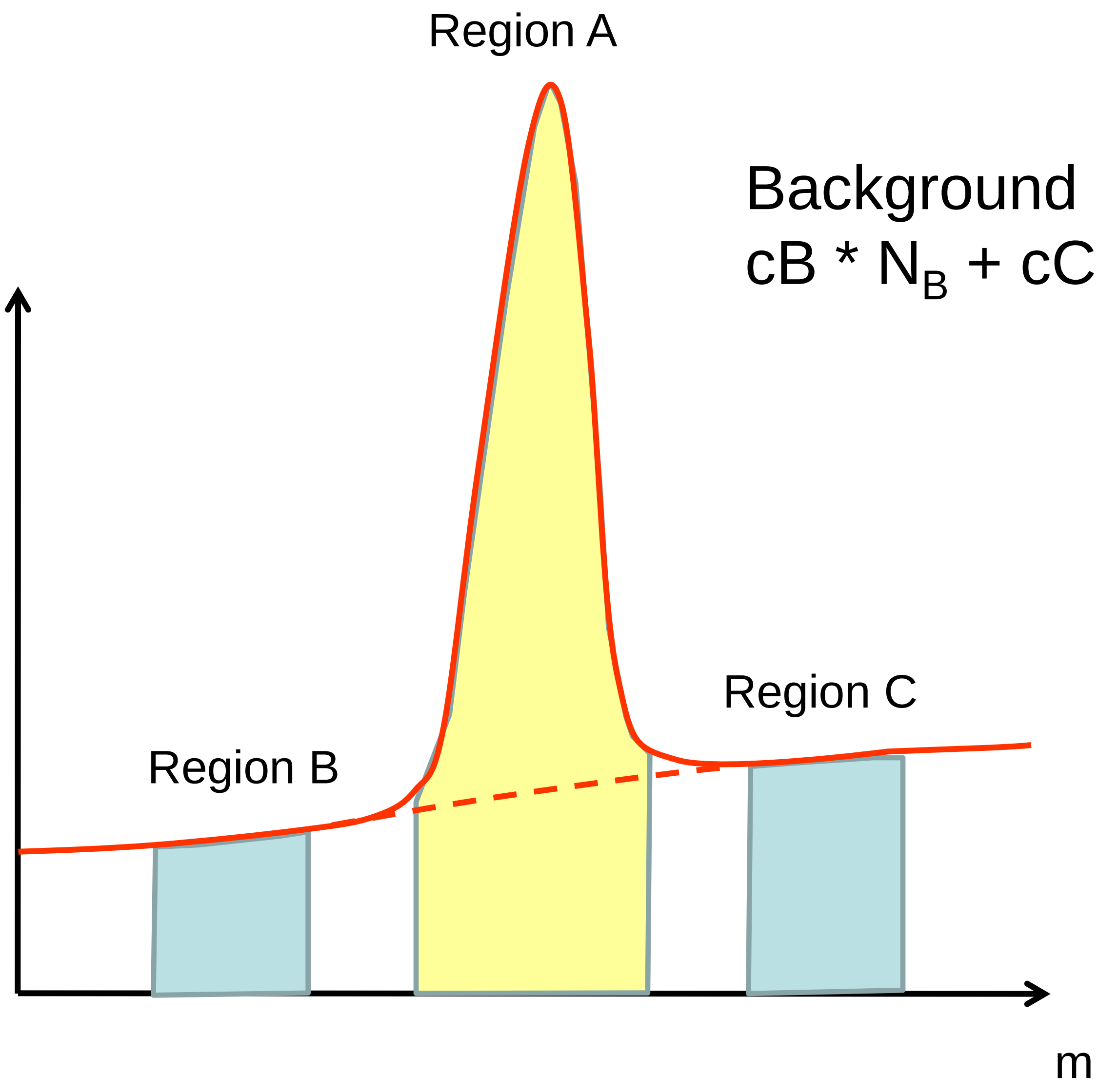
# Signal and background samples for training

- What works for a counting analysis usually works for a MVA too.
- Examples:
  - Monte Carlo
  - Sidebands (also ABCD method)
  - Event Crossing

