Multivariate Data Analysis Techniques

Andreas Hoecker (CERN)

Workshop on Statistics, Karlsruhe, Germany, Oct 12–14, 2009

Outline

- Introduction to multivariate classification and regression
- Multivariate methods and data preprocessing
- Examples

Will not cover likelihood fits à la RooFit, would deserve its own talk !

Some (incomplete!) 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,
many, many other packages!

"All inclusive" 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

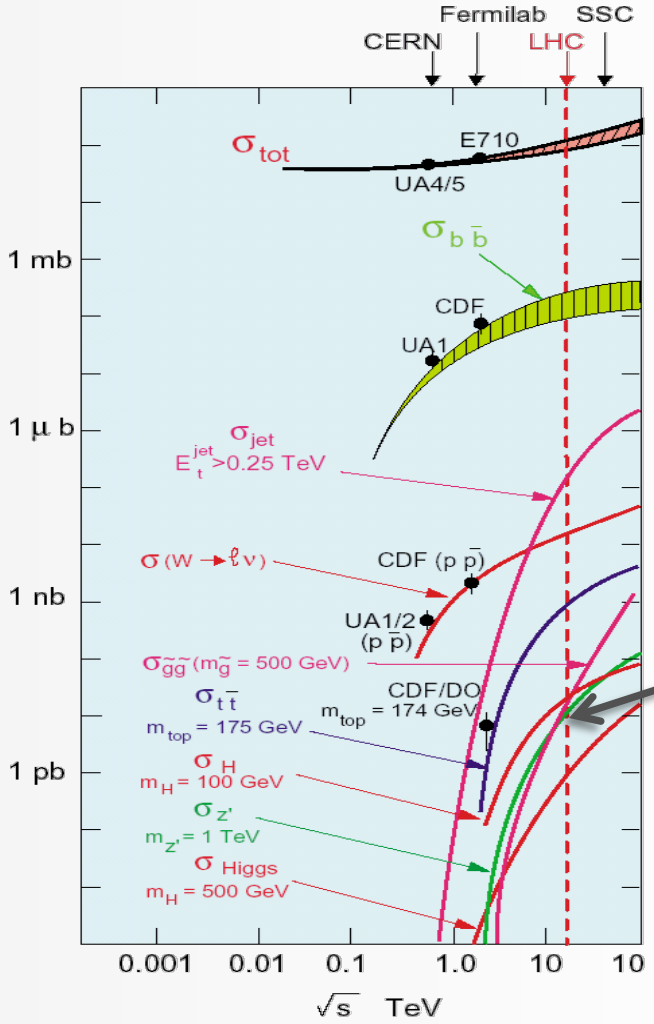
WEKA: http://www.cs.waikato.ac.nz/ml/weka/

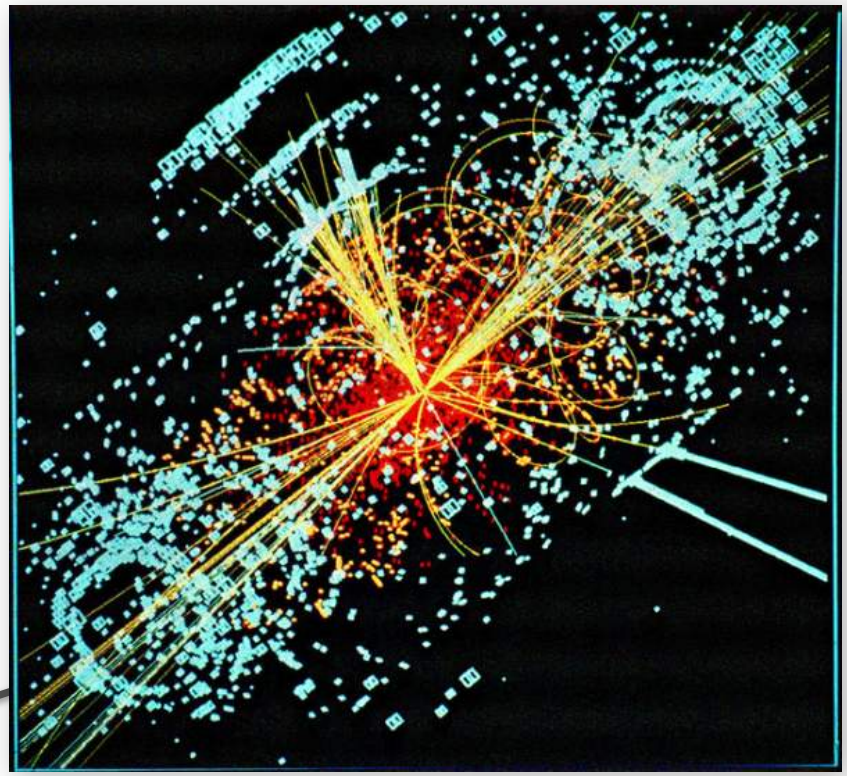
Huge data analysis library available in "R": http://www.r-project.org/

Conferences: PHYSTAT, ACAT,…

Simulated Higgs Event in CMS

That's how a "typical" higgs event looks like (underlying ~25 minimum bias events)





And not only this: such events occur only in a tiny fraction of the proton-proton collisions $O(10^{-11})$

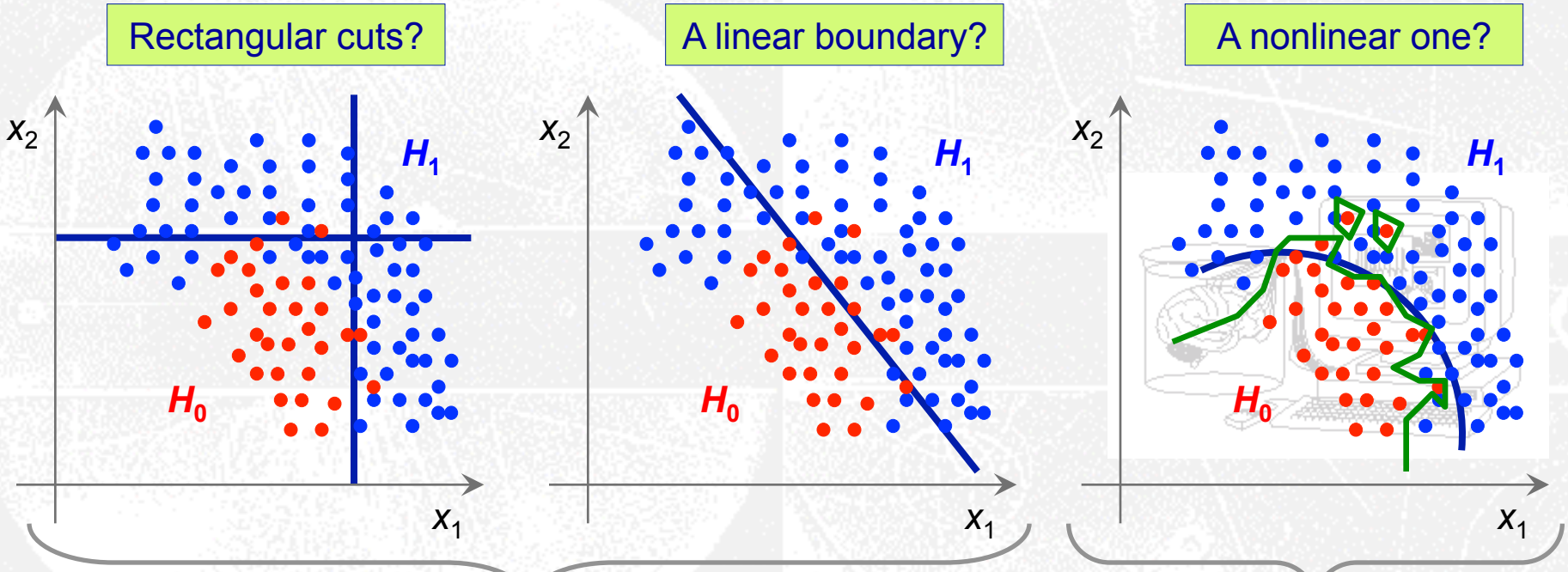
Event Classification in High-Energy Physics (HEP)

- Most HEP analyses require discrimination of signal from background:
  - Event level (Higgs searches, …)
  - Cone level (Tau-vs-jet reconstruction, …)
  - Track level (particle identification, …)
  - Lifetime and flavour tagging (*b*-tagging, …)
  - Parameter estimation (*CP* violation in *B* system, …)
  - etc.
- The multivariate input information used for this has various sources
  - 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, …)
  - etc.
- Traditionally few powerful input variables were combined; new methods allow to use up to 100 and more variables w/o loss of classification power

e.g. MiniBooNE: NIMA 543 (2005), or D0 single top: Phys.Rev. D78, 012005 (2008)

Event Classification

- Suppose data sample with two types of events: H_0, H_1
  - We have found discriminating input variables x_1, x_2, …
  - What decision boundary should we use to select events of type H_1 ?

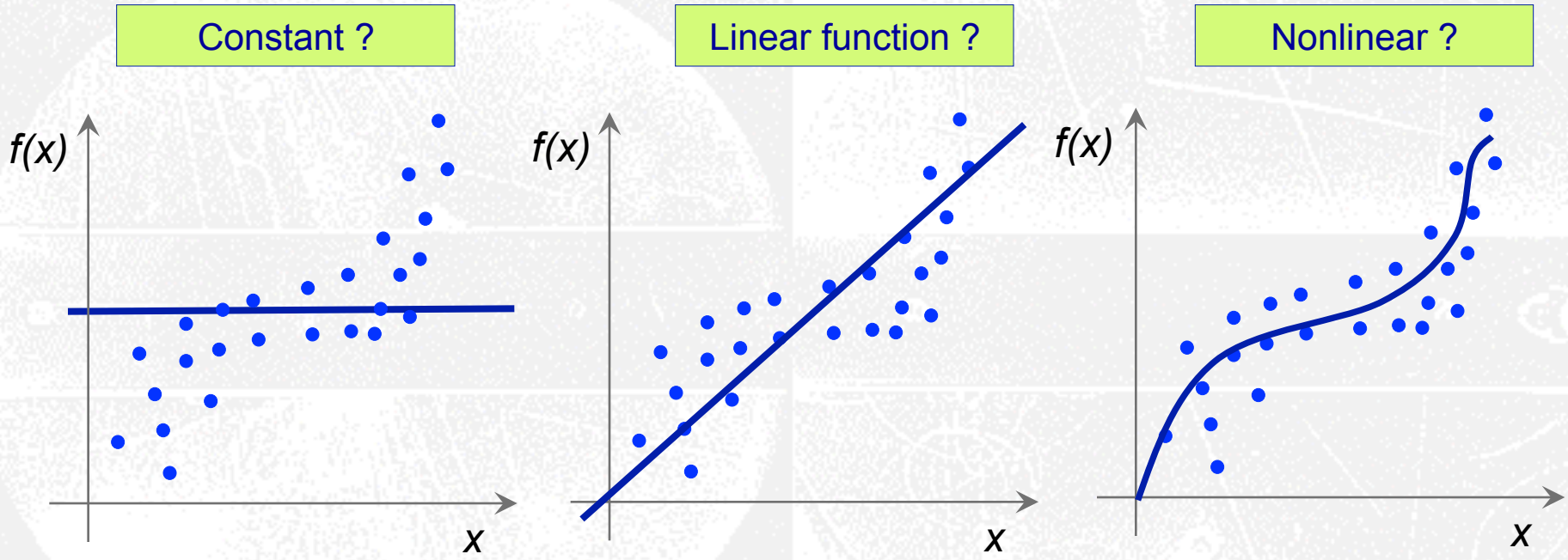


- How can we decide this in an optimal way ? → Let the machine learn it !

Low variance (stable), high bias methods — High variance, small bias methods

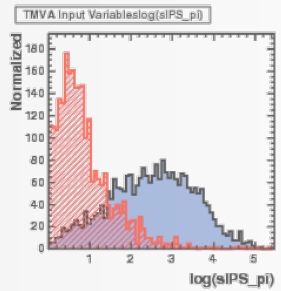
Regression

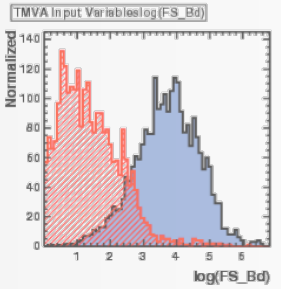
- How to estimate a “functional behaviour” from a set of measurements?
  - Energy deposit in a the calorimeter, distance between overlapping photons, …
  - Entry location of the particle in the calorimeter or on a silicon pad, …

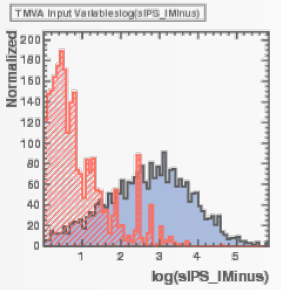


- Seems trivial? What if we have many input variables?

Multivariate Event Classification







$\mathbb{R}^D$

"feature space"

Each event, if Signal or Background, has "D" measured variables.

