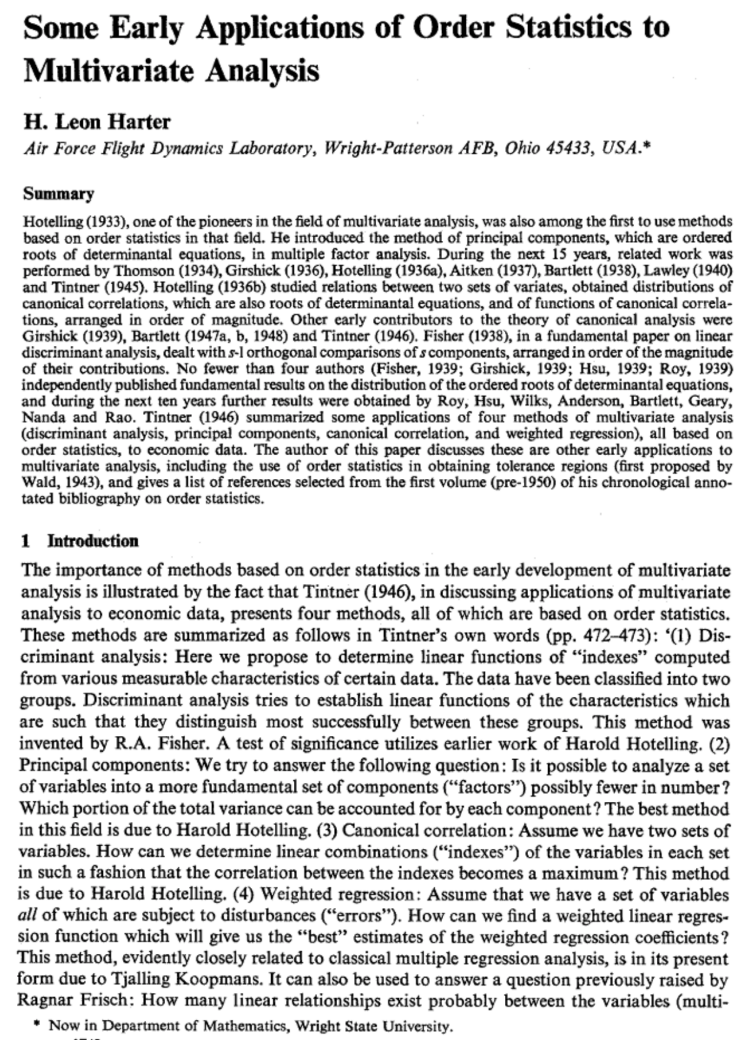
Aspects of MVA techniques

Urs Langenegger

LTP seminar
2018/04/30

- Introduction
- Boosted decision trees for $B_s^0 \to \mu^+\mu^-$
  - technicalities
  - training/optimization
  - the devil in the details
- Outlook

MVA = multivariate analysis



Some Early Applications of Order Statistics to Multivariate Analysis

H. Leon Harter
*Air Force Flight Dynamics Laboratory, Wright-Patterson AFB, Ohio 45433, USA.**

Summary

Hotelling (1933), one of the pioneers in the field of multivariate analysis, was also among the first to use methods based on order statistics in that field. He introduced the method of principal components, which are ordered roots of determinantal equations, in multiple factor analysis. During the next 15 years, related work was performed by Thomson (1934), Girshick (1936), Hotelling (1936a), Aitken (1937), Bartlett (1938), Lawley (1940) and Tintner (1945). Hotelling (1936b) studied relations between two sets of variates, obtained distributions of canonical correlations, which are also roots of determinantal equations, and of functions of canonical correlations, arranged in order of magnitude. Other early contributors to the theory of canonical analysis were Girshick (1939), Bartlett (1947a, b, 1948) and Tintner (1946). Fisher (1938), in a fundamental paper on linear discriminant analysis, dealt with s-1 orthogonal comparisons of s components, arranged in order of the magnitude of their contributions. No fewer than four authors (Fisher, 1939; Girshick, 1939; Hsu, 1939; Roy, 1939) independently published fundamental results on the distribution of the ordered roots of determinantal equations, and during the next ten years further results were obtained by Roy, Hsu, Wilks, Anderson, Bartlett, Geary, Nanda and Rao. Tintner (1946) summarized some applications of four methods of multivariate analysis (discriminant analysis, principal components, canonical correlation, and weighted regression), all based on order statistics, to economic data. The author of this paper discusses these are other early applications to multivariate analysis, including the use of order statistics in obtaining tolerance regions (first proposed by Wald, 1943), and gives a list of references selected from the first volume (pre-1950) of his chronological annotated bibliography on order statistics.

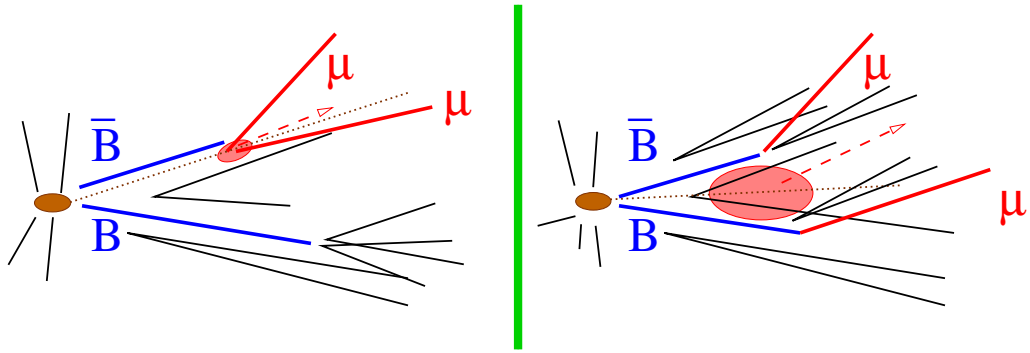
1 Introduction

The importance of methods based on order statistics in the early development of multivariate analysis is illustrated by the fact that Tintner (1946), in discussing applications of multivariate analysis to economic data, presents four methods, all of which are based on order statistics. These methods are summarized as follows in Tintner's own words (pp. 472–473): '(1) Discriminant analysis: Here we propose to determine linear functions of "indexes" computed from various measurable characteristics of certain data. The data have been classified into two groups. Discriminant analysis tries to establish linear functions of the characteristics which are such that they distinguish most successfully between these groups. This method was invented by R.A. Fisher. A test of significance utilizes earlier work of Harold Hotelling. (2) Principal components: We try to answer the following question: Is it possible to analyze a set of variables into a more fundamental set of components ("factors") possibly fewer in number? Which portion of the total variance can be accounted for by each component? The best method in this field is due to Harold Hotelling. (3) Canonical correlation: Assume we have two sets of variables. How can we determine linear combinations ("indexes") of the variables in each set in such a fashion that the correlation between the indexes becomes a maximum? This method is due to Harold Hotelling. (4) Weighted regression: Assume that we have a set of variables *all* of which are subject to disturbances ("errors"). How can we find a weighted linear regression function which will give us the "best" estimates of the weighted regression coefficients? This method, evidently closely related to classical multiple regression analysis, is in its present form due to Tjalling Koopmans. It can also be used to answer a question previously raised by Ragnar Frisch: How many linear relationships exist probably between the variables (multi-

* Now in Department of Mathematics, Wright State University.

Introduction

- Just my past experiences
  - not a lecture
- experience from $B_s^0 \to \mu^+\mu^-$
  - not 'statistics'
  - not much about the theoretical foundations (Neyman-Pearson, Bayes, . . .)

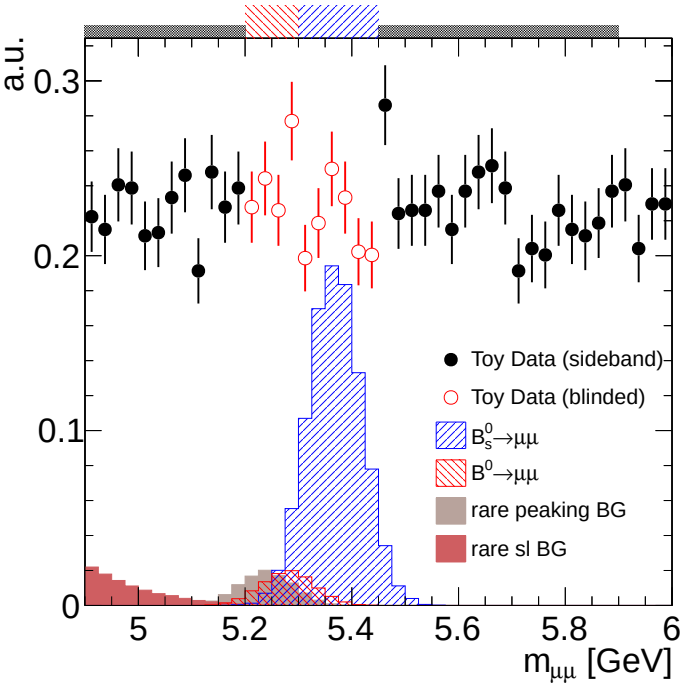


- one specific example: boosted decision trees
  - not the many alternatives, e.g. Fisher, likelihood ratios, ANN, . . .
- binary classification: signal vs background
  - not multi-class classification
  - not regression (improved calibration)
- supervised learning
  - signal from MC simulation