} works with data

# Example Analysis
# Sideband method

# Sideband method with TMVA




Background in Region A = $cB * N_B + cC * N_C$

# The tutorial

# Exercise 1: Getting started

- Go to tutorial web page
  http://www.uni-bonn.de/~etoerne/tmva/

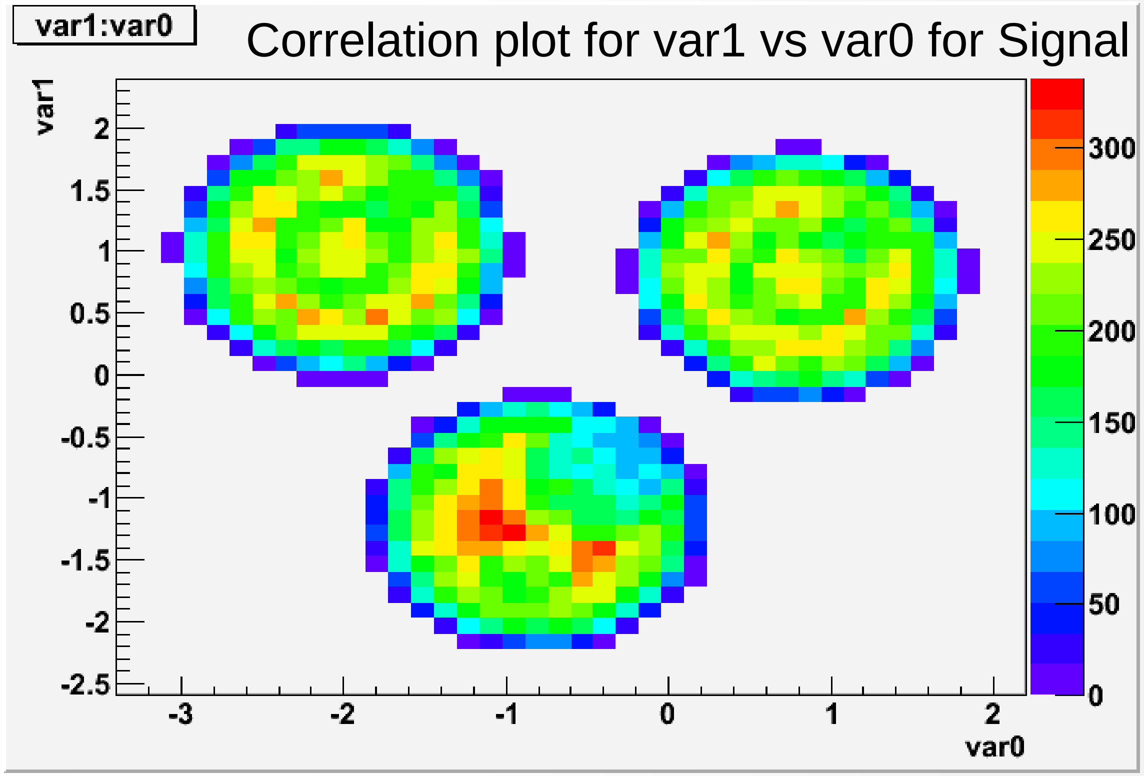
- `cd $HOME`
- `cp -r $ROOTSYS/tmva/test/ tmvatest`
- `cd tmvatest`

- Run TMVAClassification.C and TMVAClassificationApplication.C

- All information+data provided on the web page

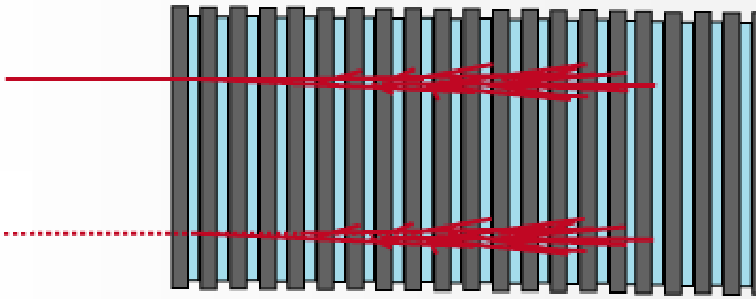
# Exercise 2

- Classification Analysis on Data "testData.root" on the tutorial page
- 3-dimensional data with complex signal and flat background
- Task: Find best classification result (BDT vs. Likelihood)



# Exercise 3: Measuring calorimeter energy

- **Regression analysis: estimate of observable (target) based on input variables.**
- **data represent measurements in a toy-calorimeter.**
- **target to be estimated: energy of calo cluster.**
- **Calorimeter is segmented**
  - five thin layers ("EM-CALO")
  - followed by eight thicker layers



- **Calorimeter is imperfect**
  - leakage at the end of the calorimeter
  - dead regions
  - non-compensation.
- **data are from jets and single particles.**
- **Always one cluster per event.**