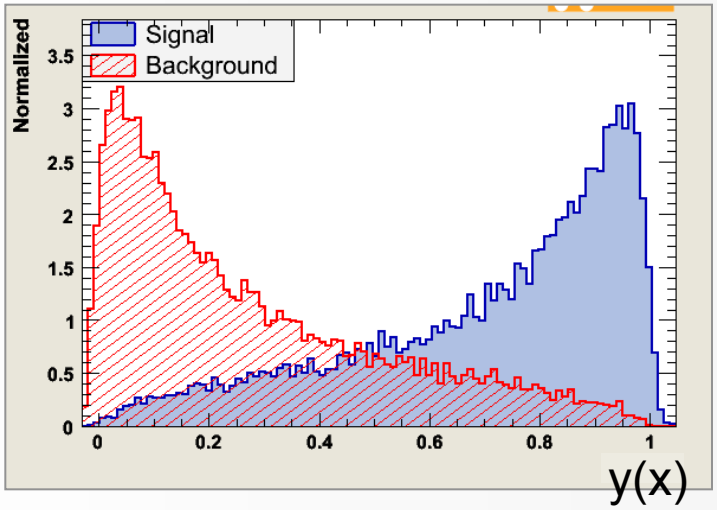
Find a mapping from D-dimensional input-observable = "feature" space to one dimensional output $y(x): \mathbb{R}^n \to \mathbb{R}$:

Most general form

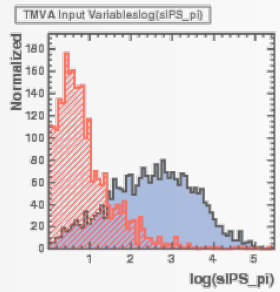
$y = y(\mathbf{x})$; $\mathbf{x}$ in $\mathbb{R}^D$

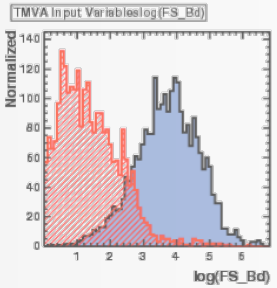
$\mathbf{x} = \{x_1, ..., x_D\}$: input variables

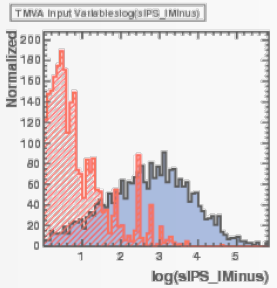
Plotting the resulting y(x) values:



Multivariate Event Classification



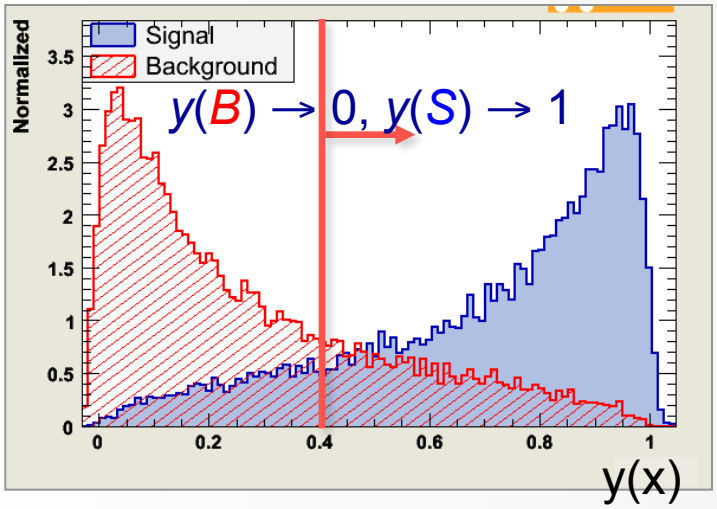




$\mathbb{R}^D$

"feature space"

Each event, if Signal or Background, has "D" measured variables.



y(x): $R^n \rightarrow R$: $\longrightarrow \mathbb{R}$

y(x): "test statistic" in D-dimensional space of input variables

Distributions of y(x): $PDF_S(y)$ and $PDF_B(y)$

Used to set the selection cut!

→ Efficiency and purity

y(x): > cut: signal; = cut: decision boundary; < cut: background

y(x) = const: surface defining the decision boundary.

Overlap of $PDF_S(y)$ and $PDF_B(y)$ affects separation power, purity

Event Classification

P(Class=C|**x**) (or simply P(C|x)) : probability that the event class is of type C, given the measured observables **x** = $\{x_1,\ldots,x_D\}$ → y(**x**)

Probability density distribution according to the measurements **x** and the given mapping function

Prior probability to observe an event of "class C", *i.e.,* the relative abundance of "signal" versus "background"

$$P(\text{Class} = C \mid y) = \frac{P(y \mid C) \cdot P(C)}{P(y)}$$

Posterior probability

Overall probability density to observe the actual measurement y(**x**), *i.e.,* $P(y) = \sum_{\text{Classes}} P(y \mid \text{Class}) P(\text{Class})$

Bayes Optimal Classification

$$P(\text{Class} = C \mid y) = \frac{P(y \mid C)P(C)}{P(y)}$$

$\mathbf{x}=\{x_1,\ldots,x_D\}$: measured observables
$y = y(\mathbf{x})$

AND

Minimum error in misclassification if C chosen such that it has maximum $P(C|\mathbf{y})$

→ to select S(ignal) over B(ackground), place decision on:

[Or any monotonic function of P(S|y) / P(B|y)]

$$\frac{P(S \mid y)}{P(B \mid y)} = \frac{P(y \mid S)}{P(y \mid B)} \cdot \frac{P(S)}{P(B)} > c$$

"c" determines efficiency and purity

Posterior odds ratio

Likelihood ratio as discriminating function y(x)

Prior odds ratio of choosing a signal event (relative probability of signal vs. bkg)

Any Decision Involves a Risk

Decide to treat an event as "Signal" or "Background"

Type-1 error:

classify event as Class C even though it is not
(accept a hypothesis although it is not true)
(reject the null-hypothesis although it would have been the correct one)
→ loss of purity (in the selection of signal events)

Type-2 error:

fail to identify an event from Class C as such
(reject a hypothesis although it would have been true)
(fail to reject the null-hypothesis/accept null hypothesis although it is false)
→ loss of efficiency (in selecting signal events)

Trying to select signal events:
(i.e. try to disprove the null-hypothesis stating it were "only" a background event)

accept as: / truly is:	Signal	Back-ground
Signal	☺	Type-2 error
Back-ground	Type-1 error	☺

"A": region of the outcome of the test where you accept the event as signal:

Significance α: Type-1 error rate:
(=p-value): α = background selection "efficiency"

$$\alpha = \int_{A} P(x|B)dx$$

should be small

Size β: Type-2 error rate:
Power: 1− β = signal selection efficiency

$$\beta = \int_{!A} P(x|S)dx$$

should be small

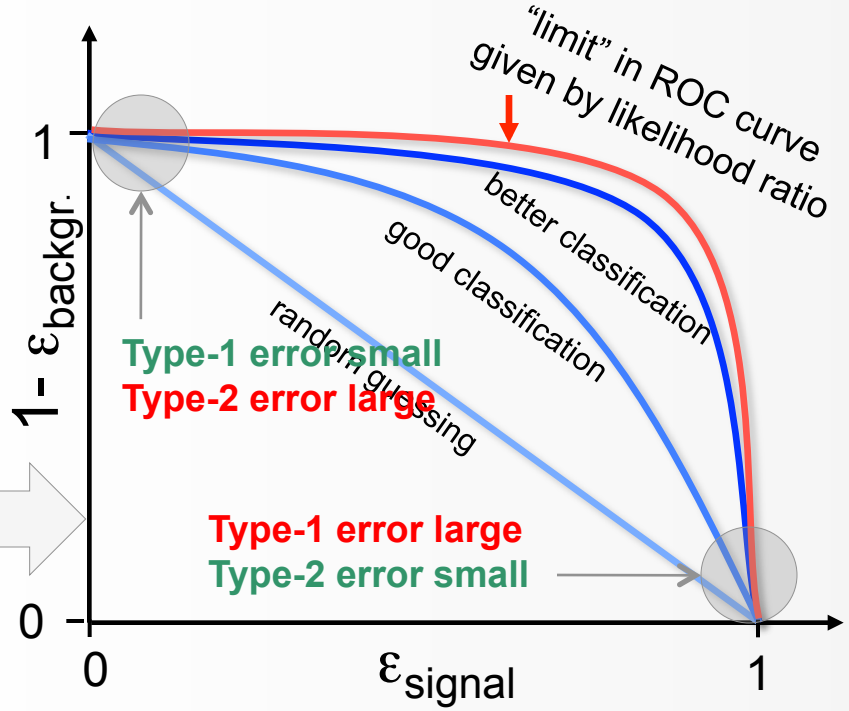
Neyman-Pearson Lemma

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

<u>Neyman-Peason</u>:

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

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



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

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

- Measurement of signal cross section: maximum of $S/\sqrt{(S+B)}$ or equiv. $\sqrt{(\varepsilon \cdot p)}$
- Discovery of a signal : maximum of $S/\sqrt{(B)}$
- Precision measurement: high purity (p)
- Trigger selection: high efficiency (ε)

Realistic Event Classification

Unfortunately, the true probability densities functions are typically unknown:
→ Neyman-Pearson's lemma doesn't really help us…

Use MC simulation, or more generally: set of known (already classified) "events"

Use

Of course, there is **no magic** in here. We still need to:

- Choose the discriminating variables
- Choose the class of models (linear, non-linear, flexible or less flexible)
- Tune the "learning parameters" → bias vs. variance trade off
- Check generalisation properties
- Consider trade off between statistical and systematic uncertainties

→ e.g. Linear Discriminator, Neural Networks, …

→ supervised (machine) learning

* hyperplane in the strict sense goes through the origin. Here is meant an "affine set" to be precise.

Multivariate Analysis Methods

- Examples for classifiers and regression methods
  - Rectangular cut optimisation
  - Projective and multidimensional likelihood estimator
  - k-Nearest Neighbor algorithm
  - Fisher and H-Matrix discriminants
  - Function discriminants
  - Artificial neural networks
  - Boosted decision trees
  - RuleFit
  - Support Vector Machine

Examples in this lecture based on *T*MVA: http://tmva.sf.net

- Examples for preprocessing methods:
  - Decorrelation, Principal Value Decomposition, Gaussianisation
- Examples for combination methods:
  - Boosting, Categorisation, MVA Committees

Data Preprocessing

Data Preprocessing: Decorrelation

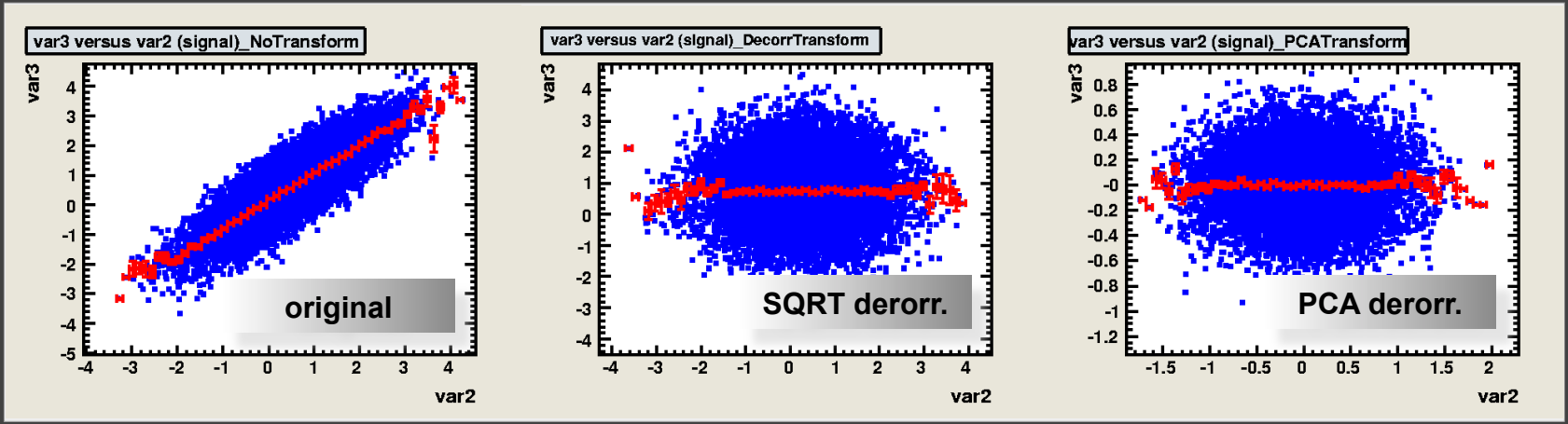
- Commonly realised for all methods in *T*MVA
- Removal of linear correlations by rotating input variables
  - Cholesky decomposition: determine *square-root* C' of covariance matrix C, *i.e.*, $C = C'C'$
  - Transform original (x) into decorrelated variable space (x') by: $x' = C'^{-1}x$
- Principal component analysis
  - Variable hierarchy: linear transformation projecting on axis to achieve largest variance

$$x_k^{\mathrm{PC}}(i_{\mathrm{event}}) = \sum_{v\in\{\mathrm{variables}\}} \left[x_v(i_{\mathrm{event}}) - \bar{x}_v\right] \cdot v_v^{(k)} \quad , \forall k \in \{\mathrm{variables}\}$$

PC of variable k | Sample means | Eigenvector

  - Matrix of eigenvectors V obeys relation: $C \cdot V = D \cdot V$ thus PCA eliminates correlations

Data Preprocessing: Decorrelation



Note that decorrelation is only complete, if

- Correlations are linear
- Input variables are Gaussian distributed
- Not very accurate conjecture in general

“Gaussian-isation”

- Improve decorrelation by pre-”Gaussianisation” of variables
  - First: “Rarity” transformation to achieve uniform distribution:

$$x_k^{\text{flat}}(i_{\text{event}}) = \int_{-\infty}^{x_k(i_{\text{event}})} p_k(x'_k)\,dx'_k \quad , \forall k \in \{\text{variables}\}$$

Rarity transform of variable *k* | Measured value | PDF of variable *k*

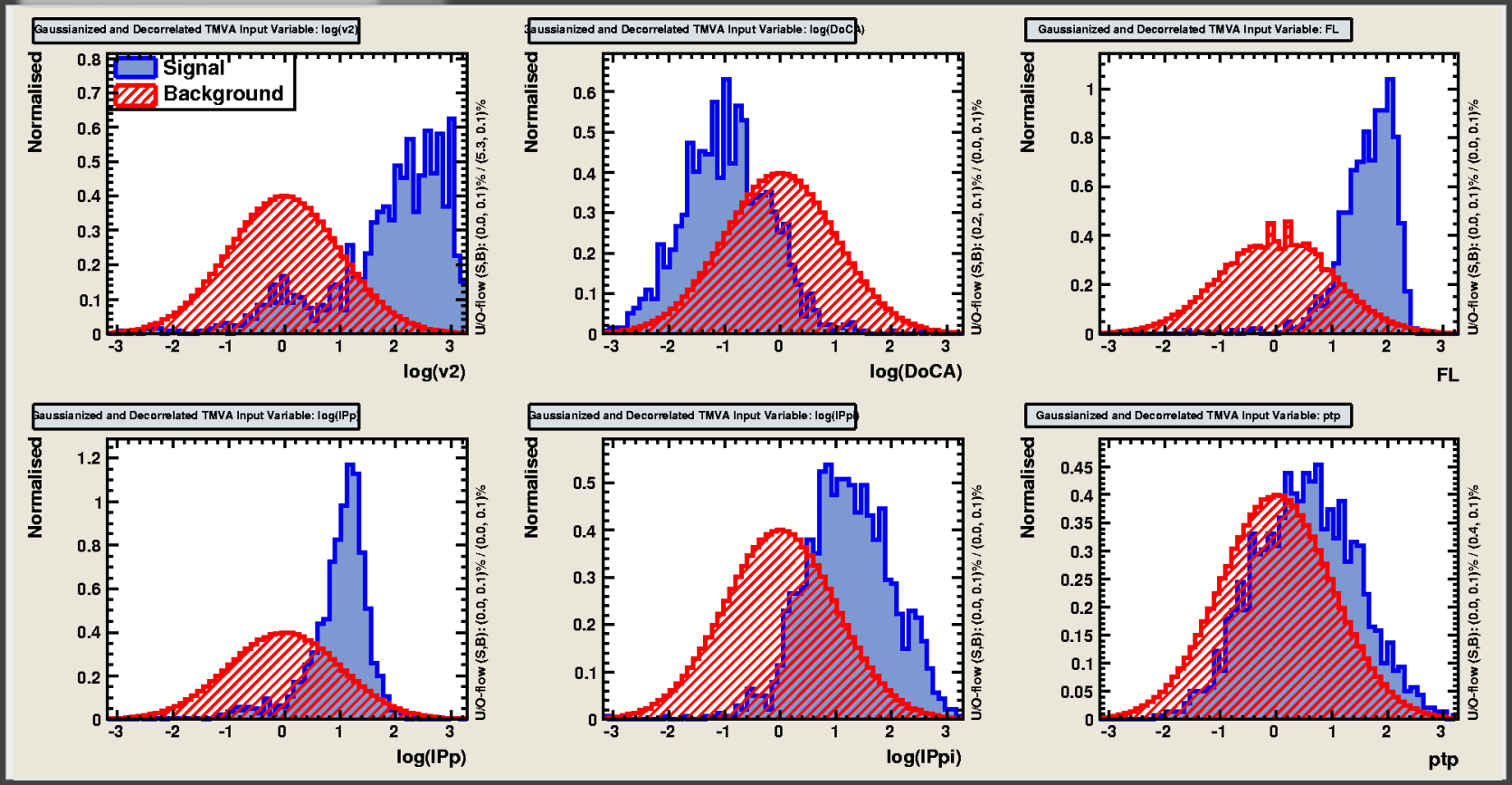
The integral can be solved in an unbinned way by event counting, or by creating non-parametric PDFs (see later for likelihood section)