  what else can you do in a search?
  - background: MC/'sidebands'/event crossing

  dimuon mass outside from signal region

  capture several/different background processes



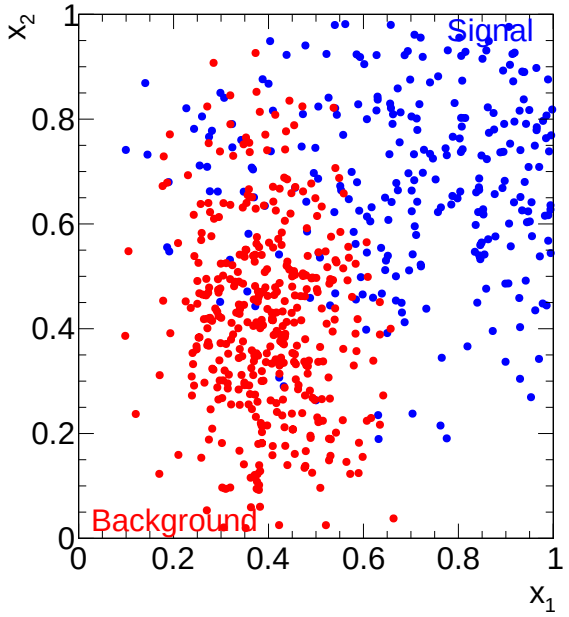
(Binary) Event classification

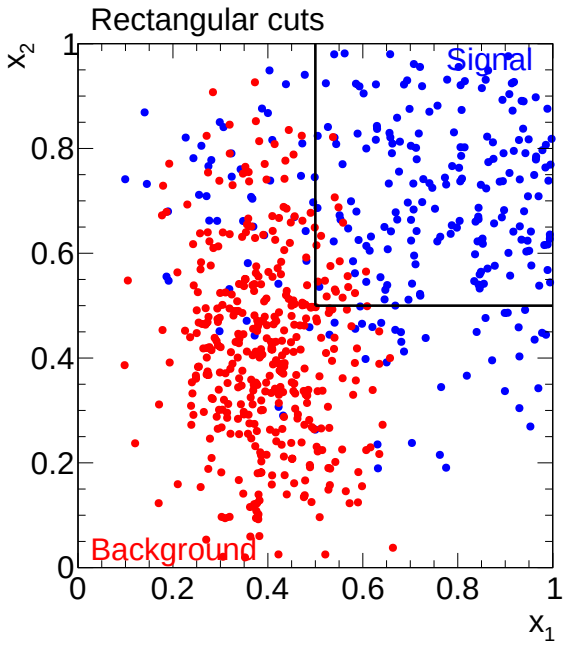
- Data set with two event types
  - ▷ signal
  - ▷ background
  - ▷ discriminating variables: x_1, x_2

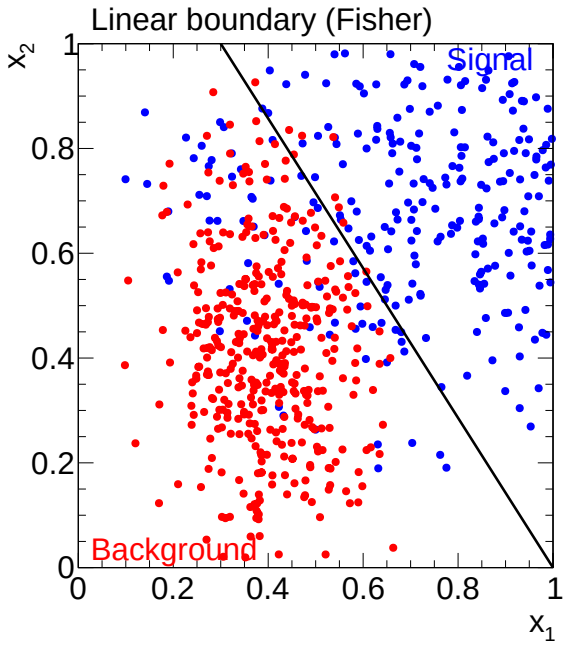
    (more in reality, but 2 are sufficient for 'multivariate' analysis)

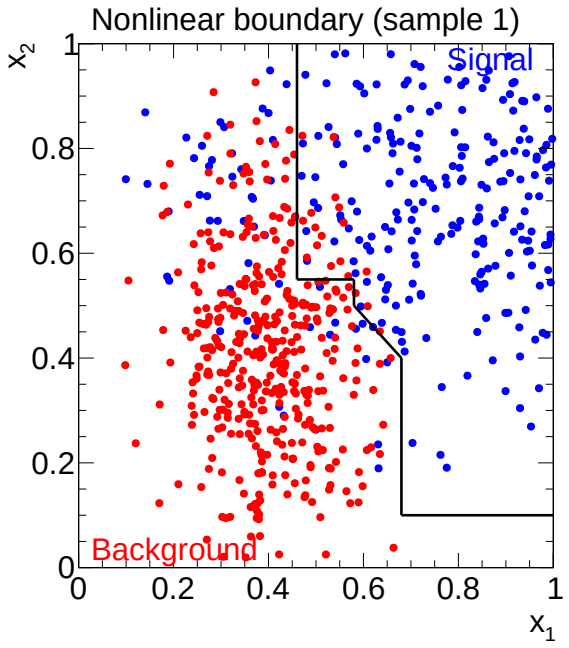
$\Rightarrow$ optimal classification algorithm?

$\rightarrow$ machine learning!









Multivariate event classification

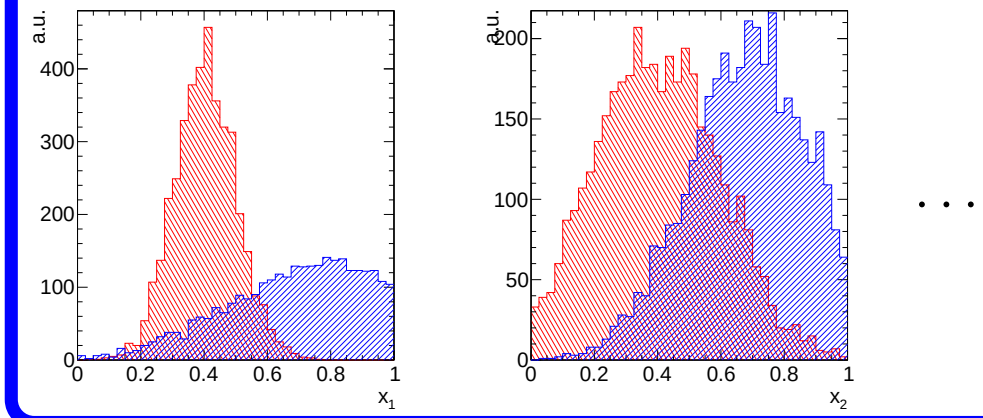
- Determine mapping from d variables to one number

  from input variables

  $$y = y(\mathbf{x})$$
  $$\mathbf{x} = \{x_1, \ldots, x_d\}$$

  find discriminating function

  $$y(\mathbf{x}) : \mathbb{R}^d \to \mathbb{R}$$

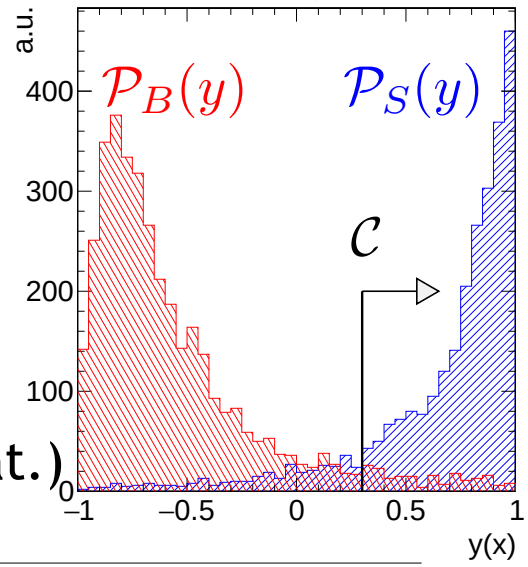


  (Probability density) Distributions of $y(\mathbf{x})$ for signal and background

  $\mathcal{P}_S(y)$ signal ('towards' $+1$)

  $\mathcal{P}_B(y)$ background ('towards' -1)

  - $y(\mathbf{x}) = const.$ defines (affine) hyperplane in $\mathbb{R}^d$



- Usage of $y(\mathbf{x})$
  - classification (statistical test)

    $y(\mathbf{x}) > \mathcal{C}$: 'signal'

    $y(\mathbf{x}) < \mathcal{C}$: 'background'