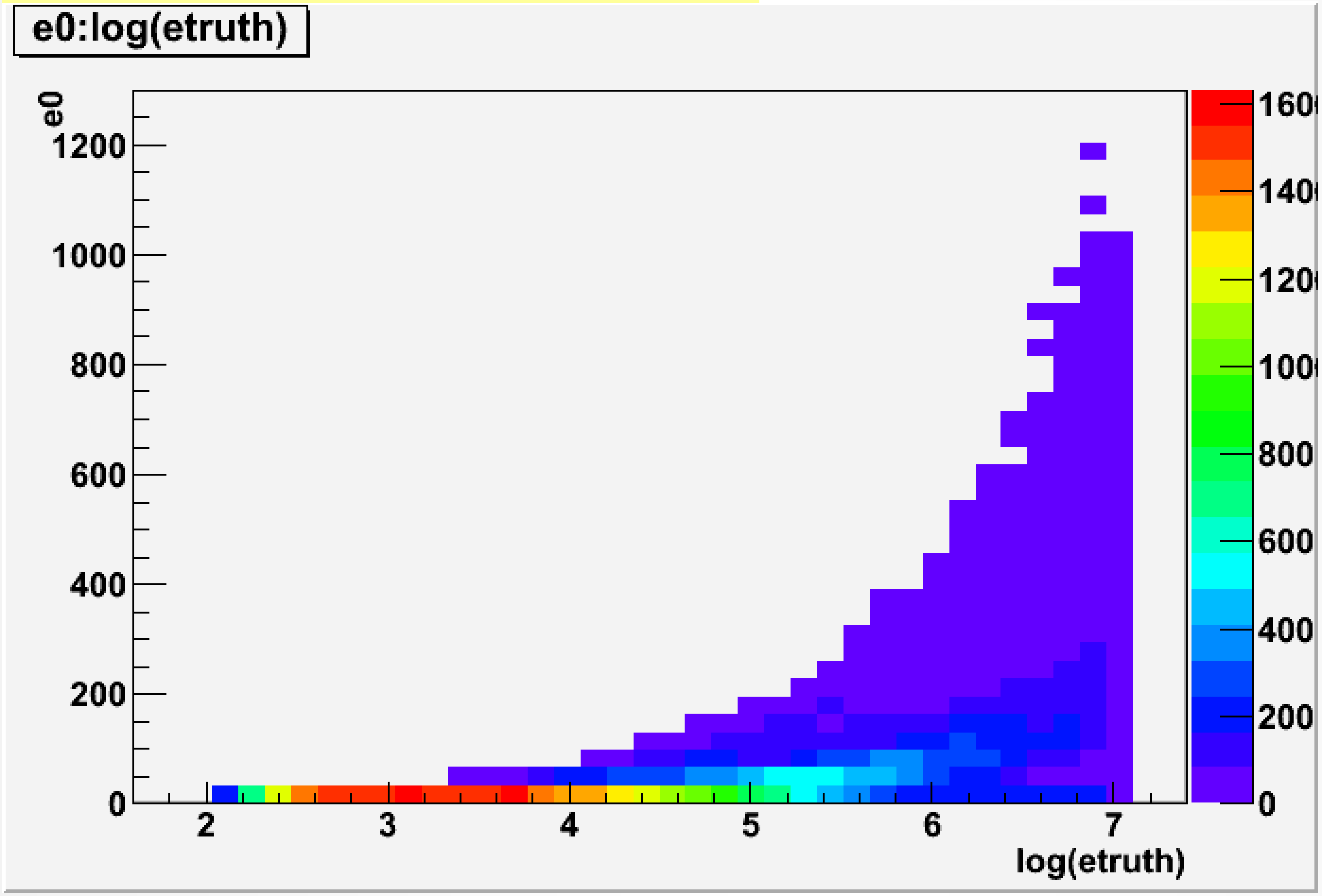
# Exercise 3: Measuring calorimeter energy

- **Variables:**
  - Energy in each layer: e0, e1, … e12. (Given in GeV)
  - Sum over all layers: esum
  - The true energy deposition: etruth
  - Cluster center-of-gravity in eta: eta, and phi: phi
  - Cluster centroid in layer 0 in eta and phi: eta0, phi0