  - Second: make Gaussian via inverse error function: $\text{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt$

$$x_k^{\text{Gauss}}(i_{\text{event}}) = \sqrt{2} \cdot \text{erf}^{-1}\left(2x_k^{\text{flat}}(i_{\text{event}}) - 1\right) \quad , \forall k \in \{\text{variables}\}$$

“Gaussian-isation”

Background - Gaussianised



We cannot simultaneously “gaussianise” both signal and background !

How to apply the Preprocessing Transformation ?

- Any type of preprocessing will be **different** for signal and background
- But: for a given test event, we do not know the species !
  - **Not so good solution**: choose one or the other, or a S/B mixture.
    As a result, none of the transformations will be perfect
  - **Good solution**: for some methods it is possible to test both S and B hypotheses with *their* transformations, and to compare them. Example, projective likelihood ratio:

$$y_L(i_{\text{event}}) = \frac{\prod_{k\in\{\text{variables}\}} p_k^S\left(x_k(i_{\text{event}})\right)}{\prod_{k\in\{\text{variables}\}} p_k^S\left(x_k(i_{\text{event}})\right) + \prod_{k\in\{\text{variables}\}} p_k^B\left(x_k(i_{\text{event}})\right)}$$

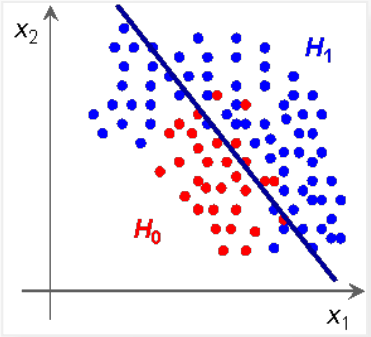
$$y_L^{\text{trans}}(i_{\text{event}}) = \frac{\prod_{k\in\{\text{variables}\}} p_k^S\left(\hat{T}^S x_k(i_{\text{event}})\right)}{\prod_{k\in\{\text{variables}\}} p_k^S\left(\hat{T}^S x_k(i_{\text{event}})\right) + \prod_{k\in\{\text{variables}\}} p_k^B\left(\hat{T}^B x_k(i_{\text{event}})\right)}$$

Classifiers

Rectangular Cut Optimisation

- Simplest method: cut in rectangular variable volume

$$x_{\text{cut}}\left(i_{\text{event}}\right)\in\{0,1\} \;=\; \bigcap_{v\in\{\text{variables}\}} \left(x_v\left(i_{\text{event}}\right)\subset\left[x_{v,\min}, x_{v,\max}\right]\right)$$



- Cuts usually benefit from prior decorrelation of cut variables

- Technical challenge: how to find optimal cuts ?
  - MINUIT fails due to non-unique solution space
  - *T*MVA uses: **Monte Carlo sampling**, **Genetic Algorithm**, **Simulated Annealing**
  - Huge speed improvement of volume search by sorting events in **binary tree**

digression

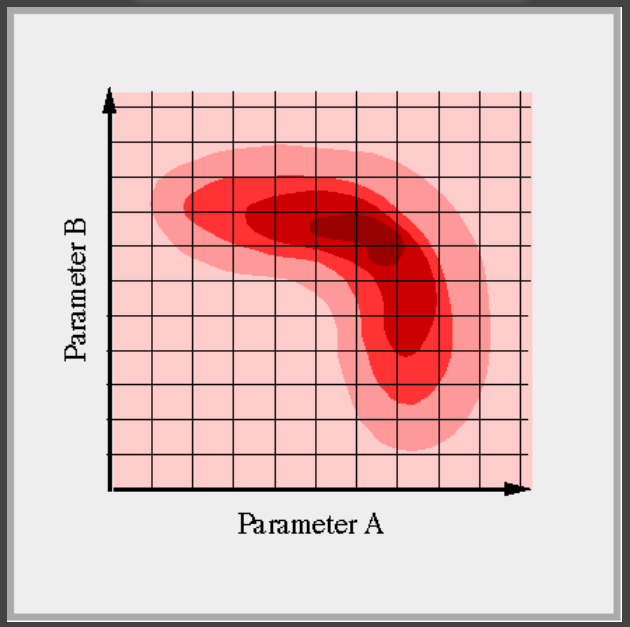
- Minimisation techniques in *T*MVA
- Binary tree sorting

Minimisation

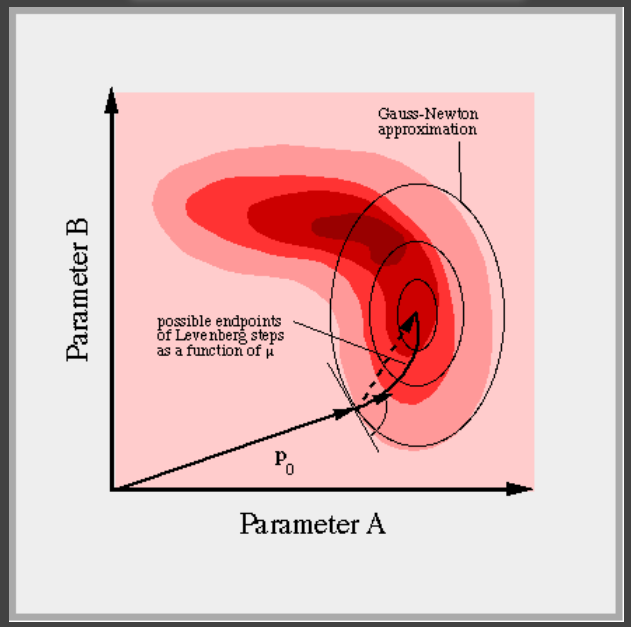
- Robust **global** minimum finder needed at various places in TMVA
- Brute force method: Monte Carlo Sampling
  - Sample entire solution space, and chose solution providing minimum estimator
  - Good global minimum finder, but poor accuracy
- Default solution in HEP: (T)Minuit/Migrad
  - Gradient-driven search, using variable metric, can use quadratic Newton-type solution
  - Poor global minimum finder, gets quickly stuck in presence of local minima
- Specific **global** optimisers implemented in TMVA:
  - **Genetic Algorithm**: biology-inspired optimisation algorithm
  - **Simulated Annealing**: slow “cooling” of system to avoid “freezing” in local solution
- TMVA allows to chain minimisers
  - For example, one can use MC sampling to detect the vicinity of a global minimum, and then use Minuit to accurately converge to it.

Minimisation Techniques

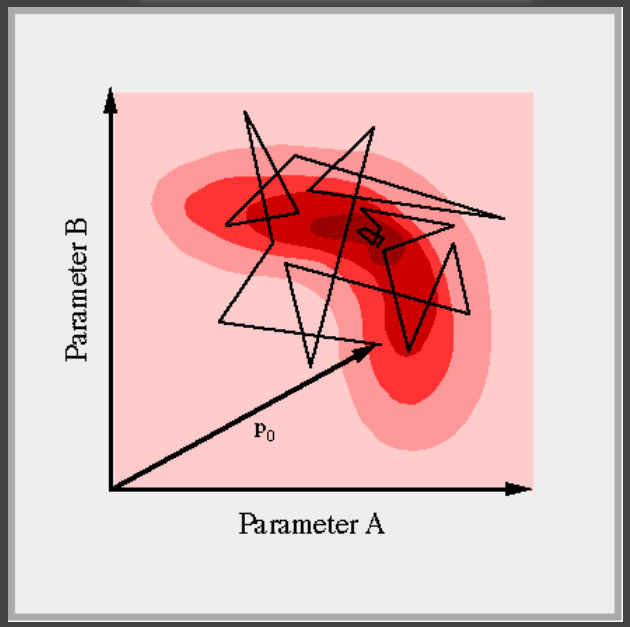
Grid search



Quadratic Newton



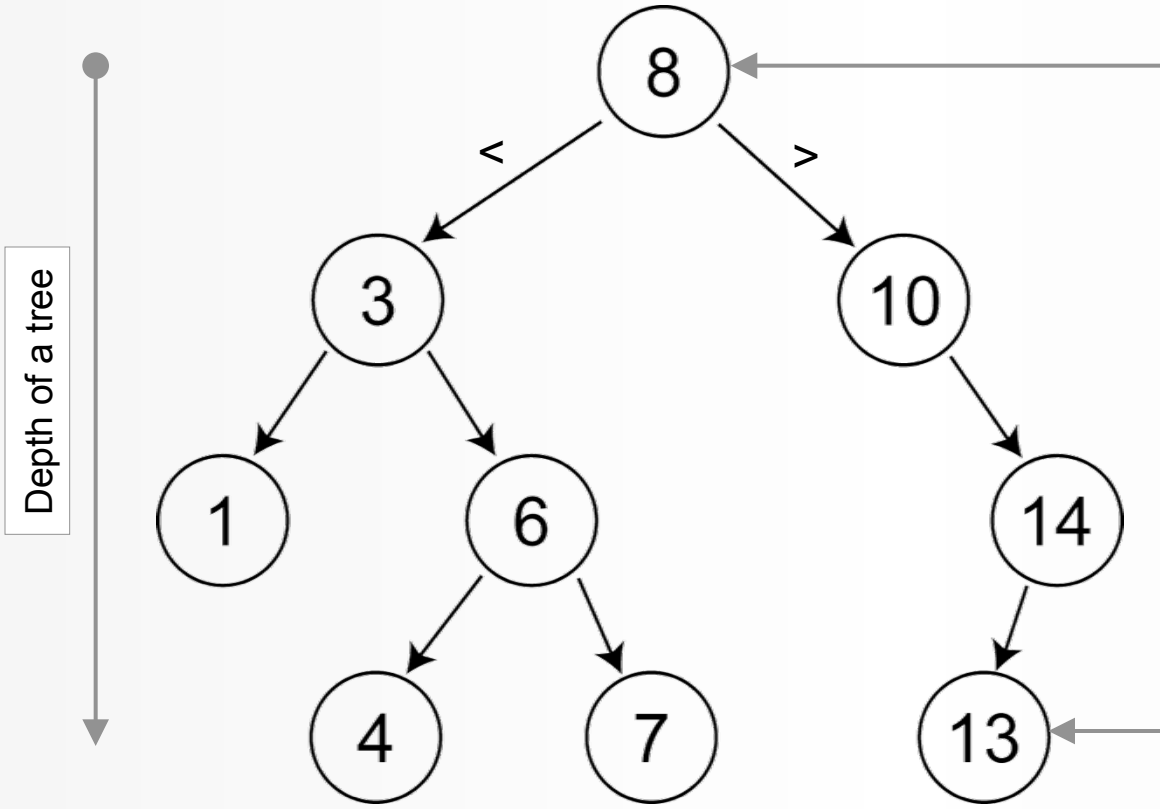
Simulated Annealing



Source: http://www-esd.lbl.gov/iTOUGH2/Minimization/minalg.html

Binary Trees

- Tree data structure in which each node has at most two children
  - Typically the child nodes are called left and right
  - Binary trees are used in TMVA to implement **binary search trees** and **decision trees**



Amount of computing time to search for events:

- Box search: ~ $(N_{events})^{Nvar}$
- BT search: ~ $N_{events} \cdot N_{var} \ln_2(N_{events})$

Projective Likelihood Estimator (PDE Approach*)

- Much liked in HEP: probability density estimators for each input variable combined in likelihood estimator

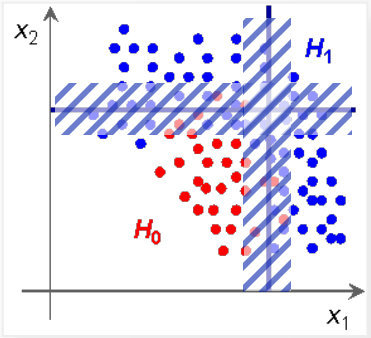
Likelihood ratio for event i_{event}

PDFs

discriminating variables

$$y_L(i_{\text{event}}) = \frac{\prod_{k\in\{\text{variables}\}} p_k^{\text{signal}}(x_k(i_{\text{event}}))}{\sum_{U\in\{\text{species}\}} \left(\prod_{k\in\{\text{variables}\}} p_k^{U}(x_k(i_{\text{event}})) \right)}$$