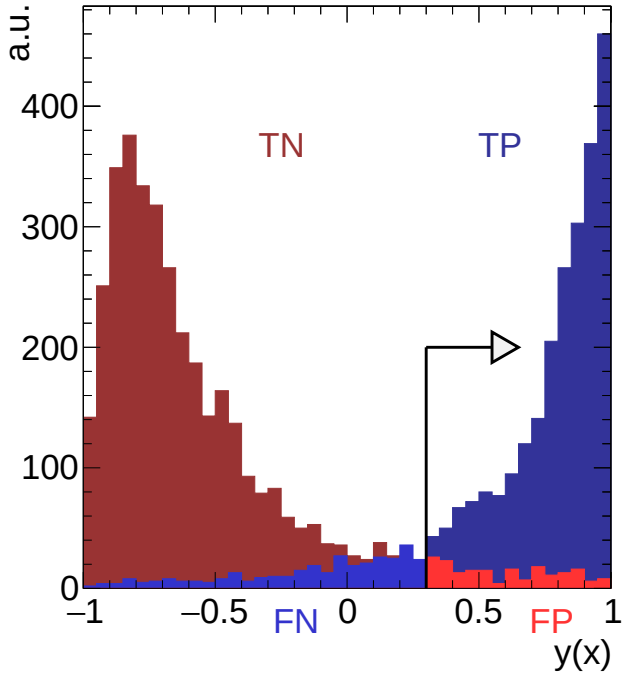
  - partitioning: categorize data ($\to$ equal signal yield/cat.)
  - fitting: move (most) signal into one place

Error $\neq$ mistake

- How to choose the 'best' MVA selection?
  - every decision carries a risk
- type-1 error: 'false positive'
  - event classified as X though it's not
  - 'fakes'
  - significance level $\alpha = \int_{y>\mathcal{C}} \mathcal{P}_B(y)dy$

    should be small (= background efficiency)
- type-2: 'false negative'
  - event not classified as X though it is
  - 'inefficiency'
  - $\beta = \int_{y<\mathcal{C}} \mathcal{P}_S(y)dy$

    $1-\beta$ is called power (= signal selection efficiency)
- 'Confusion matrix'



TN (B_{rej})	TP (S_{sel})
FN (S_{rej})	FP (B_{sel})

A type-I error is to falsely infer the existence of something that is not there, while a type-II error is to falsely infer the absence of something that is.

How to choose? (I)

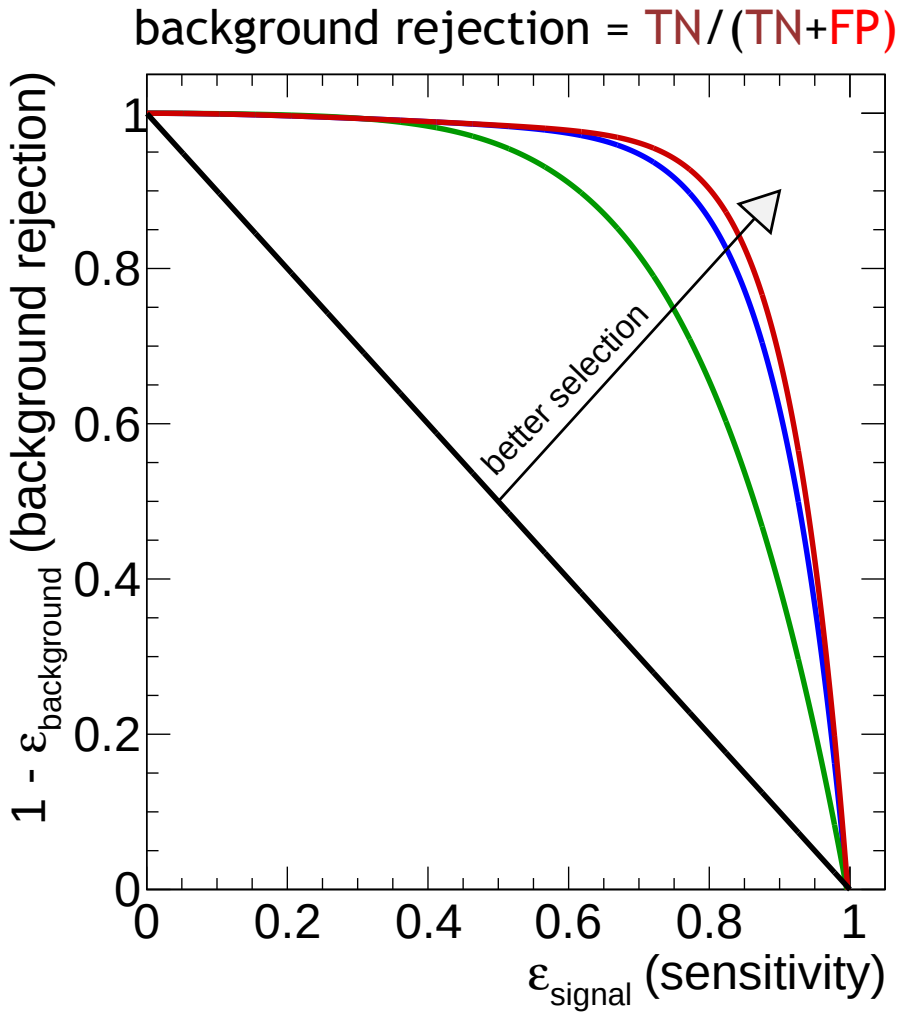
- ROC curve = Receiver Operating Characteristic curve
  - ▷ different MVA selection algorithms correspond to different curves
  - ▷ curve corresponds to different values of $\mathcal{C}$

- Popular metric
  - ▷ AUC = Area Under (roc) Curve
    between 0.5 and 1, more is better
    (Neyman-Pearson lemma: likelihood ratio test is 'best'. But . . .)

  → metric independent of $\mathcal{C}$

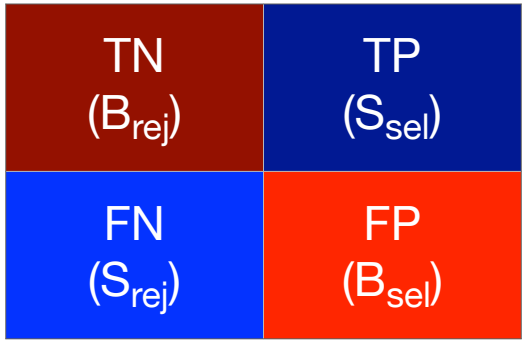
  - ▷ However critique
    ROC originated in non-physics fields
    different goals imply different tools
    n patients, have cancer? - yes or no?
    → TN important for test 'accuracy'

TN (B_{rej})	TP (S_{sel})
FN (S_{rej})	FP (B_{sel})



How to choose? (II)

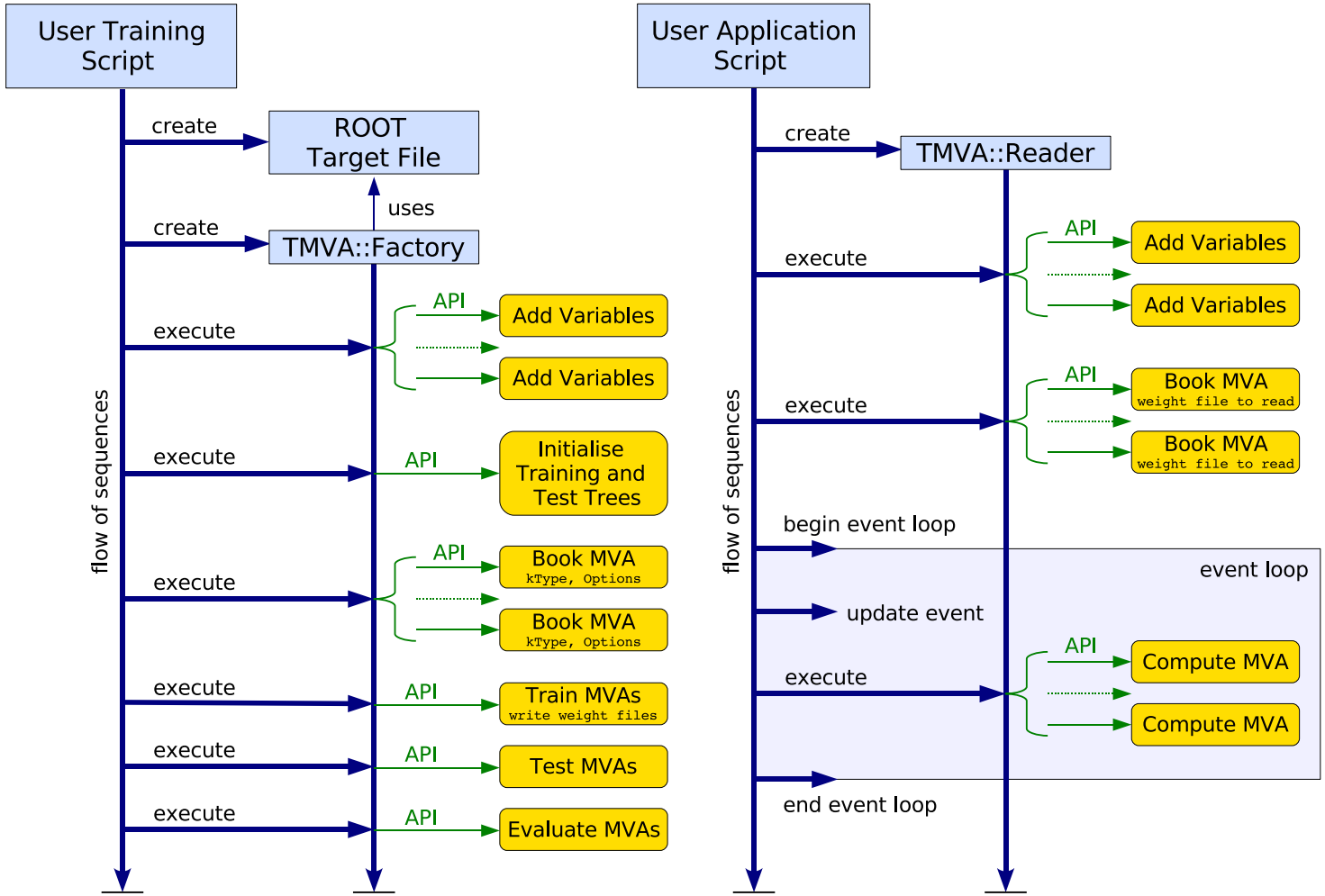
- 'We' do not care about background not entering the analysis (TN)
  - efficiency, purity, S and B require no knowledge of TN
    S = TP
    B = FP
  - Need prior probabilities (e.g. from sidebands)
    S and B abundances