- **Either use etruth or etruth/esum as target.**

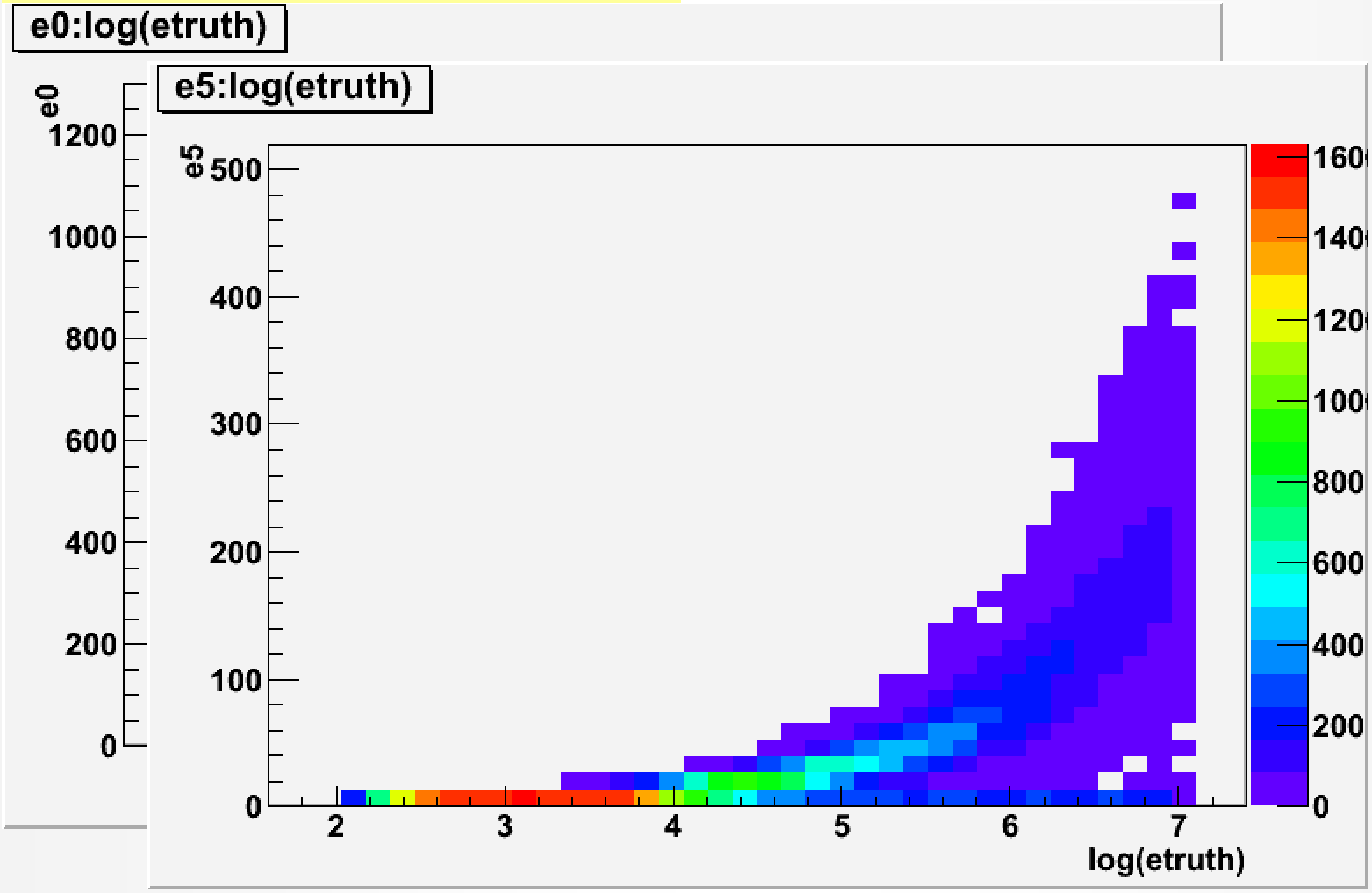
# Exercise 3: Measuring calorimeter energy