Species: signal, background types



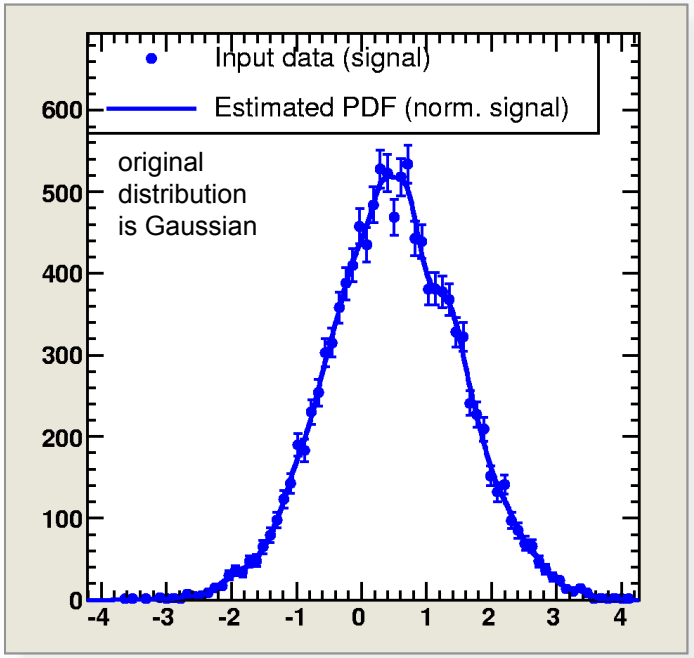
PDE introduces fuzzy logic

- Ignores correlations between input variables
  - Optimal approach if correlations are zero (or linear → decorrelation)
  - Otherwise: significant performance loss

* also denoted “Naïve Bayes”

PDE Approach: Estimating PDF Kernels

- Technical challenge: how to estimate the PDF shapes
  - **3 ways:** **parametric fitting (function)** | **nonparametric fitting** | **event counting**
    - Difficult to automate for arbitrary PDFs
    - Easy to automate, can create artefacts/suppress information
    - Automatic, unbiased, but sub-optimal
- *T*MVA uses nonparametric fitting
  - Binned shape interpolation using spline functions and adaptive smoothing
  - Unbinned adaptive kernel density estimation (KDE) with Gaussian smearing
  - *T*MVA performs automatic validation of goodness-of-fit



Multidimensional PDE Approach

- Use a single PDF per event class (sig, bkg), which spans N_{var} dimensions
  - PDE Range-Search: count number of signal and background events in "vicinity" of test event → preset or **adaptive** volume defines "vicinity"

k-Nearest Neighbor

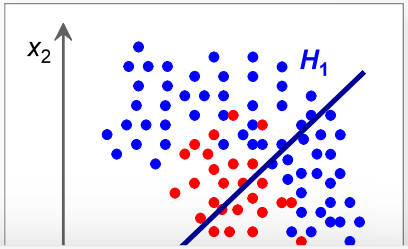
Better than searching within a volume (fixed or floating), count adjacent reference events till statistically significant number reached

- Method intrinsically adaptive
- Very fast search with kd-tree event sorting

- Improve y_{PDERS} estimate within V by using various N_{var}-D kernel estimators
- Enhance speed of event counting in volume by binary tree search

Fisher's Linear Discriminant Analysis (LDA)

- Well known, simple and elegant classifier
  - LDA determines axis in the input variable hyperspace such that a projection of events onto this axis pushes signal and background as far away from each other as possible, while confining events of



Function discriminant analysis (FDA)

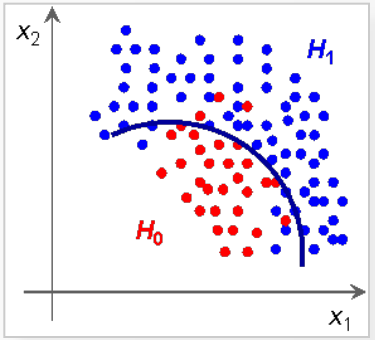
Fit any user-defined function of input variables requiring that signal events return →1 and background →0

- Parameter fitting: Genetics Alg., MINUIT, MC and combinations
- Easy reproduction of Fisher result, but can add nonlinearities
- Very transparent discriminator

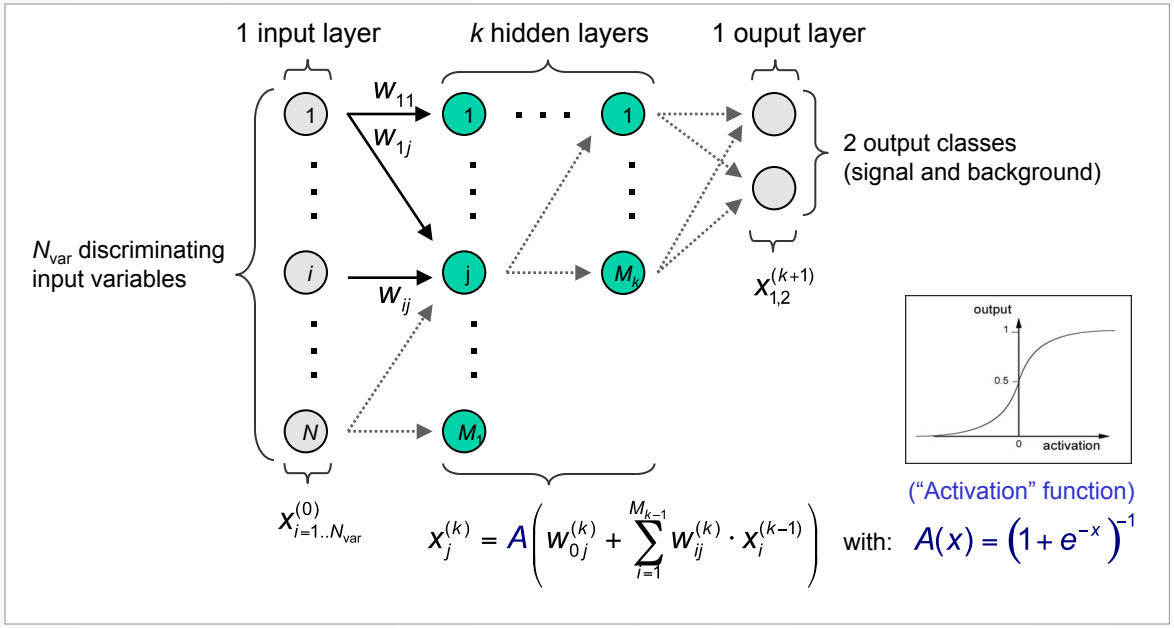
- Compute Fisher coefficients from signal and background covariance matrices
- Fisher requires distinct sample means between signal and background
- Optimal classifier (Bayes limit) for linearly correlated Gaussian-distributed variables

Nonlinear Analysis: Artificial Neural Networks

- Achieve nonlinear classifier response by "activating" output nodes using nonlinear weights



Feed-forward Multilayer Perceptron

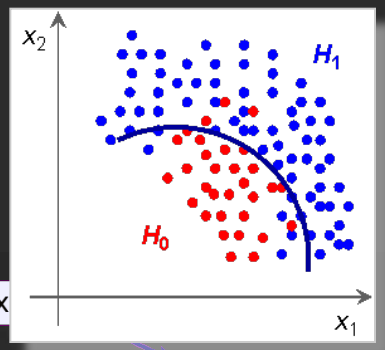


Weight adjustment using analytical back-propagation

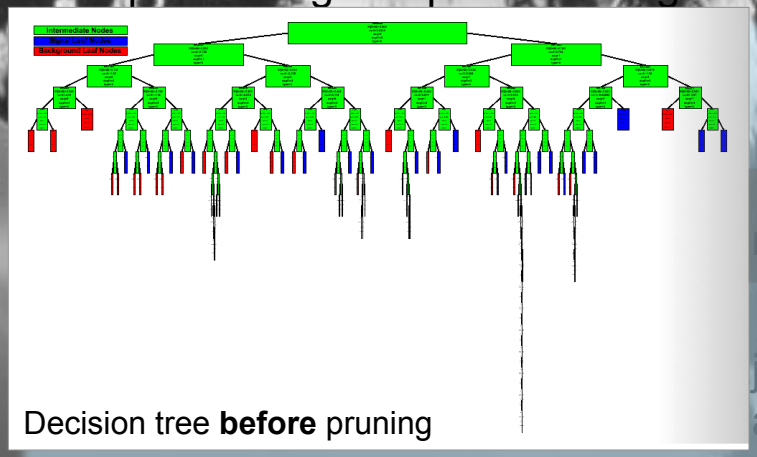
- Three different implementations in TMVA (all are Multilayer Perceptrons)
  - **TMlpANN:** Interface to ROOT's MLP implementation
  - **MLP:** TMVA's own MLP implementation for increased speed and flexibility
  - **CFMlpANN:** ALEPH's Higgs search ANN, translated from FORTRAN

Decision Trees

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



- Growing a decision tree:
  - Start with Root node
  - Split training sample according to



- Bottom-up "pruning" of the decision tree
  - Remove statistically insignificant nodes

- How to grow and prune a decision tree ?
  - Recursive splitting of binary node splits

Boosted Decision Trees (BDT)

- Data mining with decision trees is popular in science (so far mostly outside of HEP)
  - Advantages:
    - Independent of monotonous variable transformations, immune against outliers
    - Weak variables are ignored (and don't (much) deteriorate performance)
  - Shortcomings:
    - Instability: small changes in training sample can dramatically alter the tree structure
    - Sensitivity to overtraining (→ requires pruning)

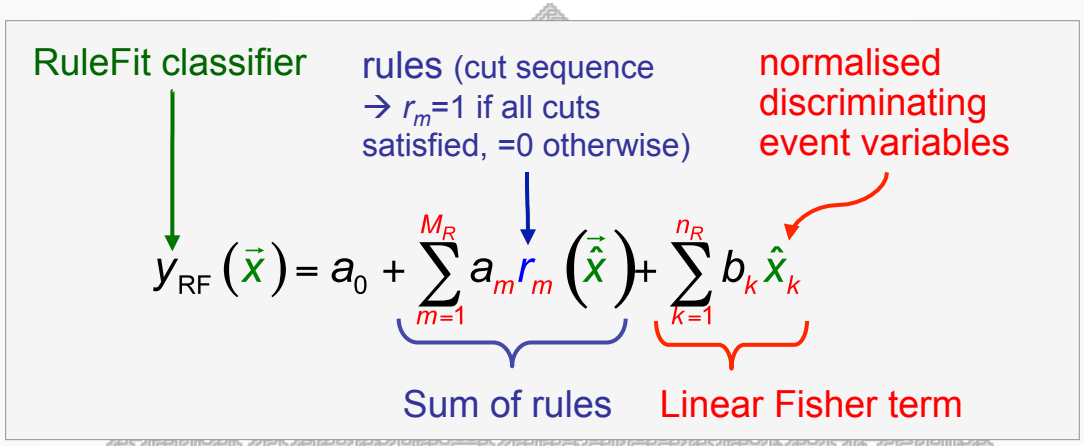
- *Boosted* decision trees: combine *forest* of decision trees, with differently weighted events in each tree (trees can also be weighted), by majority vote
  - e.g., "AdaBoost": incorrectly classified events receive larger weight in next decision tree
  - "Bagging" (instead of boosting): random event weights, resampling with replacement
  - Boosting or bagging are means to create set of "basis functions": the final classifier is linear combination (*expansion*) of these functions → improves stability !

Predictive Learning via Rule Ensembles (RuleFit)

- Following RuleFit approach by Friedman-Popescu

Friedman-Popescu, Tech Rep, Stat. Dpt, Stanford U., 2003

- Model is linear combination of *rules*, where a rule is a sequence of cuts

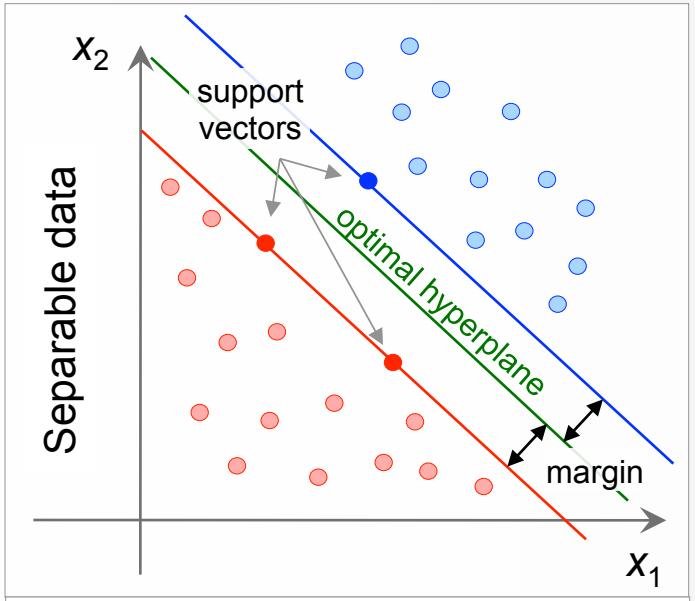


- The problem to solve is
  - Create rule ensemble: use forest of decision trees
  - Fit coefficients a_m, b_k: gradient direct regularization minimising *Risk* (Friedman et al.)
- Pruning removes topologically equal rules" (same variables in cut sequence)

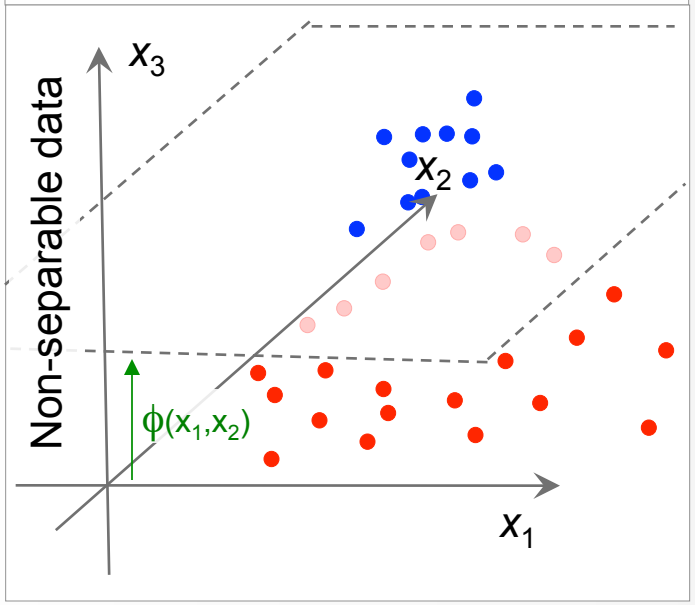
One of the elementary cellular automaton rules (Wolfram 1983, 2002). It specifies the next color in a cell, depending on its color and its immediate neighbors. Its rule outcomes are encoded in the binary representation 30=00011110$_2$.