- Various possibilities known/established
  - measurement: $\max(S/\sqrt{S+B})$
  - discovery: $\max(S/\sqrt{B})$
  - precision meas: $\max(p)$, where p = purity = $S/(S+B)$
  - trigger selection: $\max(\varepsilon)$, where ε = efficiency = TP/(TP+FN)
    $\max(r)$, where r = background rejection = TN/(TN+FP)
    $= 1 - \varepsilon(B)$
  - comparison of data/MC simulation
  - improved versions of the above, taking into account background error(s)

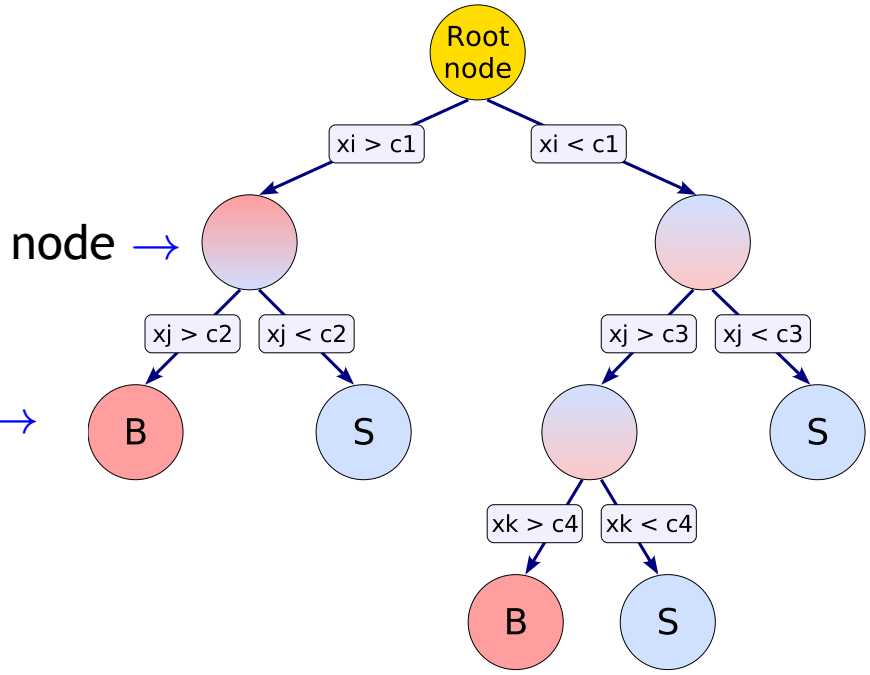
Technicalities

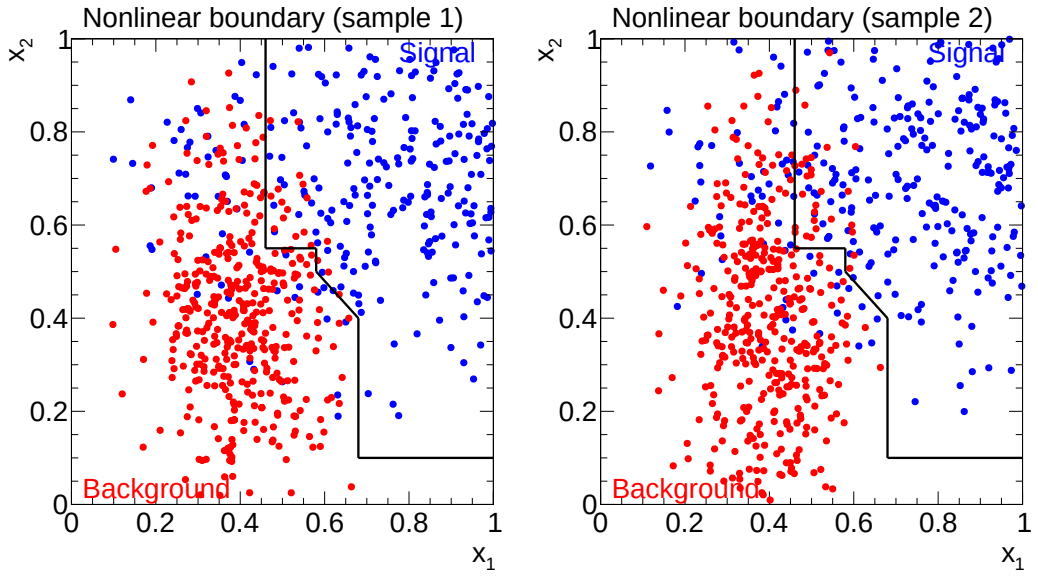
- Many packages exist that make MVA straightforward to set up
  - TMVA (integrated into ROOT)
  - SNNS, MLPFit, NEUROBAYES, StatPatternRecognition, . . .
  - TensorFlow (even in your browser :-)



Decision Tree(s)

- Sequential application of cuts splits sample into nodes
  - ▷ final nodes are classified as signal S or background B
  - ▷ variables and cuts determined by best 'separation gain'
    'Gini' index $G \equiv p(1-p)$
    $\rightarrow N_{\text{parent}} \times G_{\text{parent}} - N_{\text{left}} \times G_{\text{left}} - N_{\text{right}} \times G_{\text{right}}$
  - ▷ stop when a given criterion is reached
    number of events in node
    purity
    maximum depth
  - ▷ prune tree (cut back)





→ completely straightforward interpretation

→ problem: sensitive to fluctuations in training sample

Boosted decision trees (BDT)

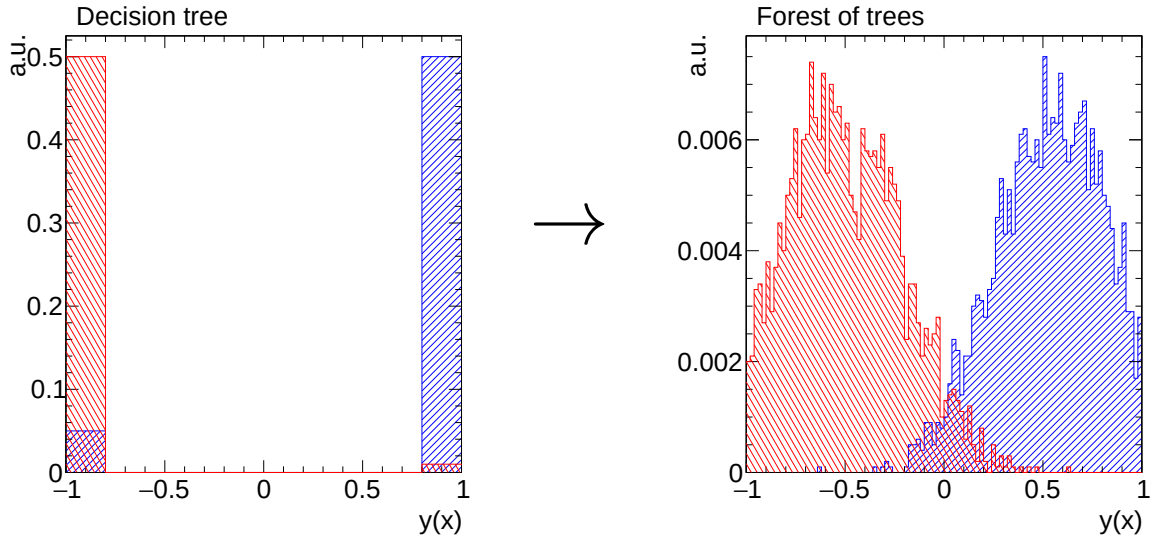
- Create ensemble of decision trees by reweighting events
  - misclassified events are given higher weight α in subsequent tree(s)

$$\alpha = \frac{1 - \text{err}}{\text{err}}$$

  - For ensemble of N trees calculate weighted sum for $y_{\text{BDT}}(\mathbf{x})$