Energy in first layer vs true energy



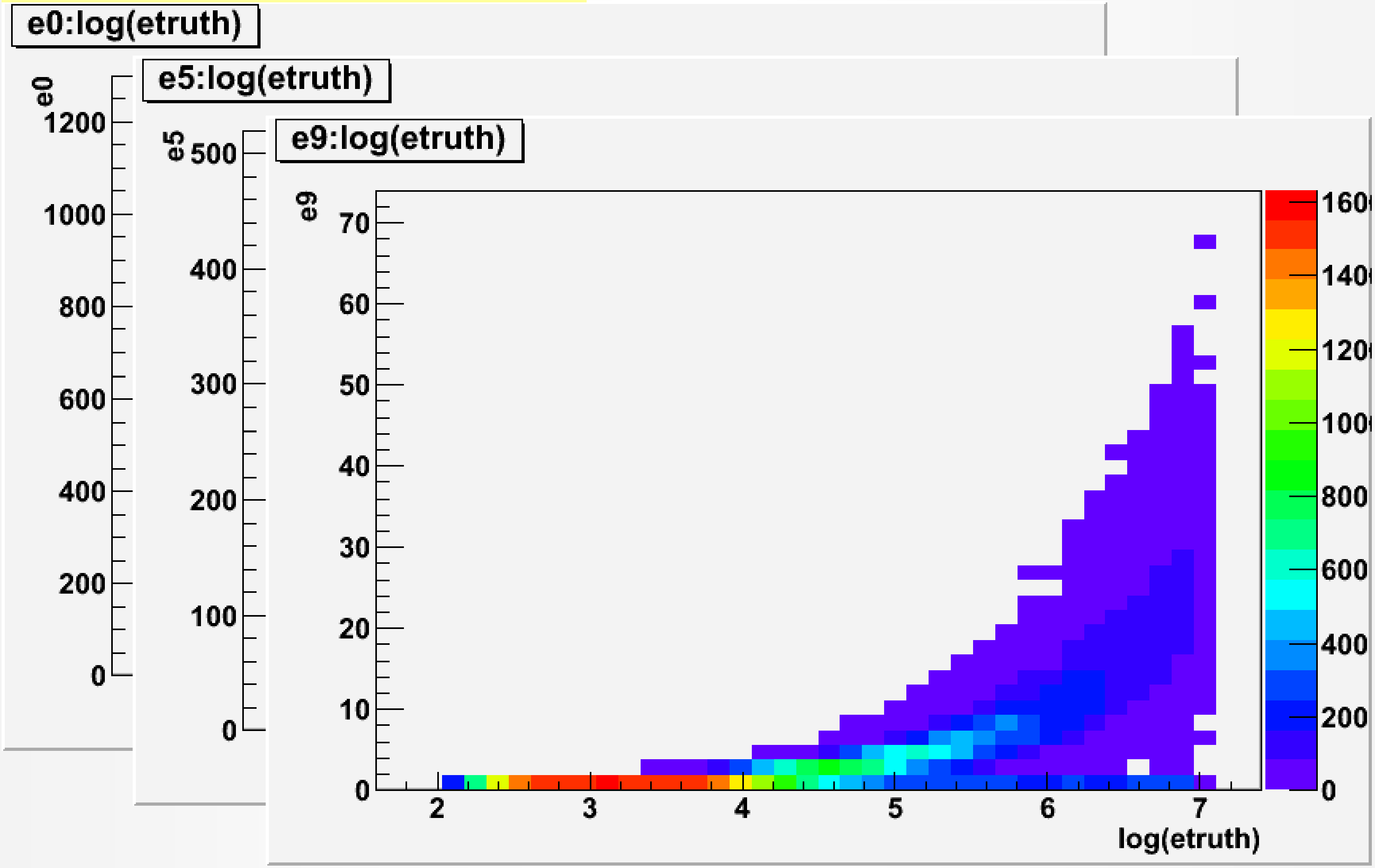
# Exercise 3: Measuring calorimeter energy