Support Vector Machine (SVM)

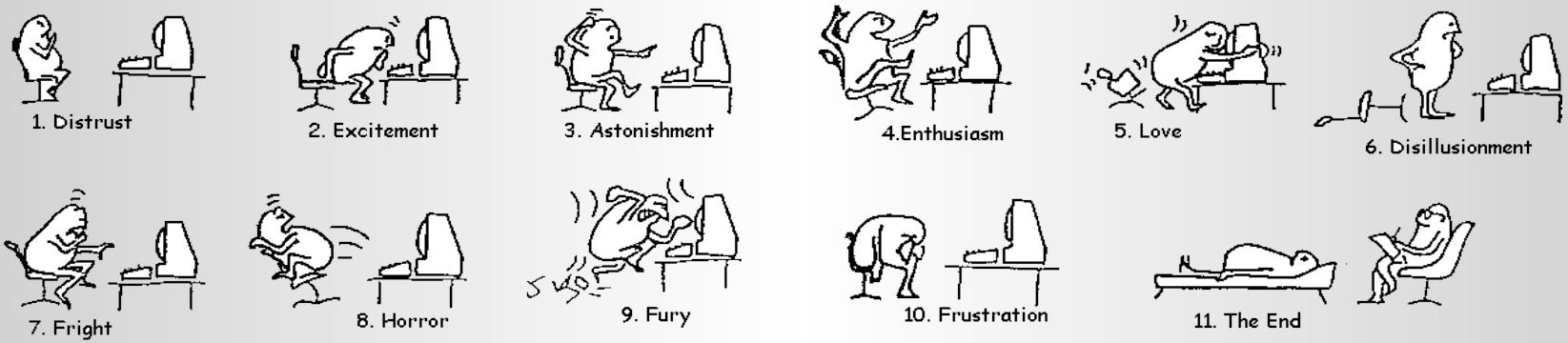
- Linear case: find hyperplane that best separates signal from background
  - Best separation: maximum distance (margin) between closest events (*support*) to hyperplane
  - Linear decision boundary
  - If data non-separable add *misclassification cost* parameter to minimisation function



- Non-linear cases:
  - Transform variables into higher dim. space where a linear boundary can fully separate the data
  - Explicit transformation not required: use kernel functions to approximate scalar products between transformed vectors in the higher dim. space
  - Choose Kernel and fit the hyperplane using the techniques developed for linear case



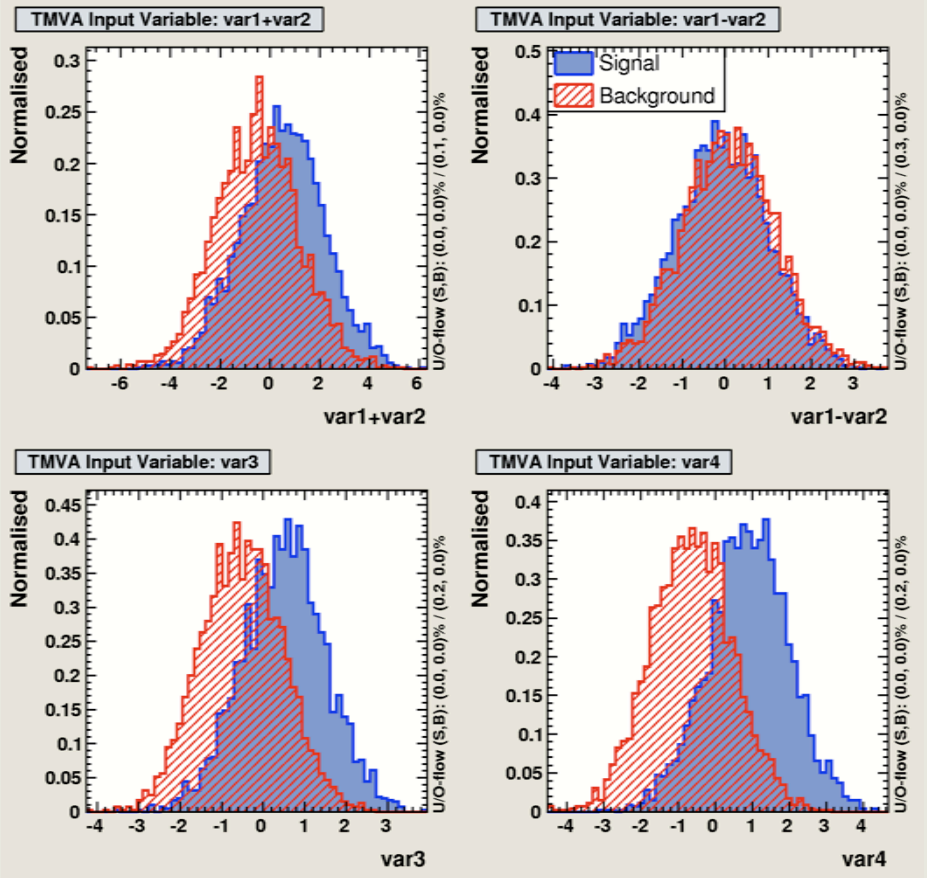
Examples

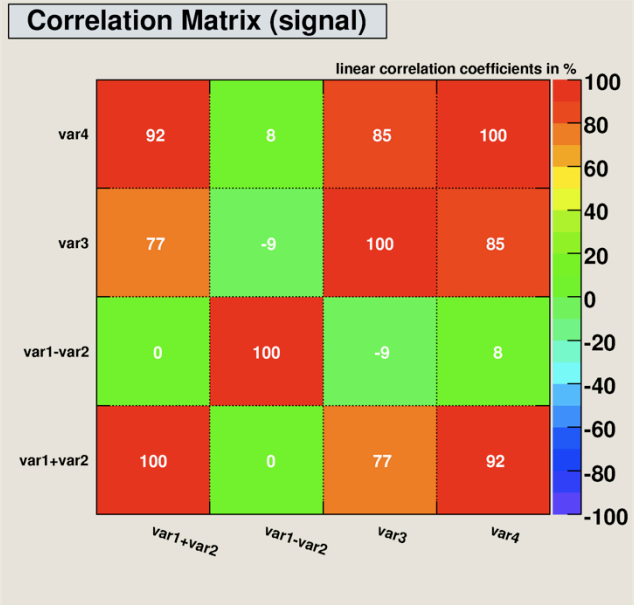


→ *T*MVA tutorial

A Toy Example (idealized)

- Use data set with 4 linearly correlated Gaussian distributed variables:





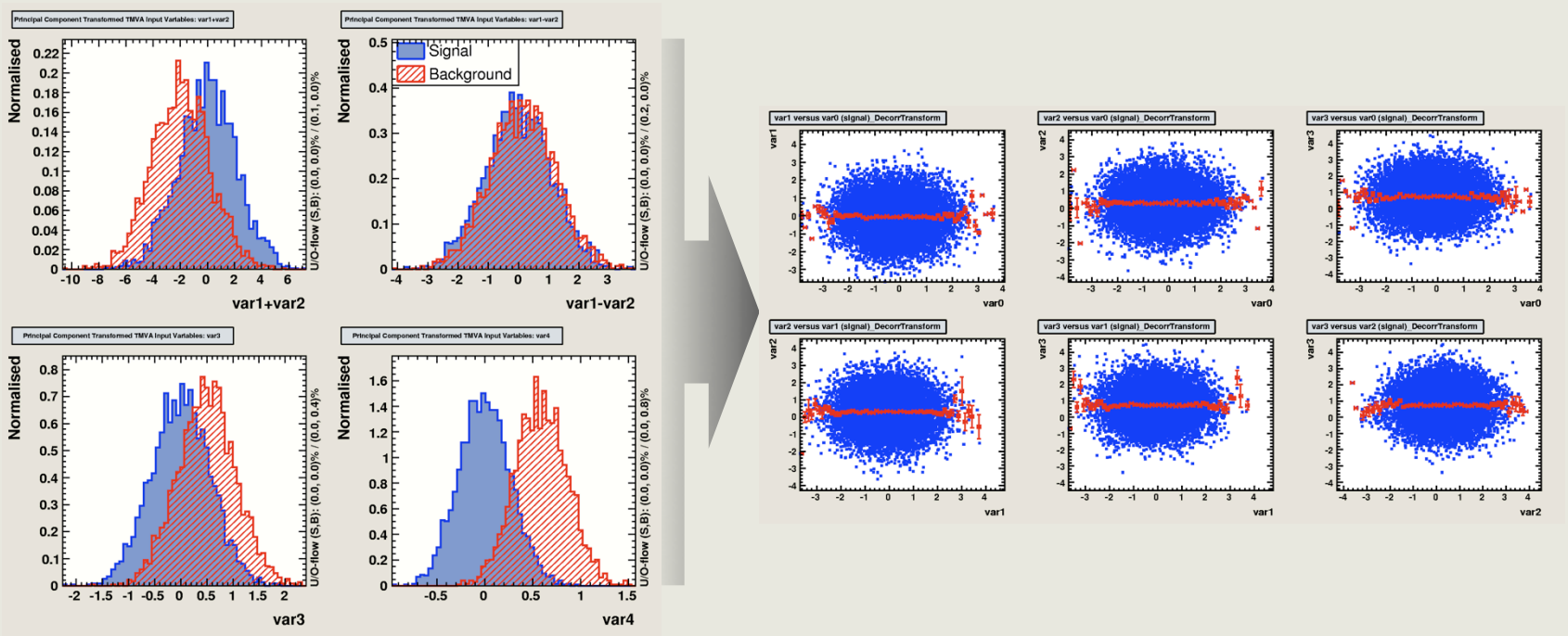
Rank : Variable : Separation

---

1 : var4 : 0.606
2 : var1+var2 : 0.182
3 : var3 : 0.173
4 : var1-var2 : 0.014

Preprocessing the Input Variables

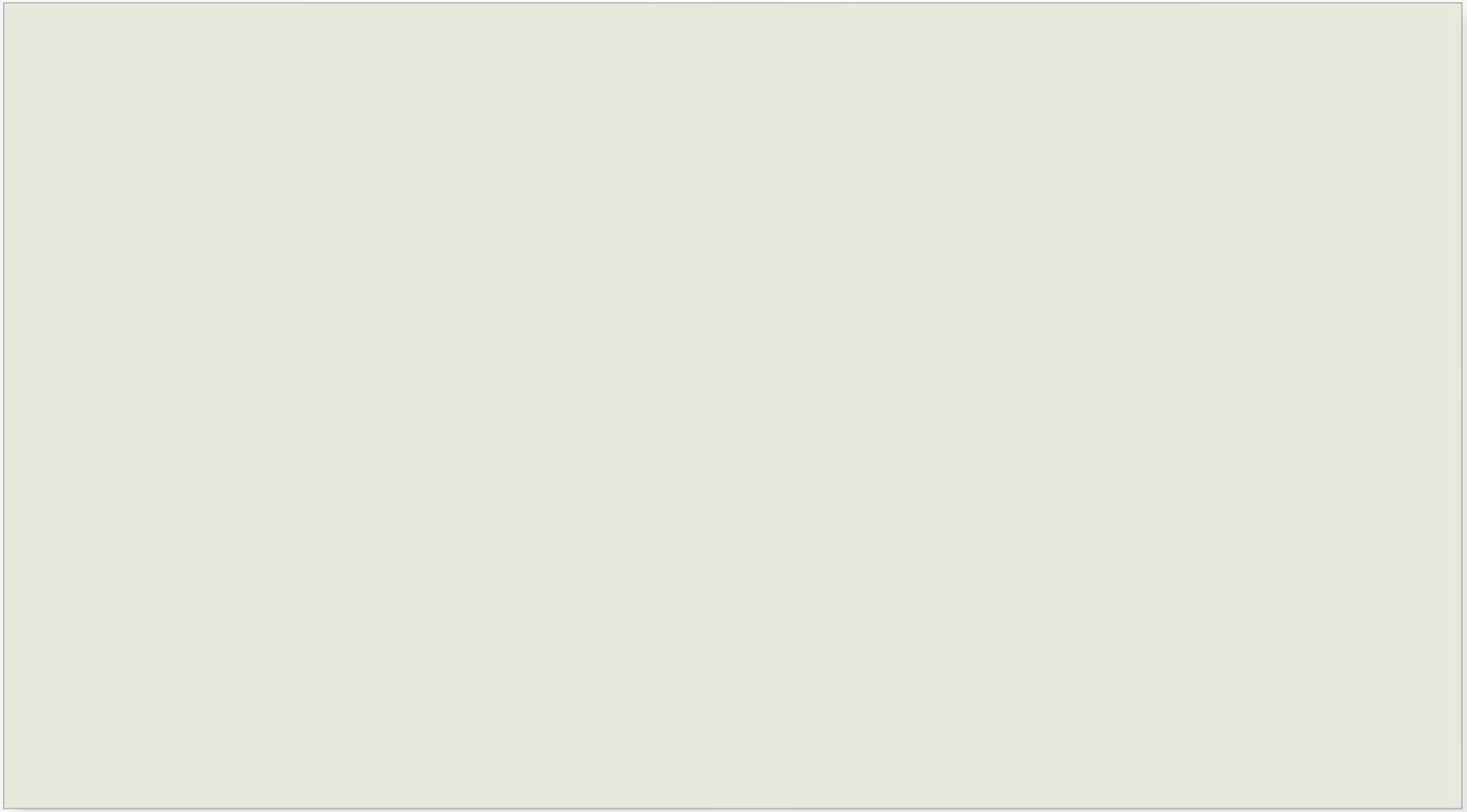
- Decorrelation of variables before training is useful for *this* example



- Note that in cases with non-Gaussian distributions and/or nonlinear correlations decorrelation may do more harm than any good