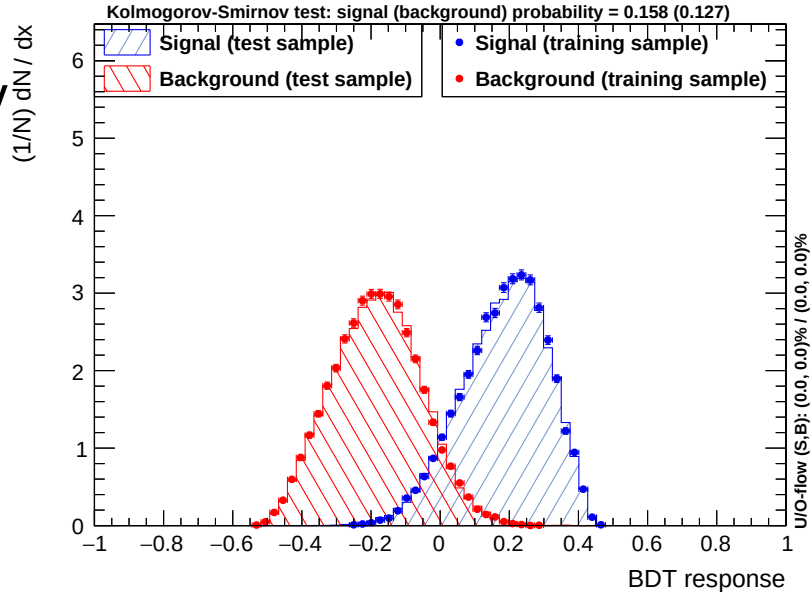
$$y_{\text{BDT}}(\mathbf{x}) = \frac{1}{N} \sum_{i}^{N} \ln(\alpha_i) y_i(\mathbf{x})$$

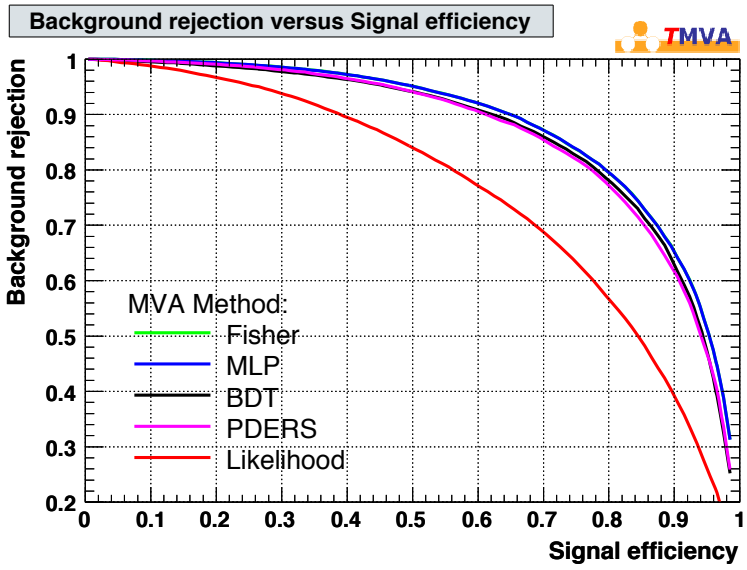
  - This is known as adaptive boosting (AdaBoost)
    events and decision tree output are reweighted
  - Other boosting methods available, e.g.
    gradient boosting
    bagging (resampling)



Technicalities II

- TMVA workflow
  - training: determine 'forest' structure
    XML file with explicit cuts and topology
  - testing: against overtraining*)
    independent sample(s)
    Kolmogorov-Smirnov prob.
  - evaluation: performance determination
    (ROC curve, by default)
  - validation:
    stability of training?
    do the cuts make sense?
    outliers anywhere?
  - application in your analysis
    reader (XML)
    function (C++ code fragment)





*) *"Overtraining occurs when a machine learning problem has too few degrees of freedom, because too many model parameters of an algorithm were adjusted to too few data points."*

Explicit TMVA example

- CMS $B_s^0 \to \mu^+\mu^-$ TMVA setup (w/o programmable variations):

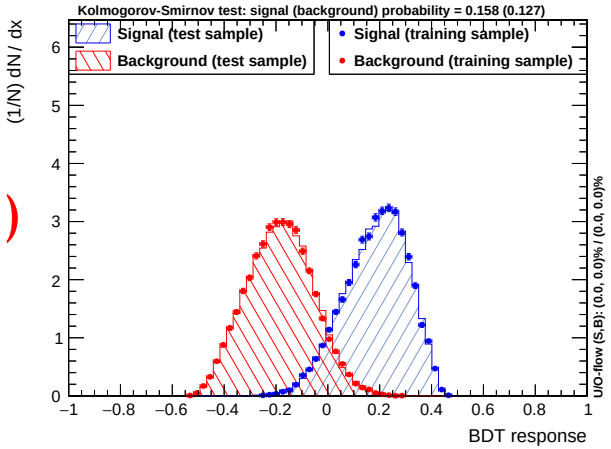
```
// --- load library and setup basics
TMVA::Tools::Instance();

TFile *oFile           = TFile::Open("aname.root", "RECREATE");
TMVA::Factory *factory = new TMVA::Factory("aname", oFile,
                         "V:!Silent:!Color:!DrawProgressBar:Transformations=I:AnalysisType=Classification");
TMVA::DataLoader *dataloader = new TMVA::DataLoader("dataset");

// --- setup variables and trees
dataloader->AddVariable("pt", 'F');
dataloader->AddVariable("iso", 'F');
// etc.
dataloader->AddSpectator("m",  "mass", "GeV", 'F' );

TTree *trainSg = (TTree*)inFile->Get("signalEvents0/events");
TTree *testSg  = (TTree*)inFile->Get("signalEvents1/events");
TTree *trainBg = (TTree*)inFile->Get("sidebandEvents0/events");
TTree *testBg  = (TTree*)inFile->Get("sidebandEvents1/events");

dataloader->AddSignalTree(trainSg,     signalWeight,      TMVA::Types::kTraining);
dataloader->AddSignalTree(testSg,      signalWeight,      TMVA::Types::kTesting);
dataloader->AddBackgroundTree(trainBg, cbackgroundWeight, TMVA::Types::kTraining);
dataloader->AddBackgroundTree(testBg,  tbackgroundWeight, TMVA::Types::kTesting);

dataloader->PrepareTrainingAndTestTree("", "nTrain_Signal=1.e4:nTest_Signal=1.e4:nTrain_Background=1.e4\\
                                            :nTest_Background=1.e4:SplitMode=Block:NormMode=None:V");

// --- Book, train, test, and evaluate methods
factory->BookMethod(dataloader, TMVA::Types::kBDT, "BDT", "!H:V:NTrees=800:BoostType=AdaBoost:AdaBoostBeta=0.5");
factory->TrainAllMethods();
factory->TestAllMethods();
factory->EvaluateAllMethods();
```

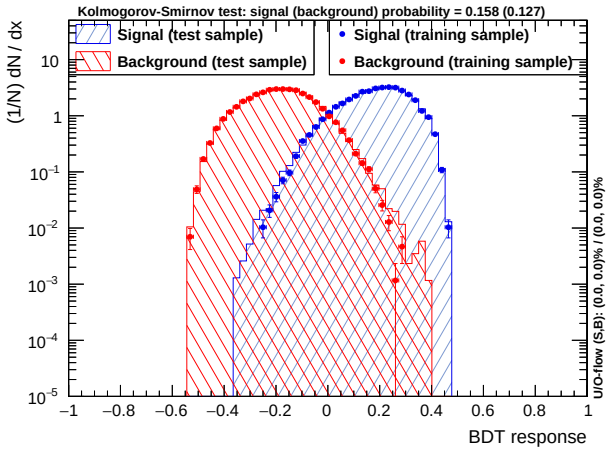
- timing? a few minutes

  3 BDTs with 800 trees/a dozen variables/event statistics $\mathcal{O}(10^4)$

Setup subtleties

- It is easy to setup an MVA application
  - ▷ but to get into the best possible world, care is required
- Input samples
  - ▷ statistics: imbalance between background and signal not useful
  - ▷ partitioning: different regions of detector/phasespace
  - ▷ preselection: outliers/statistics
- Variables
  - ▷ insensitive to useless variables (but: statistics!)
  - ▷ transformations
- 2 samples? 3 samples?!
  - → is a sideband interpolation biased?
  - ▷ training
  - ▷ testing
  - ▷ application
  - → especially for very delicate searches!
  - → TMVA does not help you, diy (e.g. evt%3==0[1,2])





Parameter choices

- Default TMVA parameters settings are quite good
  - scanning ranges will never hurt

Option	Array	Default	Predefined Values	Description
NTrees	–	800	–	Number of trees in the forest
MaxDepth	–	3	–	Max depth of the decision tree allowed
MinNodeSize	–	5%	–	Minimum percentage of training events required in a leaf node (default: Classification: 5%, Regression: 0.2%)
nCuts	–	20	–	Number of grid points in variable range used in finding optimal cut in node splitting
BoostType	–	AdaBoost	AdaBoost, RealAdaBoost, Bagging, AdaBoostR2, Grad	Boosting type for the trees in the forest (note: AdaCost is still experimental)
AdaBoostR2Loss	–	Quadratic	Linear, Quadratic, Exponential	Type of Loss function in AdaBoostR2
UseBaggedBoost	–	False	–	Use only a random (bagged) subsample of all events for growing the trees in each iteration.
Shrinkage	–	1	–	Learning rate for GradBoost algorithm
AdaBoostBeta	–	0.5	–	Learning rate for AdaBoost algorithm
UseRandomisedTrees	–	False	–	Determine at each node splitting the cut variable only as the best out of a random subset of variables (like in RandomForests)
UseNvars	–	2	–	Size of the subset of variables used with RandomisedTree option
UsePoissonNvars	–	True	–	Interpret UseNvars not as fixed number but as mean of a Possion distribution in each split with RandomisedTree option
BaggedSampleFraction	–	0.6	–	Relative size of bagged event sample to original size of the data sample (used whenever bagging is used (i.e. UseBaggedBoost, Bagging,)
UseYesNoLeaf	–	True	–	Use Sig or Bkg categories, or the purity=S/(S+B) as classification of the leaf node -> Real-AdaBoost

Option Table 22: 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 60 can also be configured. The table is continued in Option Table 24.