Energy in layer 5 vs true energy

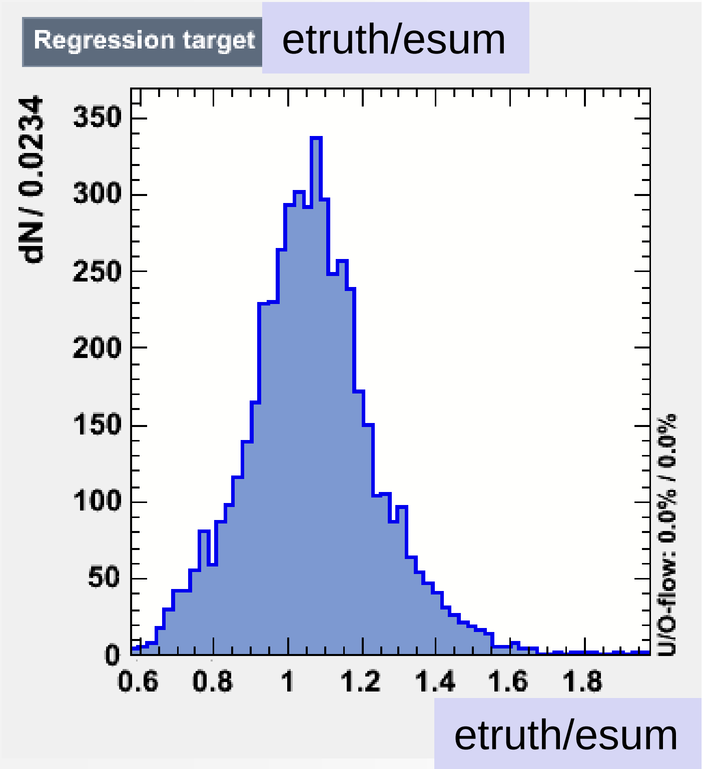


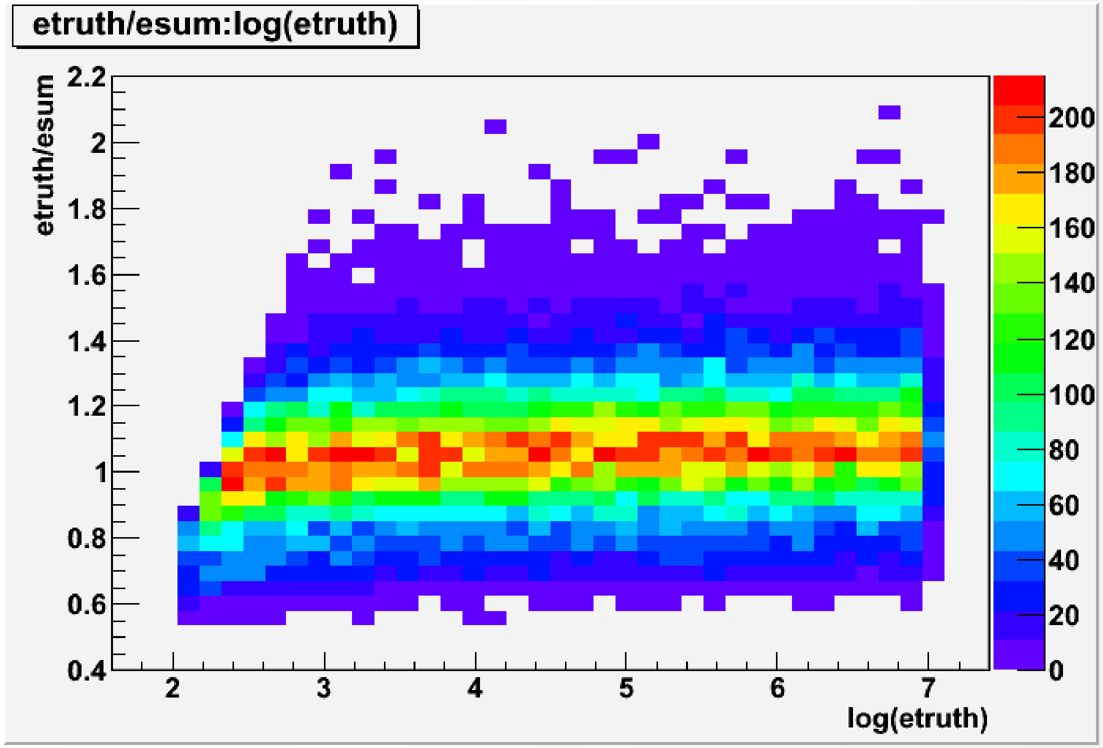
# Exercise 3: Measuring calorimeter energy

Energy in layer9 vs true energy



# Exercise 3: Measuring calorimeter energy




Average ratio < etruth/esum > = 1.06

Standard deviation of ratio etruth/esum = **0.175**

Regression-estimate (std-dev of estimate – truth) should be much less than 0.175.