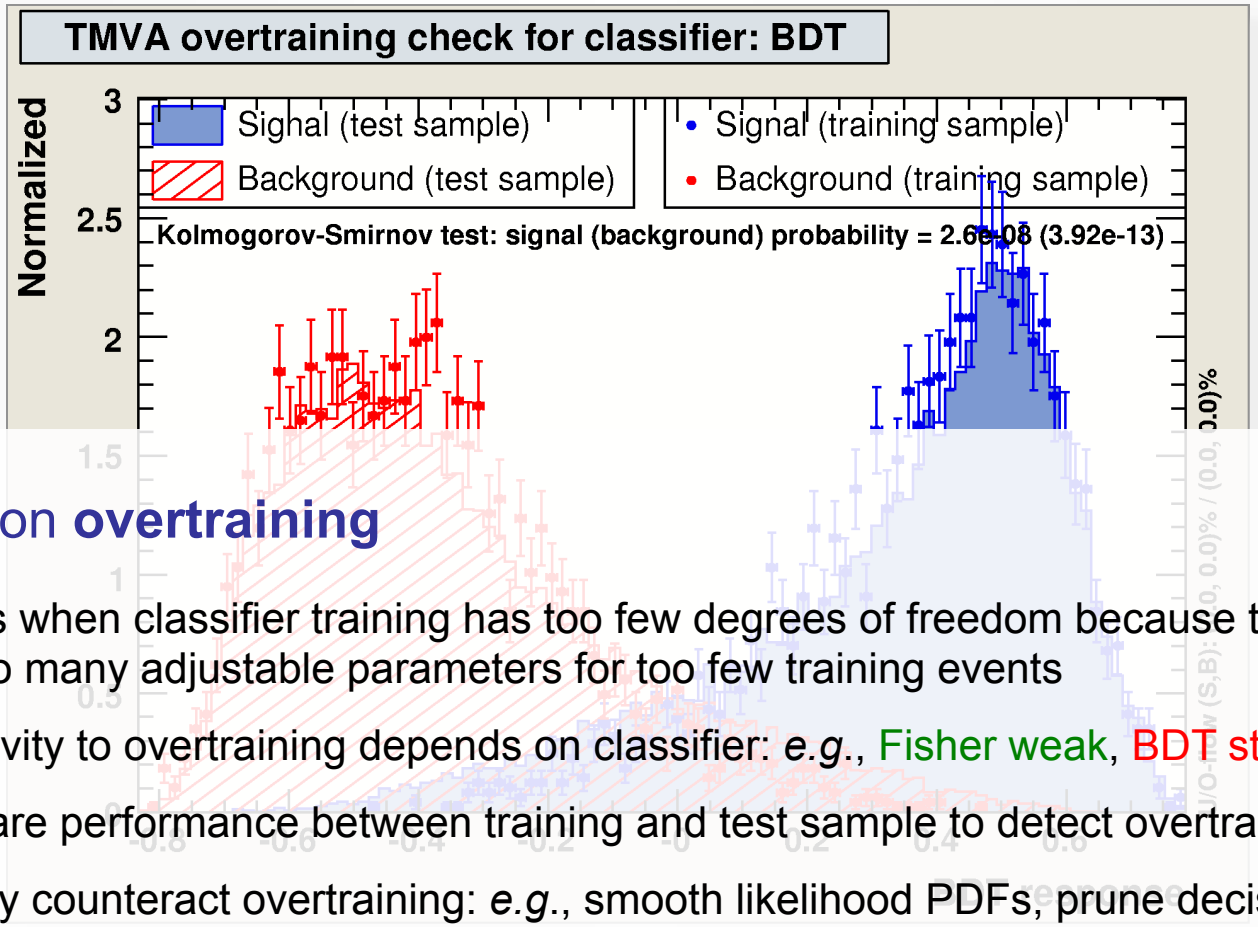
Testing the Classifiers

- Classifier output distributions y(x) for independent test sample:



Evaluating the Classifier Training (II)

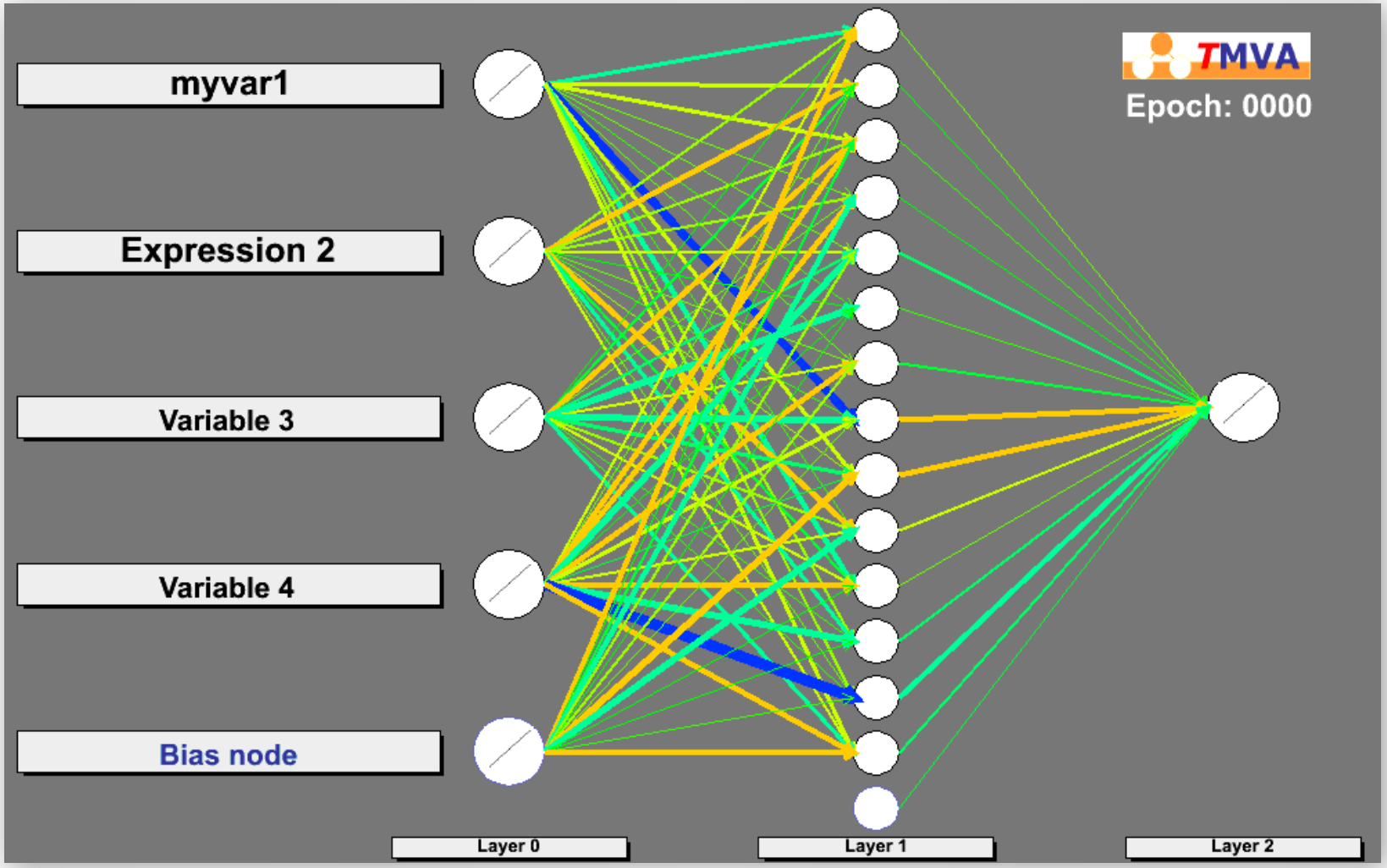
- Check for overtraining: classifier output for test *and* training samples …



- Remark on **overtraining**
  - Occurs when classifier training has too few degrees of freedom because the classifier has too many adjustable parameters for too few training events
  - Sensitivity to overtraining depends on classifier: *e.g.*, Fisher weak, BDT strong
  - Compare performance between training and test sample to detect overtraining
  - Actively counteract overtraining: *e.g.*, smooth likelihood PDFs, prune decision trees, …

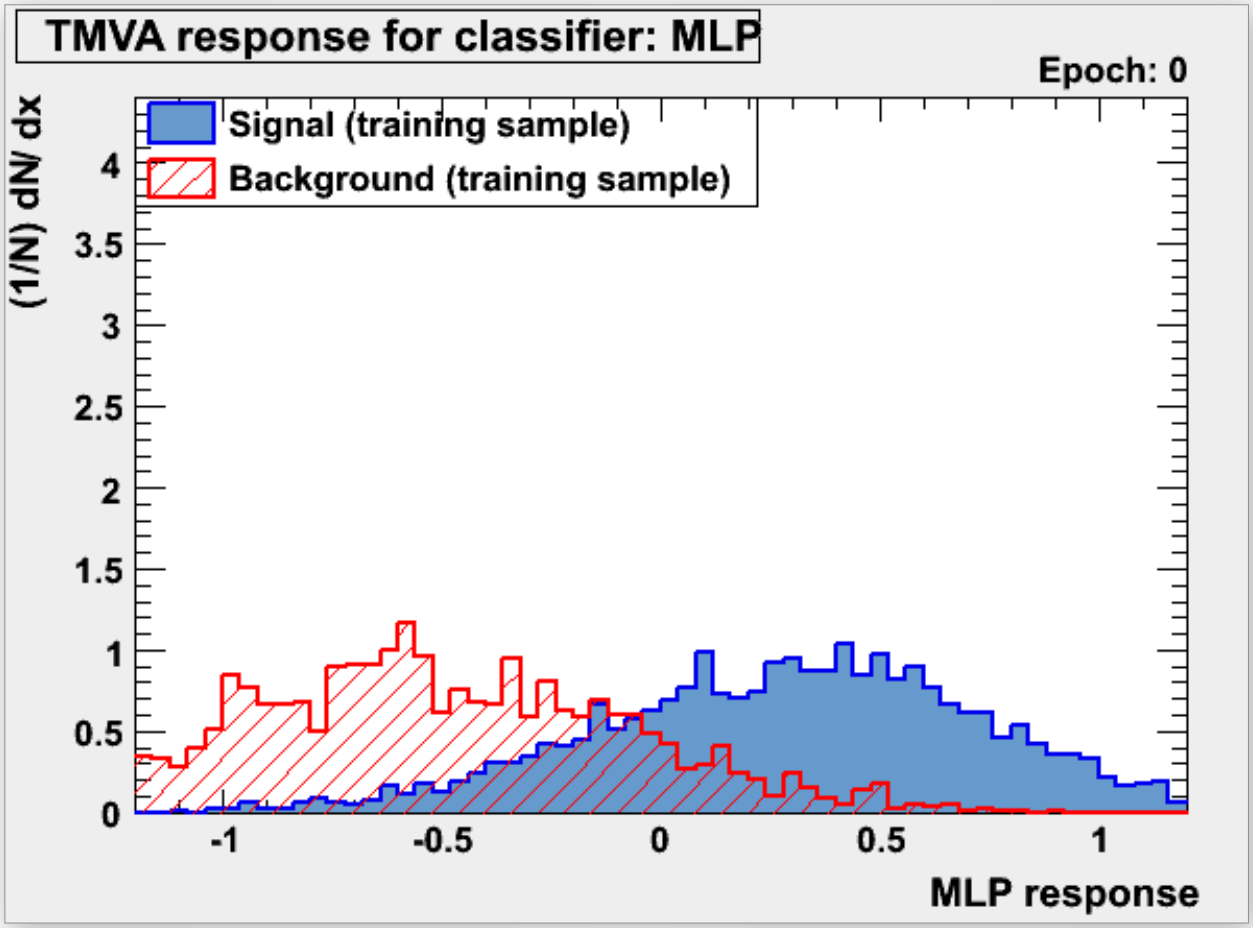
Watching at the Training Progress

- For MLP, plot architecture after each training epoch



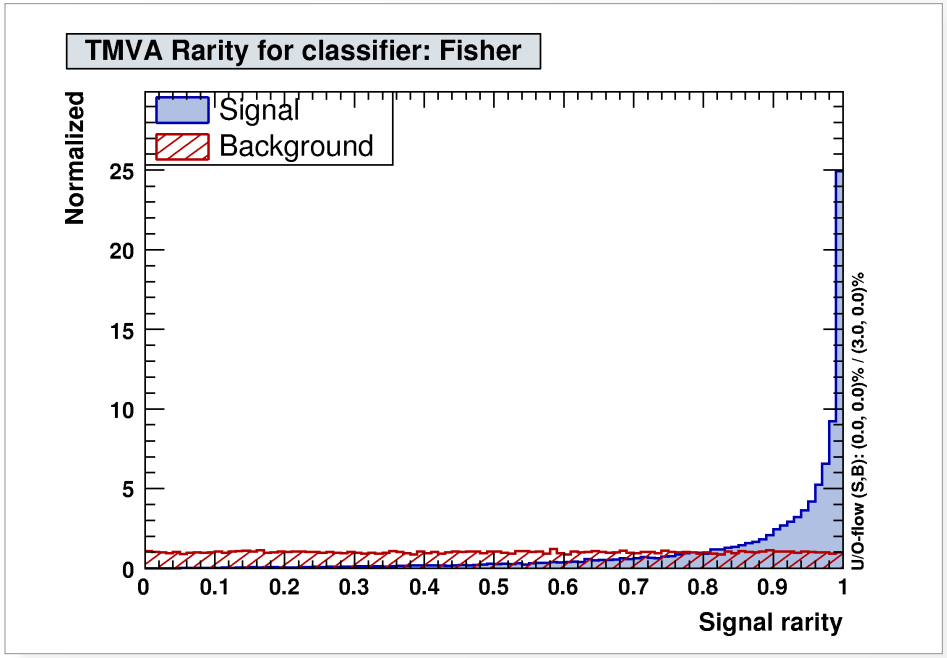
Watching at the Training Progress

- For MLP, plot output distributions after each training epoch



Evaluating the Classifier Training (IV)

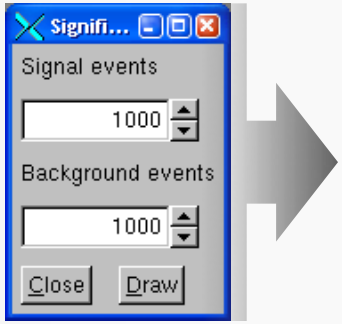
- There is no unique way to express the performance of a classifier
  → several benchmark quantities computed by *T*MVA
  - Signal eff. at various background effs. (= 1 – rejection) when cutting on classifier output
  - The *Separation*: $\frac{1}{2}\int \frac{(\hat{y}_S(y) - \hat{y}_B(y))^2}{\hat{y}_S(y) + \hat{y}_B(y)} dy$
  - "Rarity" implemented (background flat): $R(y) = \int_{-\infty}^{y} \hat{y}(y')dy'$