Option	Array	Default	Predefined Values	Description
NegWeightTreatment	–	InverseBoostNegWeights	IgnoreNegWeightsInTraining, PairNegWeightsGlobal, Pray	How to treat events with negative weights in the BDT training (particular the boosting) : IgnoreInTraining; Boost With inverse boostweight; Pair events with negative and positive weights in traning sample and *annihilate* them (experimental!)
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
DoBoostMonitor	–	False	–	Create control plot with ROC integral vs tree number
UseFisherCuts	–	False	–	Use multivariate splits using the Fisher criterion
MinLinCorrForFisher	–	0.8	–	The minimum linear correlation between two variables demanded for use in Fisher criterion in node splitting
UseExclusiveVars	–	False	–	Variables already used in fisher criterion are not anymore analysed individually for node splitting
DoPreselection	–	False	–	and and apply automatic pre-selection for 100% efficient signal (bkg) cuts prior to training
RenormByClass	–	False	–	Individually re-normalize each event class to the original size after boosting
SigToBkgFraction	–	1	–	Sig to Bkg ratio used in Training (similar to NodePurityLimit, which cannot be used in real adaboost

Option Table 23: Continuation of Option Table 22.

Optimization workflow

- Prerequisites
  - ▷ code setup to allow changing 'everything' from the command line
  - ▷ preselection of training/testing trees/files
  - ▷ variables
  - ▷ ranges and steps of BDT parameters to optimize
- Operationally
  - ▷ run on cluster with worker nodes
  - → distribute training/testing files to local /scratch disks
  - ▷ abort individual trainings asap
  - → define minimum criteria and calculate required ingredients
    - KS-probability
    - sensitivity
    - similarity of $y(\mathbf{x})$ distribution

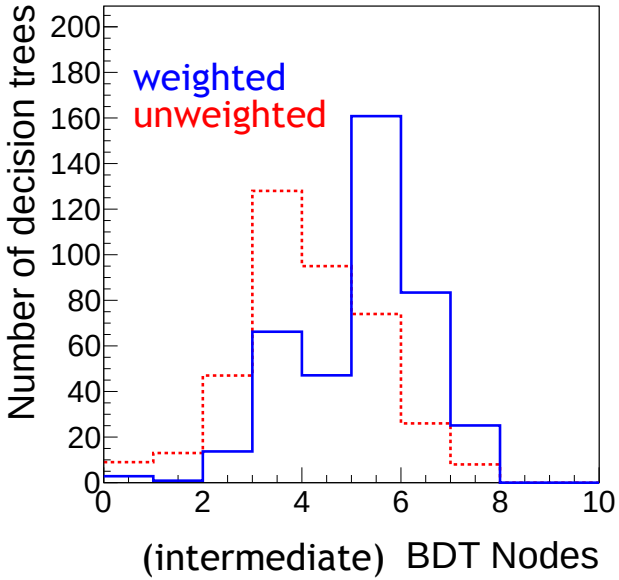
⇒ $\mathcal{O}(10^4)$ jobs

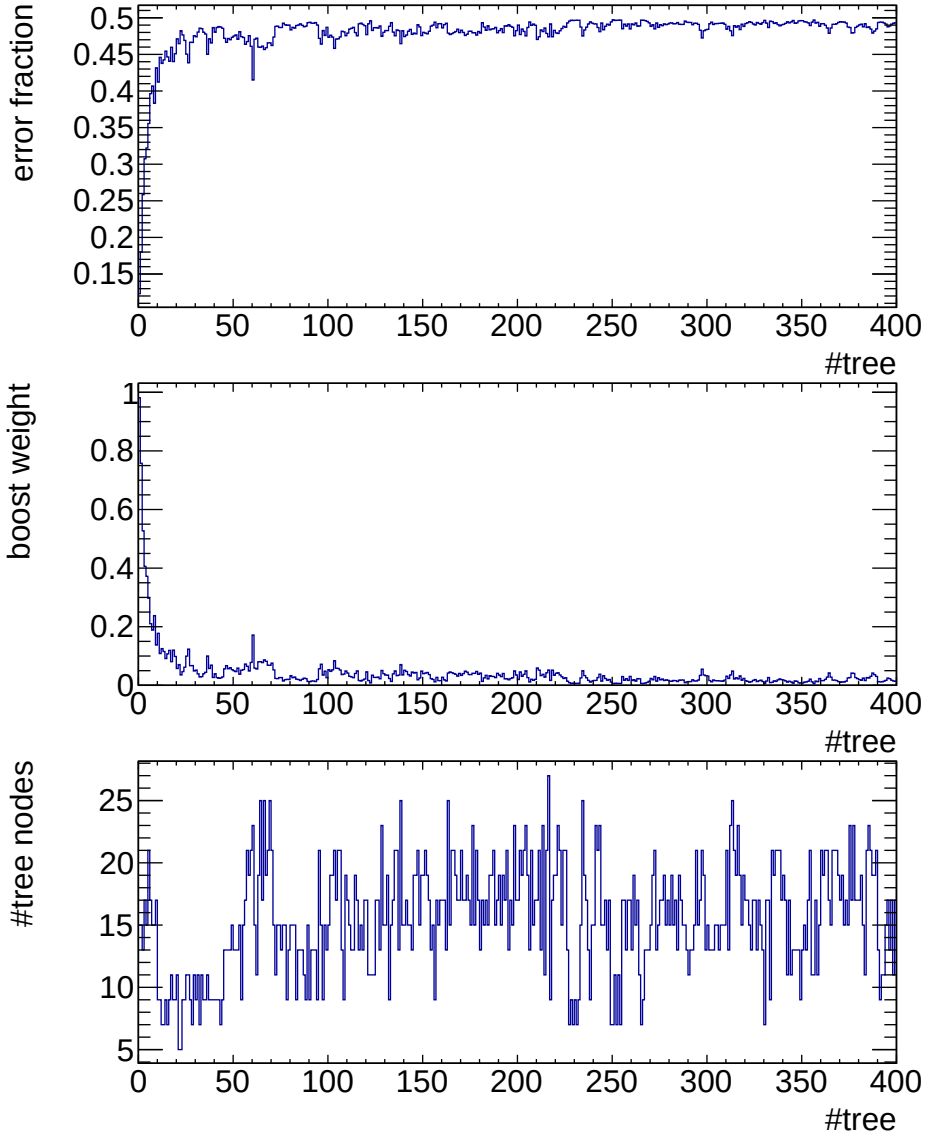
sometimes several iterations

(but this also applies to (random) grid searches, etc)

BDT technical evaluation

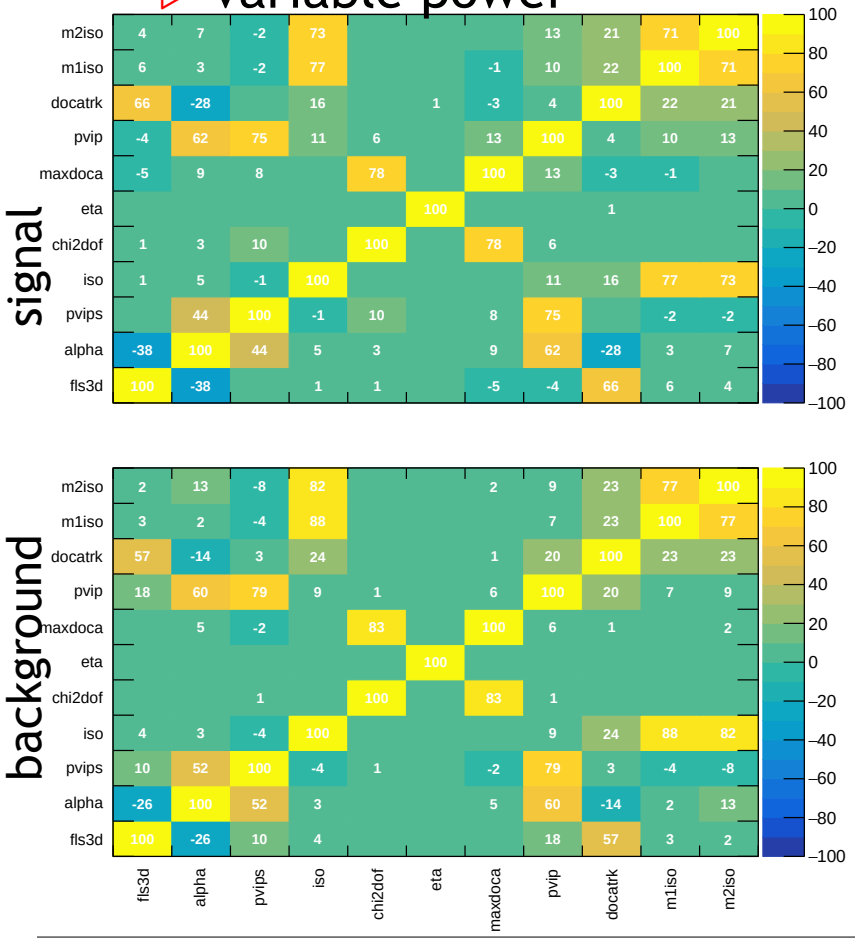
- error fraction
  - unboosted (original) events
  - for 'difficult' events $\rightarrow$ 0.5
- boost weight
  - remember its definition?
- # tree nodes
  - w/ or w/o leaf nodes
  - depends on depth of BDT