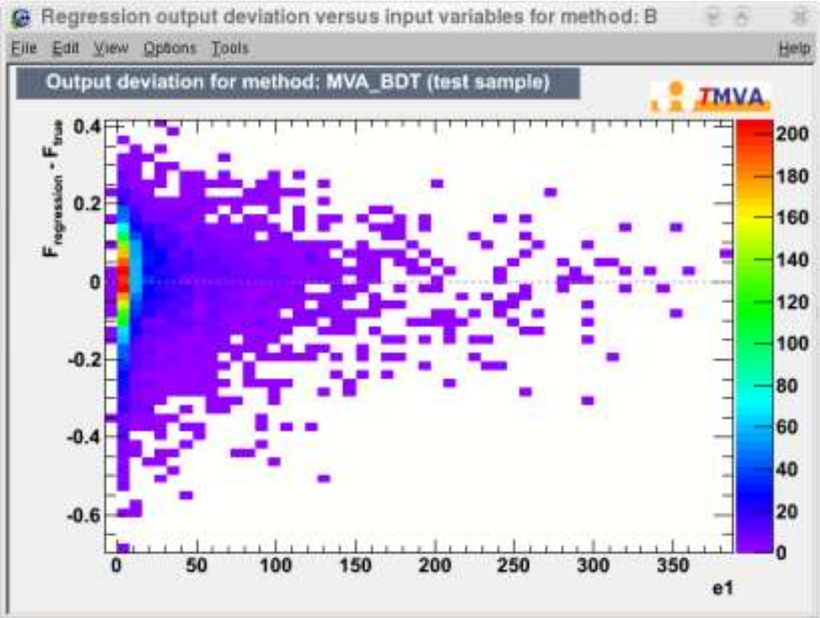
# MVA methods for regression

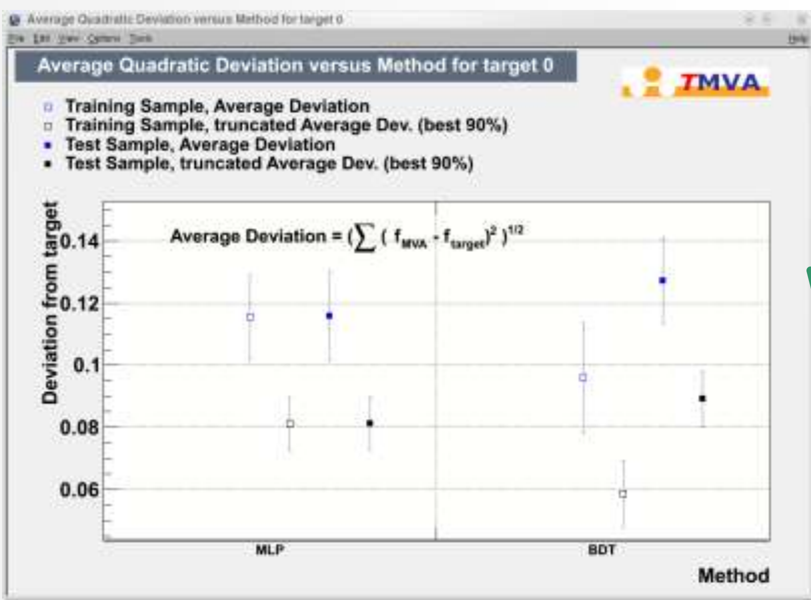
Several TMVA methods are still under development for regression

For this exercise, consider to use:

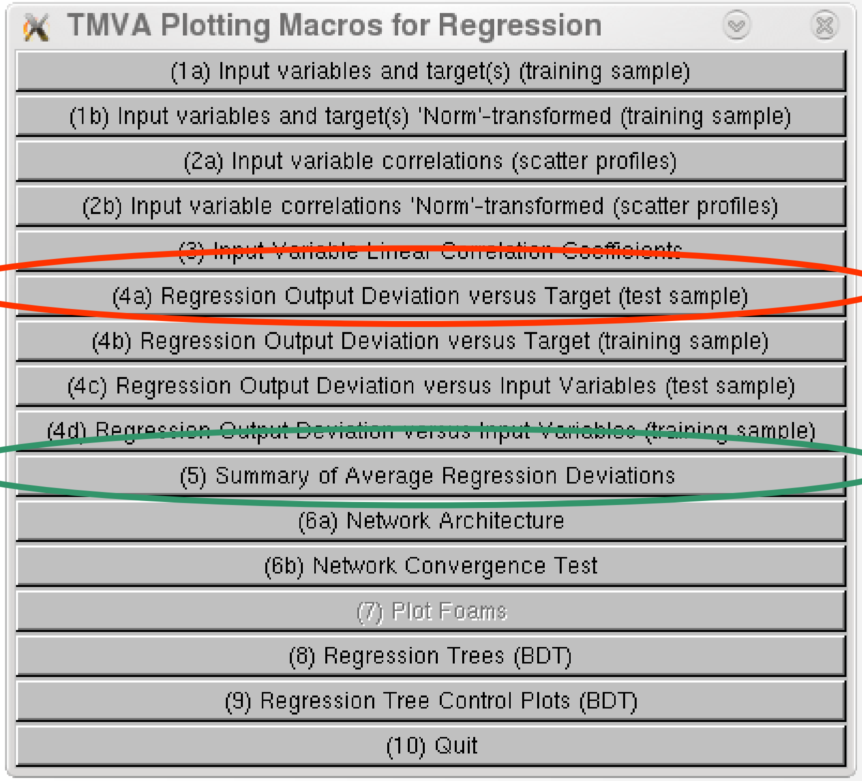
- MLP with BFGS training (option TrainingMethod=BFGS)
- BDT with BoostMethod=Grad
- PDEFoam
- FDA