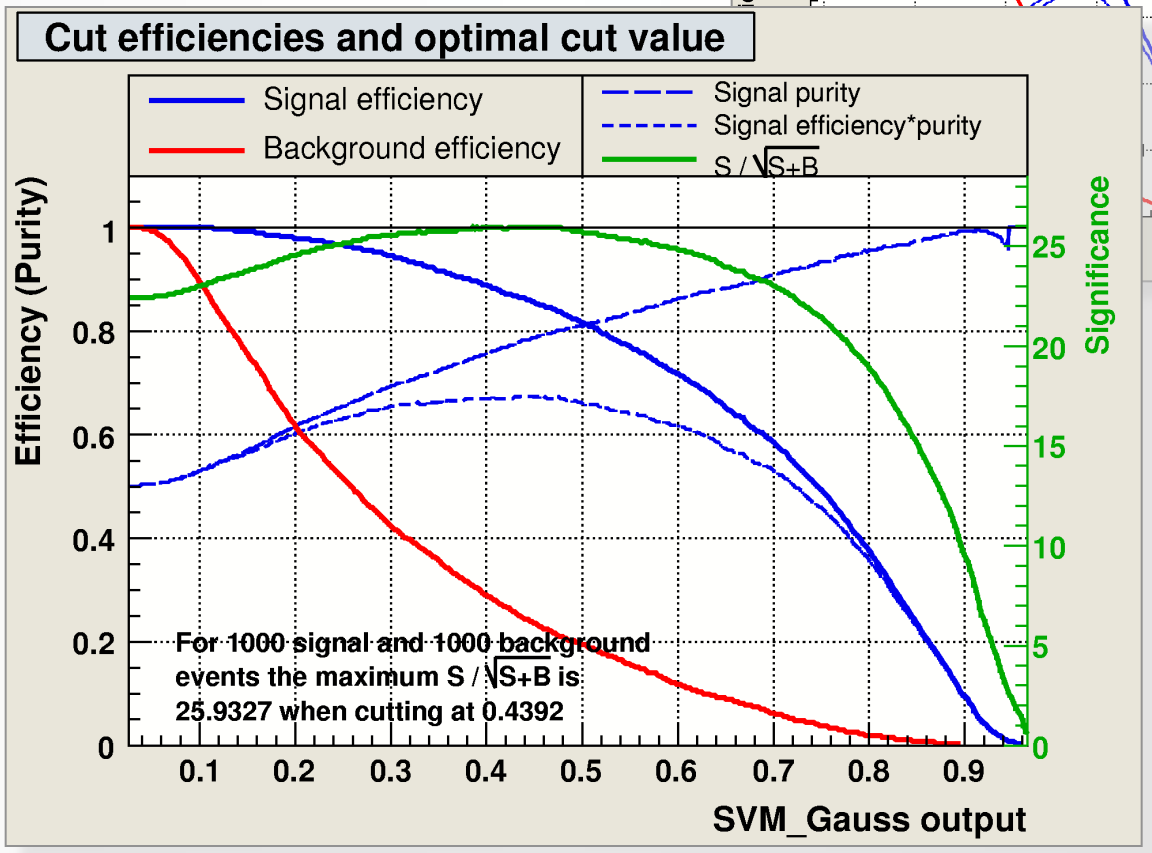


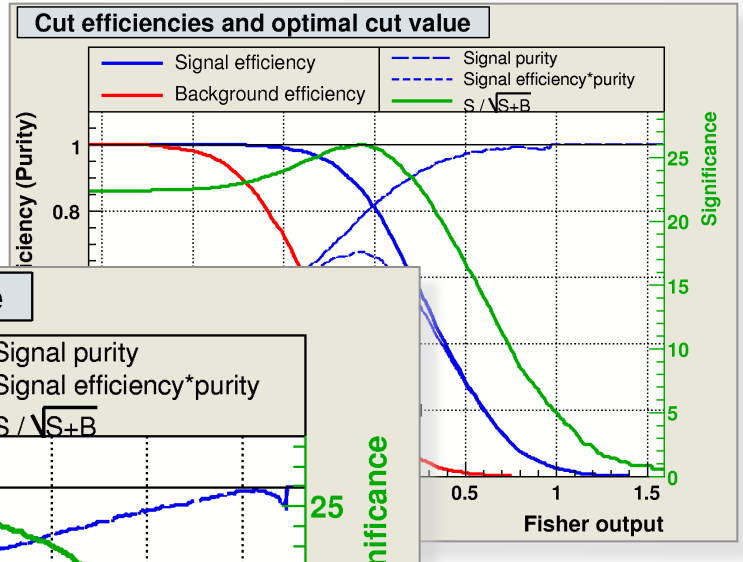
Evaluating the Classifier Training (V)

- Optimal cut for each classifiers …

  Determine the optimal cut (working point) on a classifier output







Evaluating the Classifiers Training (VI) (taken from *T*MVA output…)

Input Variable Ranking

```
--- 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 discriminating is a variable ?

Classifier correlation and overlap

```
--- 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 select the same events as signal and background ?
  If not, there is something to gain !

Evaluating the Classifiers Training (VII) (taken from TMVA output…)

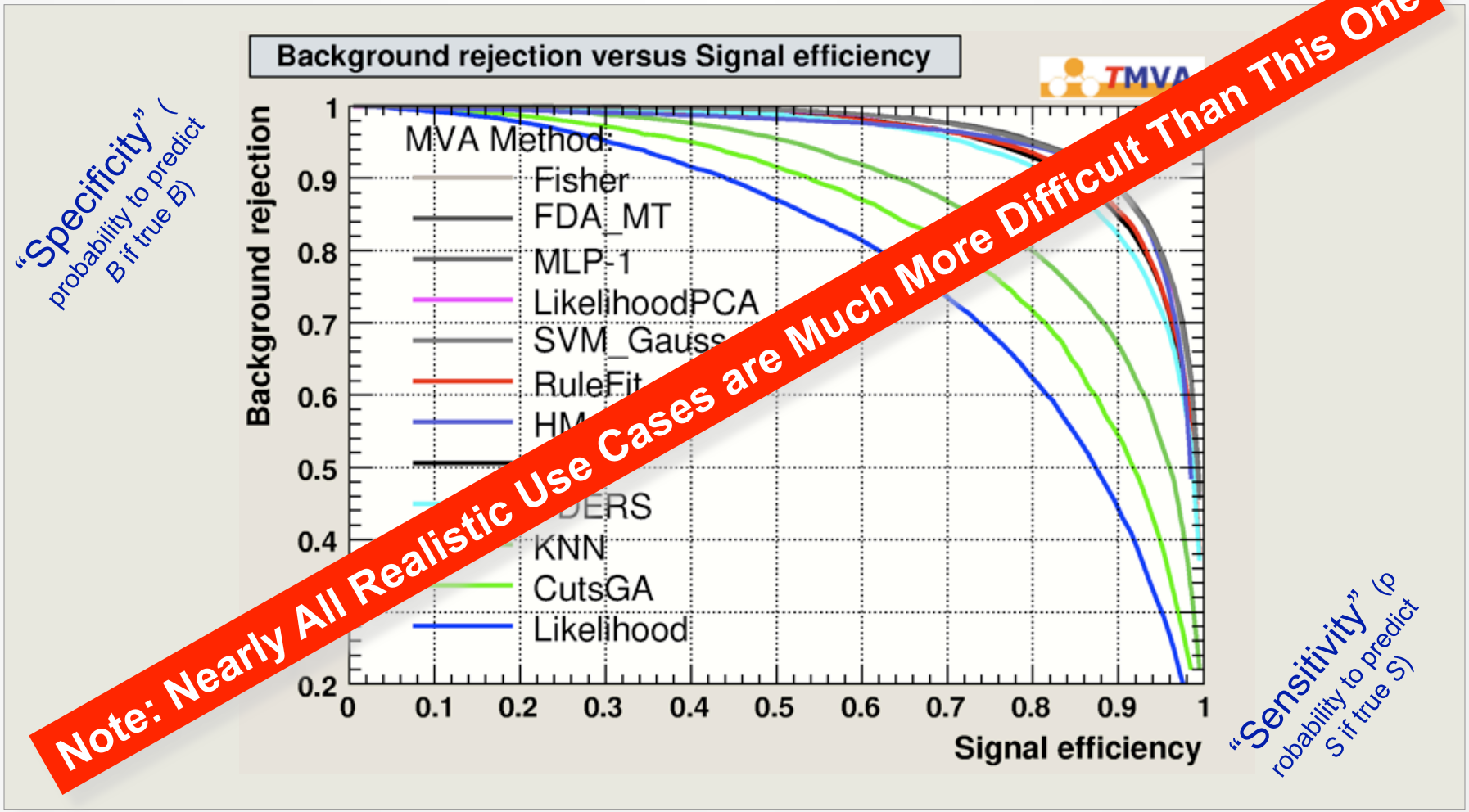
Evaluation results ranked by best signal efficiency and purity (area)

MVA Methods:	Signal efficiency at bkg eff. (error): @B=0.01	@B=0.10	@B=0.30	Area	Separation:	Significance:
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

Better classifier

Receiver Operating Characteristics (ROC) Curve

- Smooth background rejection versus signal efficiency curve:
  (from cut on classifier output)

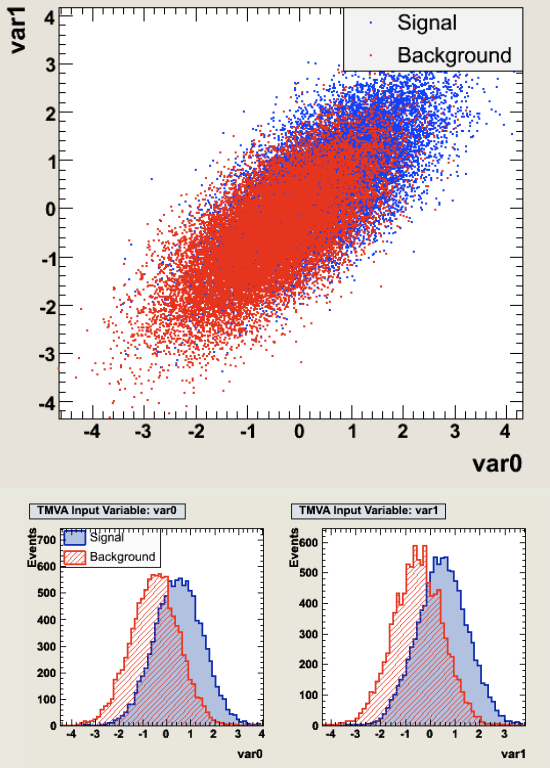


More Toy Examples

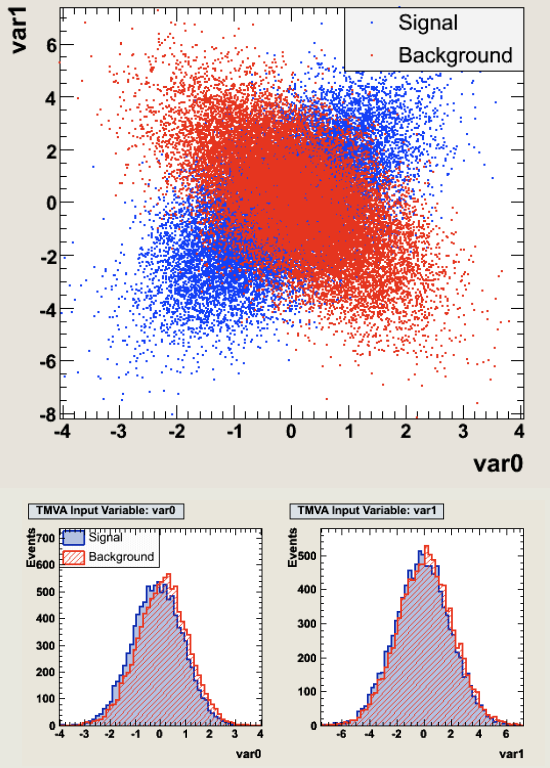
More Toys: Linear-, Cross-, Circular Correlations

- Illustrate the behaviour of linear and nonlinear classifiers

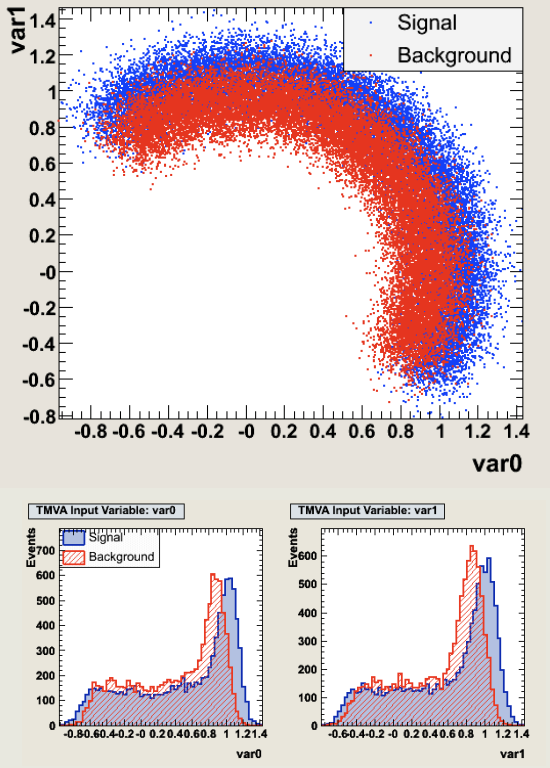
Linear correlations
(same for signal and background)



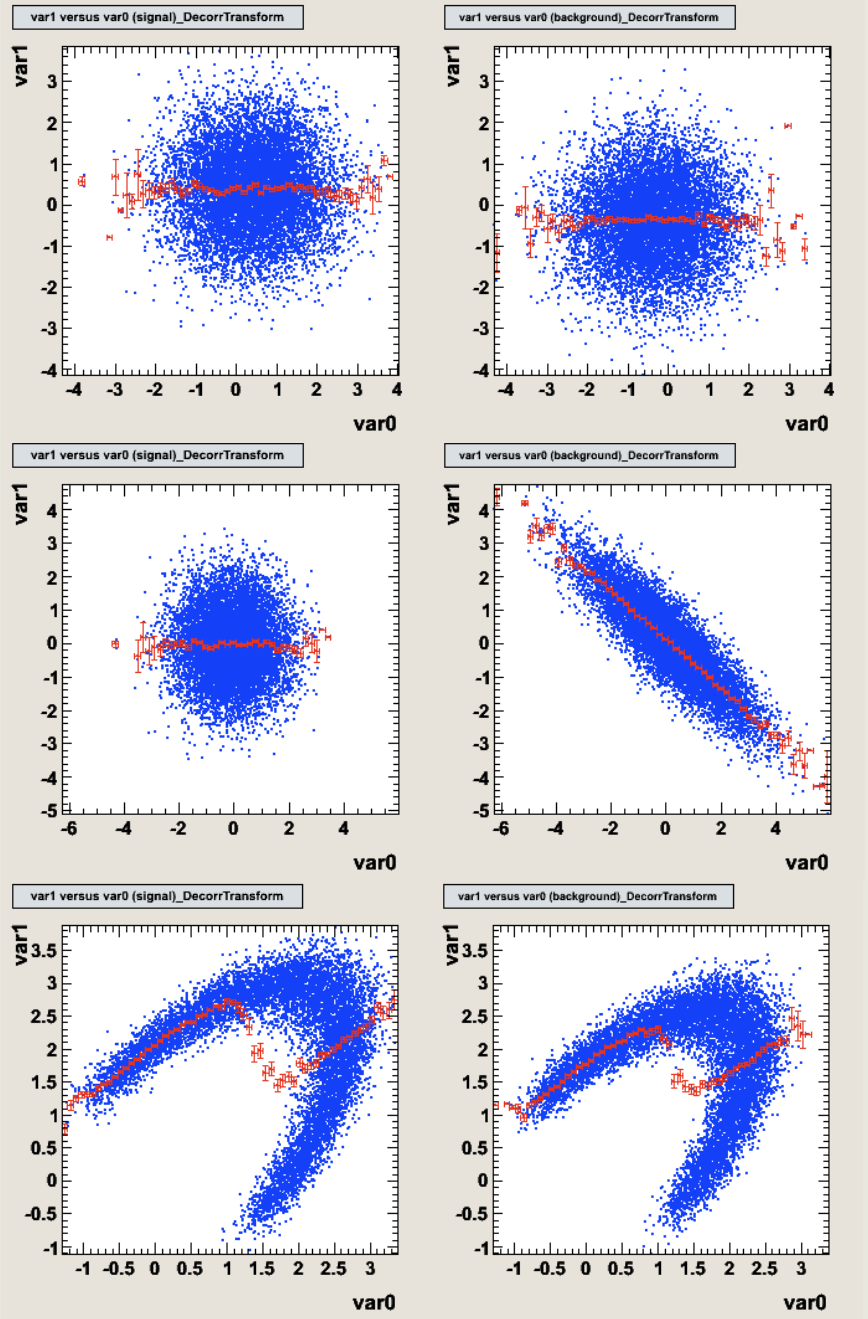
Linear correlations
(opposite for signal and background)



Circular correlations
(same for signal and background)



- How does linear decorrelation affect strongly nonlinear cases ?