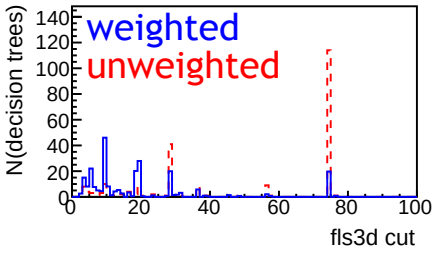


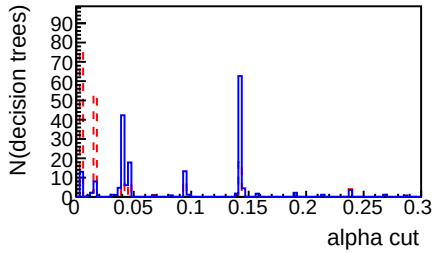


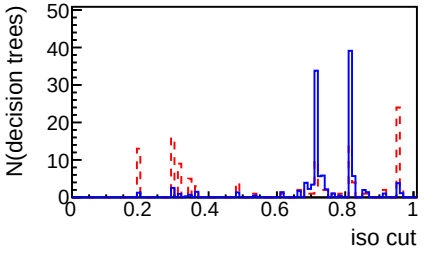
BDT validation of variables

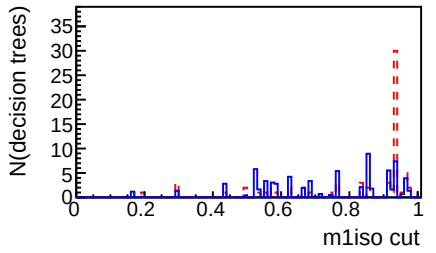
- Check (from XML file)
  - distribution of cuts
  - usage of variables in trees
  - variable correlations
  - variable power

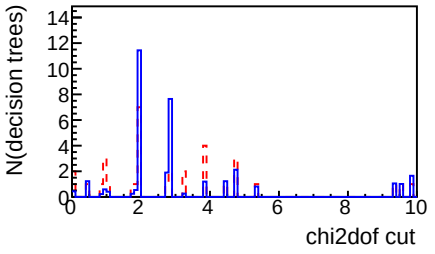


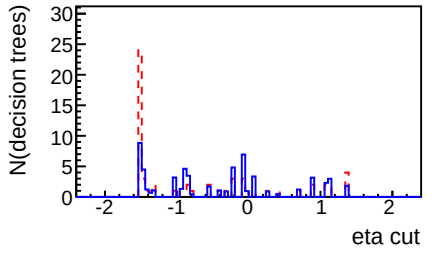


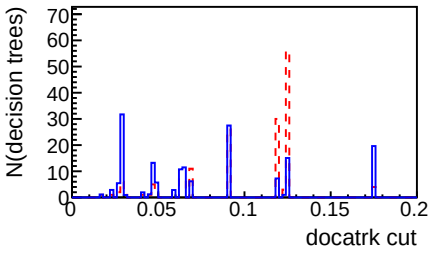


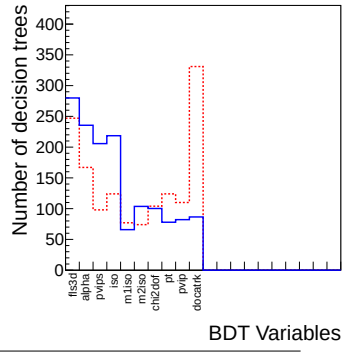








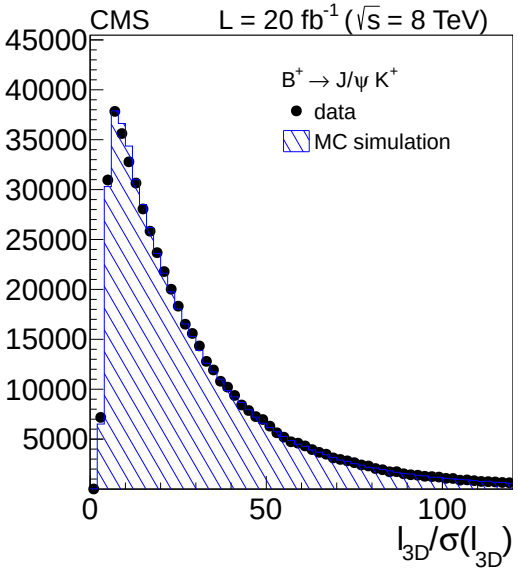


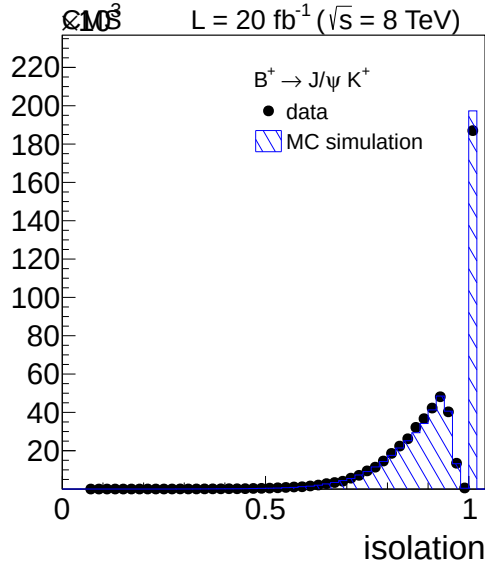


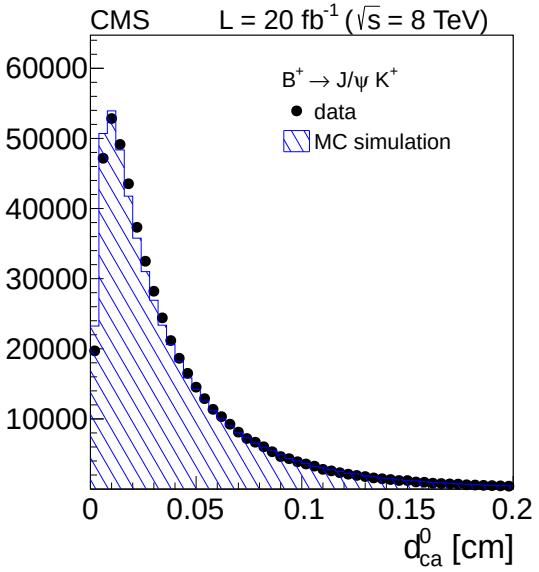
Data/MC validation

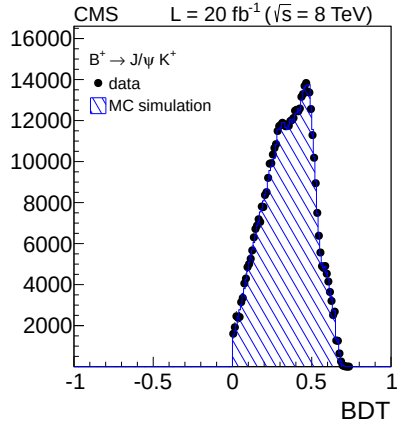
- MC simulation description of data
  - ▷ check variable distributions for specific $\mathcal{C}$
  - ▷ check $y(\mathbf{x})$ for various preselections
- Samples to compare?
  - ▷ background MC with background data
  - ▷ 'control' samples
    - identical topology as signal not mandatory