# Regression macros



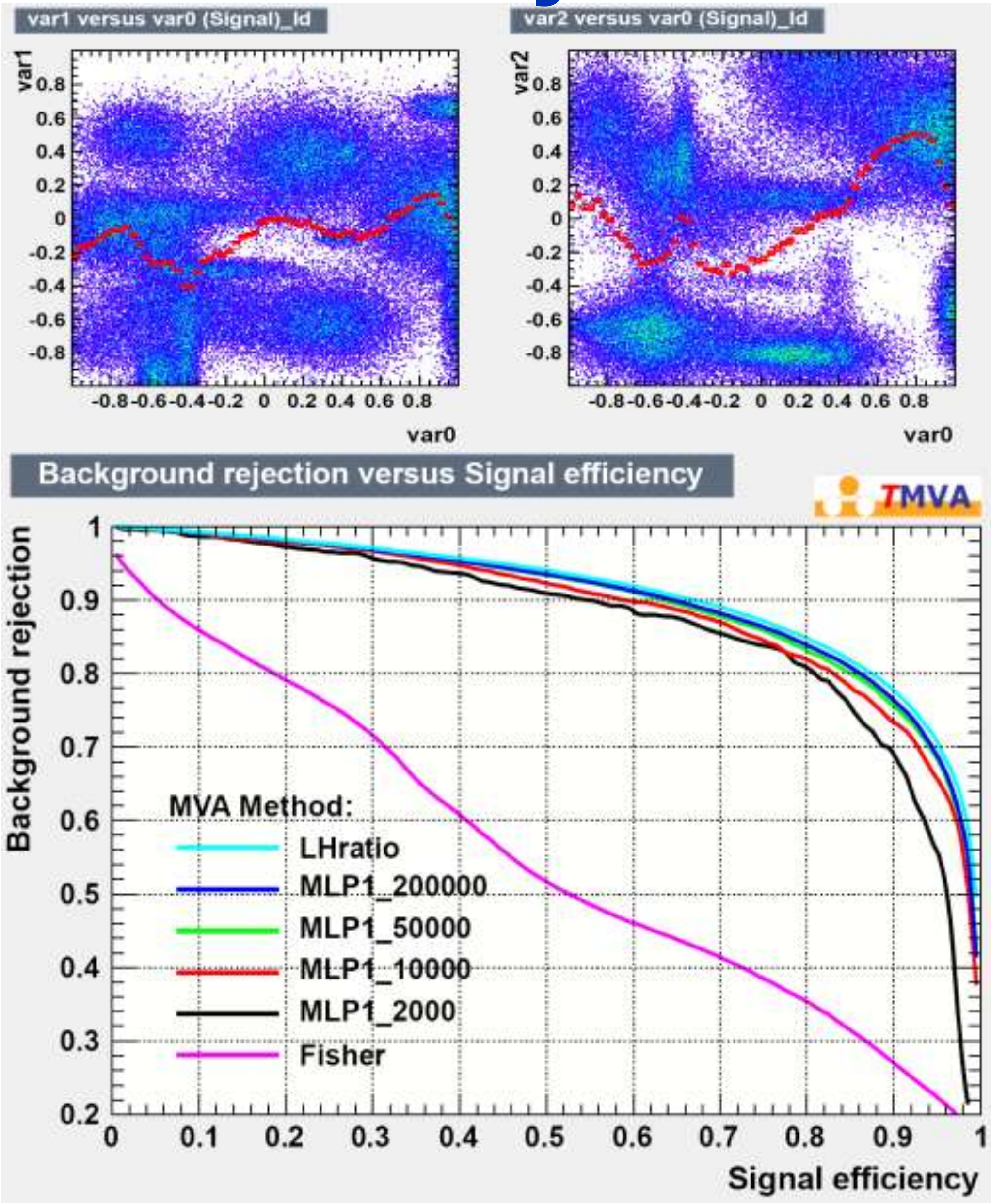

TMVARegGui.C



# BACKUP

# Performance with toy data

- 3-dimensional distribution
  - Signal: sum of Gaussians
  - Background=flat
- Theoretical limit calculated using Neyman-Pearson Lemma
- Neural net (MLP) with two hidden layers and backpropagation training. Bayesian option has little influence on high statistics training
- TMVA-ANN converges towards theoretical limit for sufficient Ntrain (~100k)



# TMVA stand-alone vs. TMVA in ROOT

**Using TMVA stand-alone**: the TMVA in ROOT is ignored. The tmva/test directory may serve as an example setup. Do not forget to run setup.sh

**Using TMVA in ROOT:** use ROOT's include and library pathes, need to modify tmva/test/Makefile

- Erase all reference to stand-alone include directories
- Replace –l TMVA.1 by –l TMVA
- You no longer need to run setup.sh
- No changes for using macros (do not source setup.sh)

**This tutorial:** We will use TMVA in ROOT

TMVA version 4.1.3, (the version included in ROOT 5.34)

# No Single Best Classifier…

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