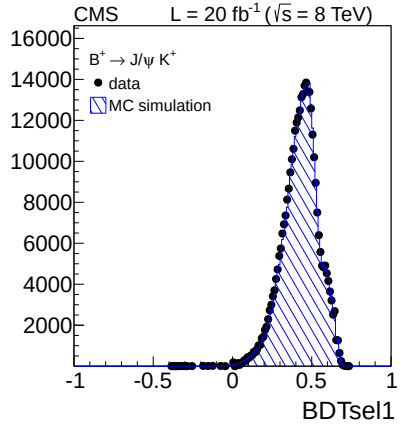
  → calculate variables correspondingly

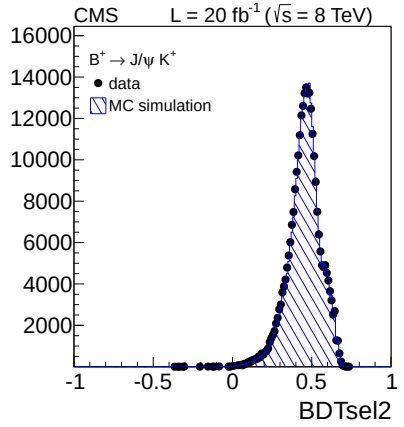






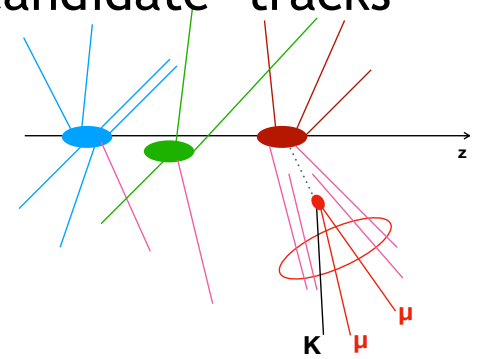




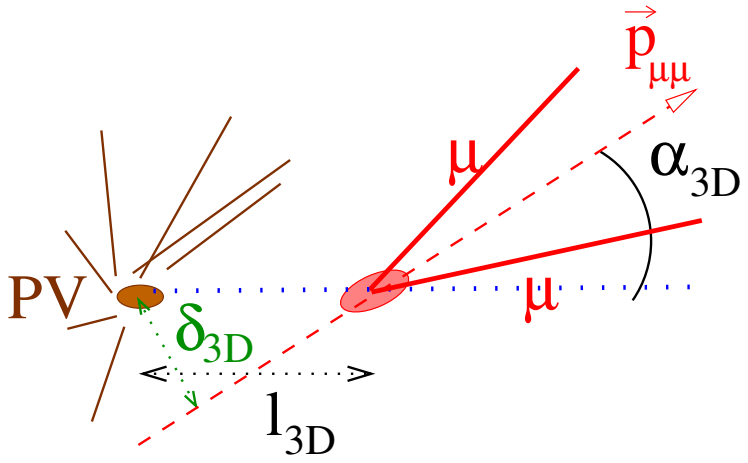


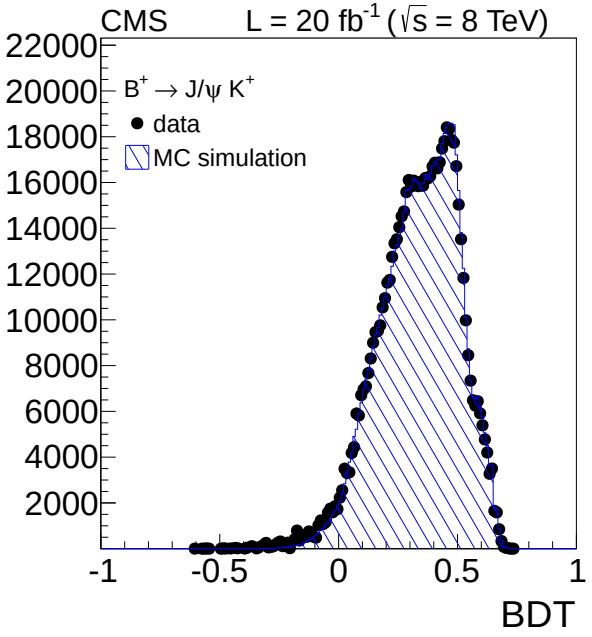
‘Control’ samples

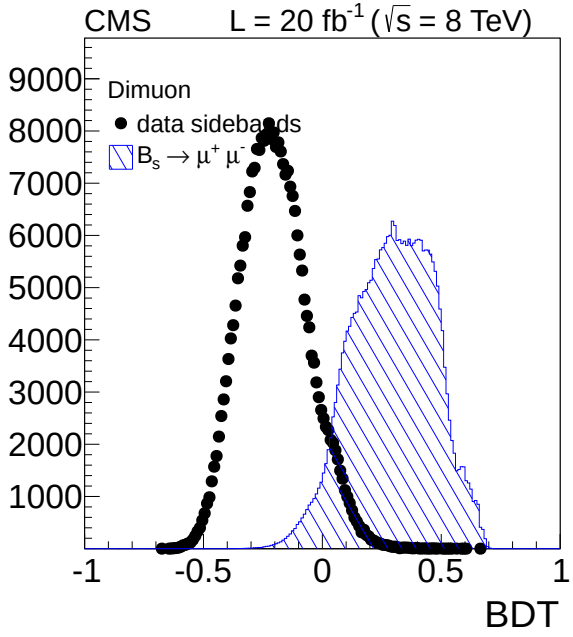
- ‘Candidate’ variables are ‘trivial’
  - dimuon $B_s^0 \to \mu^+\mu^-$
  - $B^+ \to J/\psi\, K^+ \to \mu^+\mu^- K^+$
- degrees of freedom
  - restrict to subset (dimuon vtx)
  - renormalize (χ^2/dof)
- ‘environment’ variables:
  - do not include ‘candidate’ tracks



- $y(\mathbf{x})$ will look different:
  - different topology
  - does not matter!

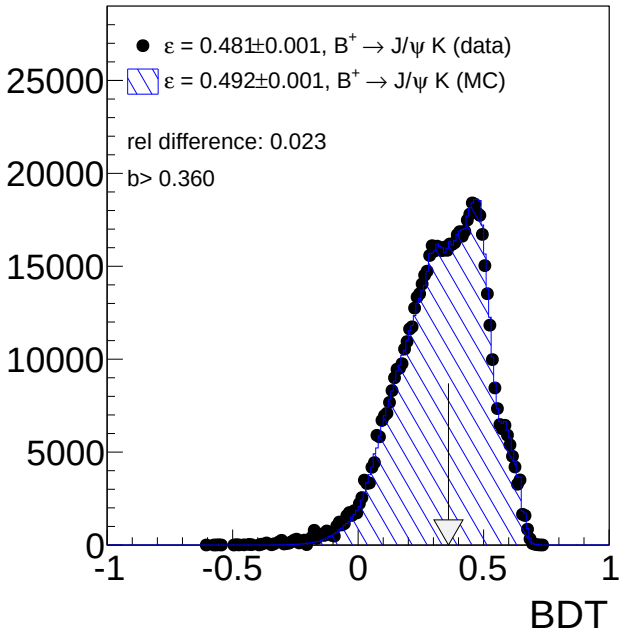


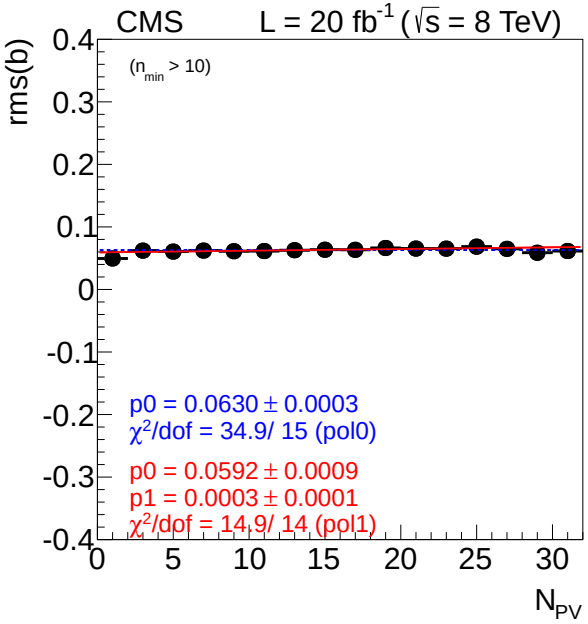


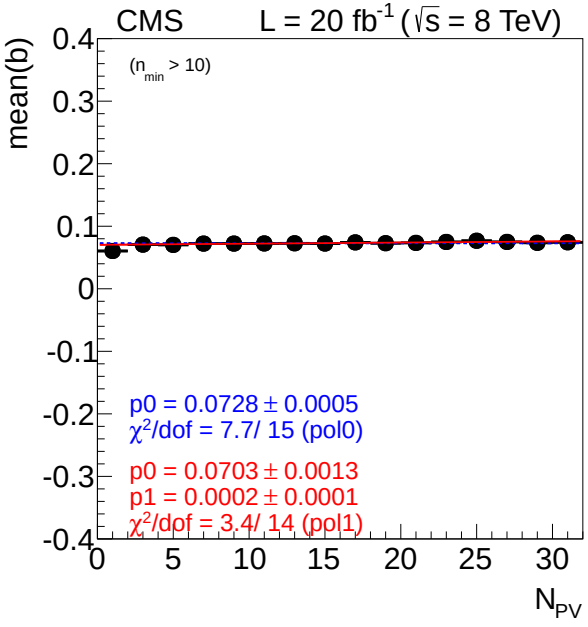


Systematics

- One number/event allows systematic studies
  - dependence of $y(\mathbf{x})$ distribution vs. key variables
- Generate signal MC with different
  - mass, lifetime, . . .
- quantification of systematics?
  - reweighing for data/MC differences
  - e.g. different 'cut' efficiency in data/MC (control samples)







Outlook

- There is no magic involved in supervised machine learning
  - it will not ruin your analysis
  - it will not save you
  - it will boost your sensitivity by $\mathcal{O}(20\%)$, compared to 'good' cuts
- Disclaimers
  - garbage in, garbage out
  - you control the training and validation
  - performance differences between ANN/BDT/. . . normally smaller than other factors
- Many ways to test/optimize/validate any MVA setup
  - absolutely required
- More information, e.g.
  - tmva.sourceforge.net
  - http://tmva.sourceforge.net/talks.shtml and references in there