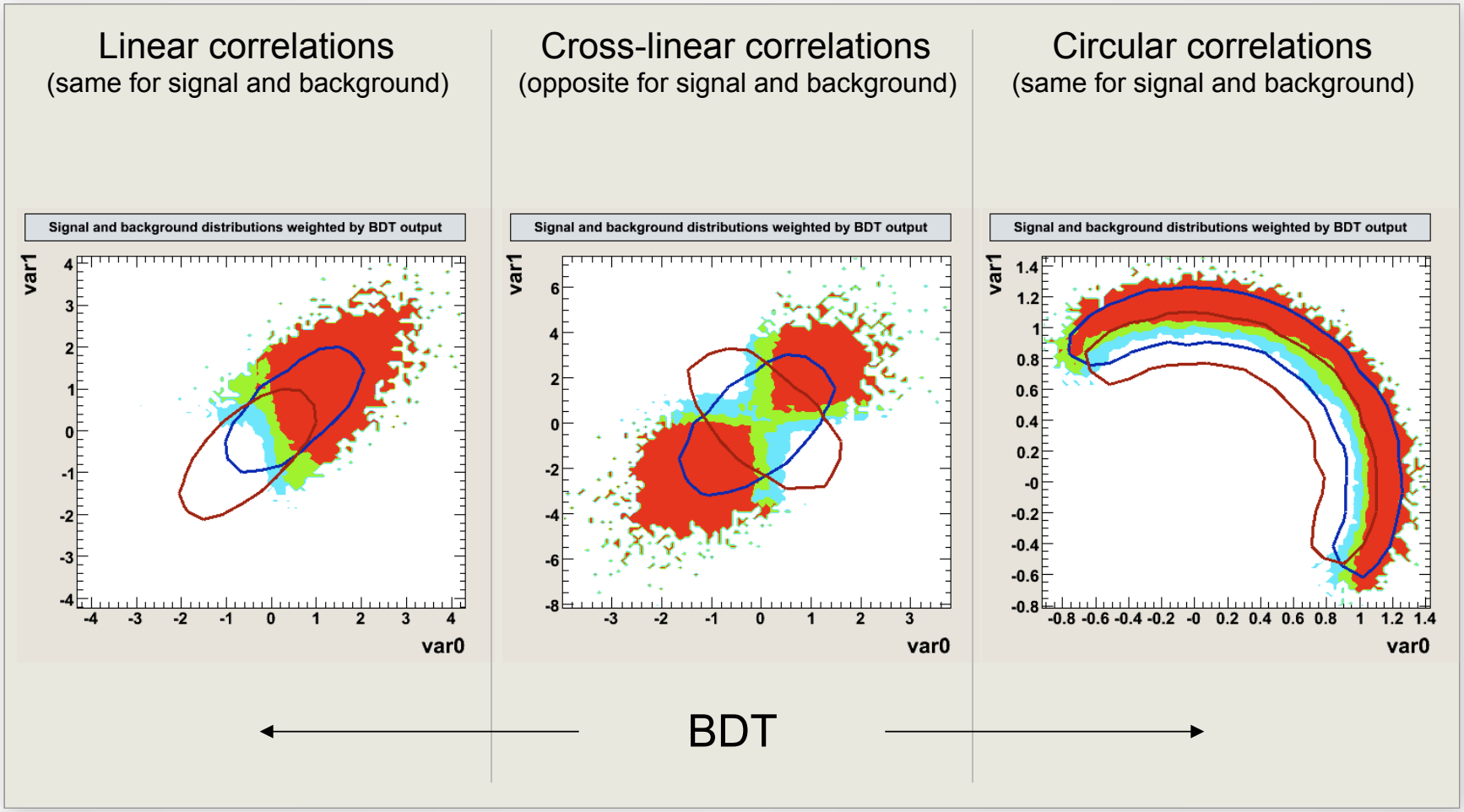


Original correlations

SQRT decorrelation

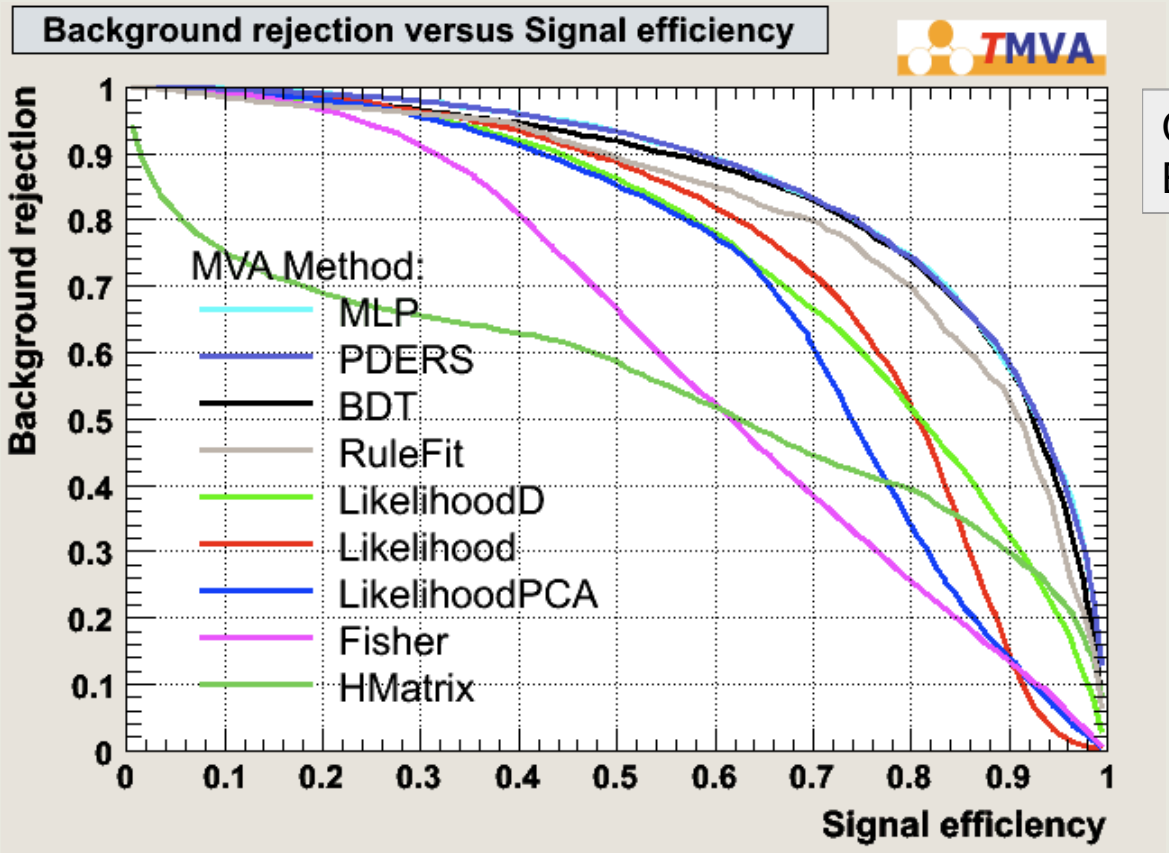
Weight Variables by Classifier Output

- How well do the classifier resolve the various correlation patterns ?



Final Classifier Performance

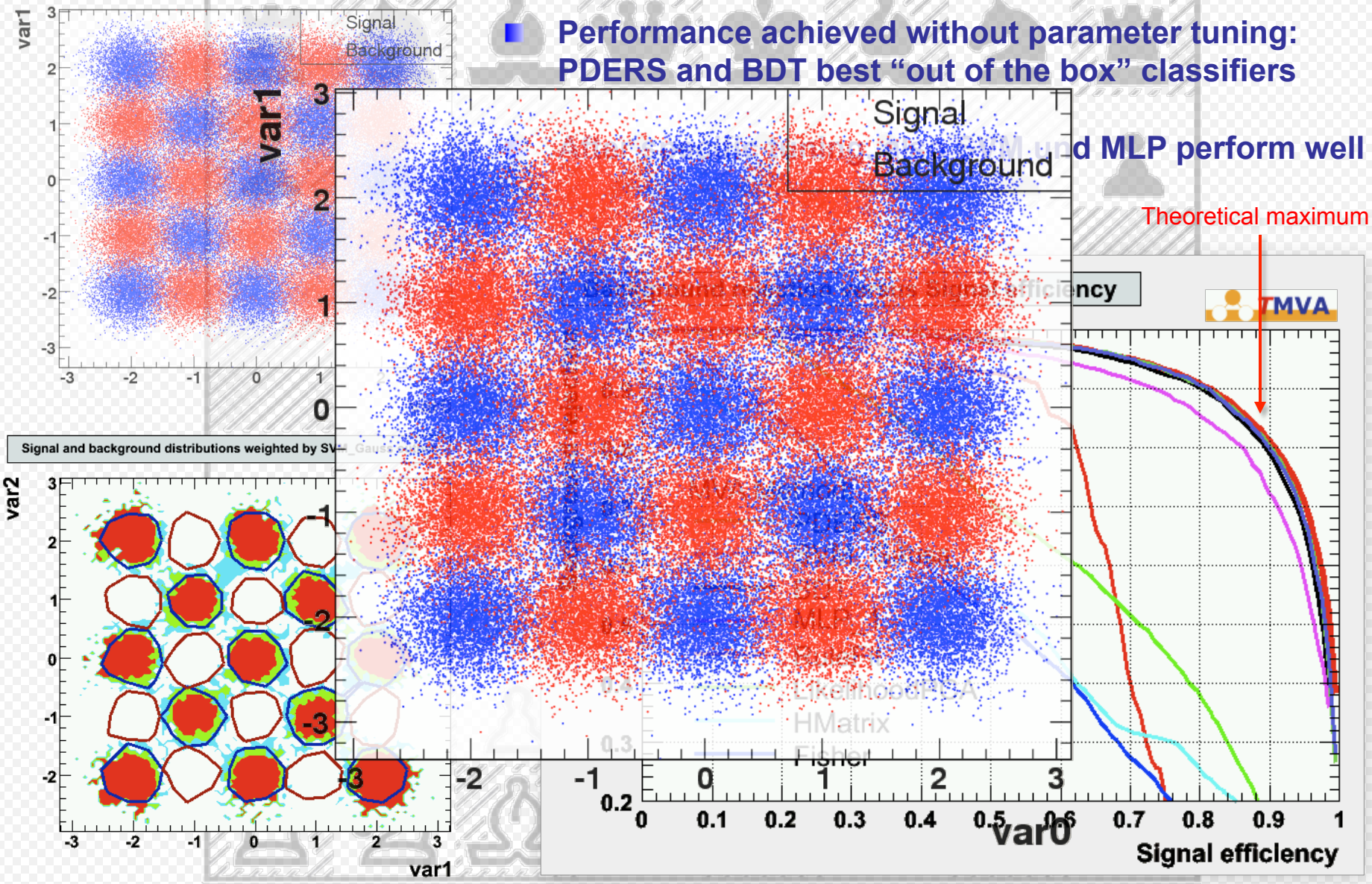
- Background rejection versus signal efficiency curve:



Circular Example

The *Schachbrett* Toy

- **Performance achieved without parameter tuning: PDERS and BDT best "out of the box" classifiers**
- **d MLP perform well**



Combining Classifers

- Generalised Boosting of any classifier
- Categorising classifiers
- Committee classifiers

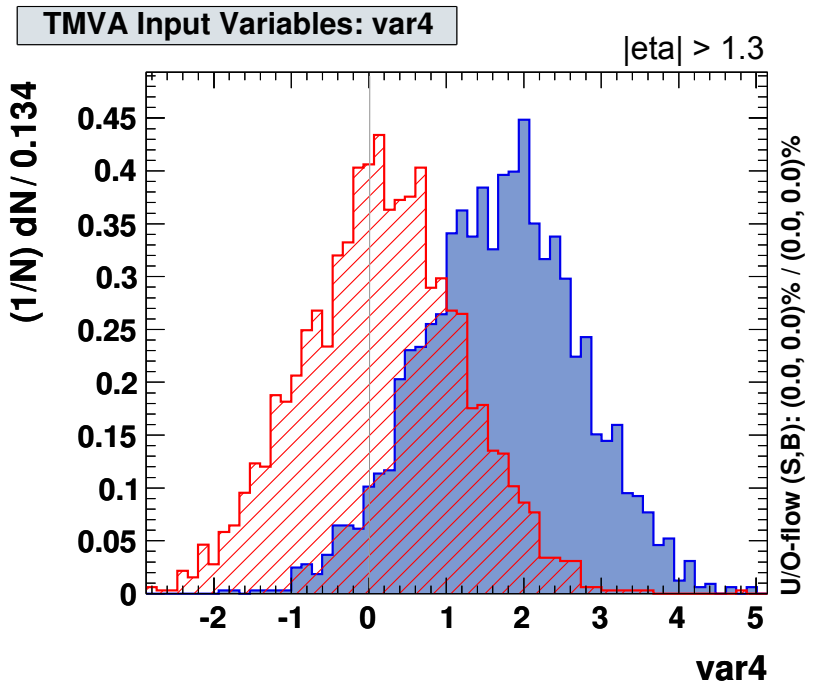
Categorising Classifiers

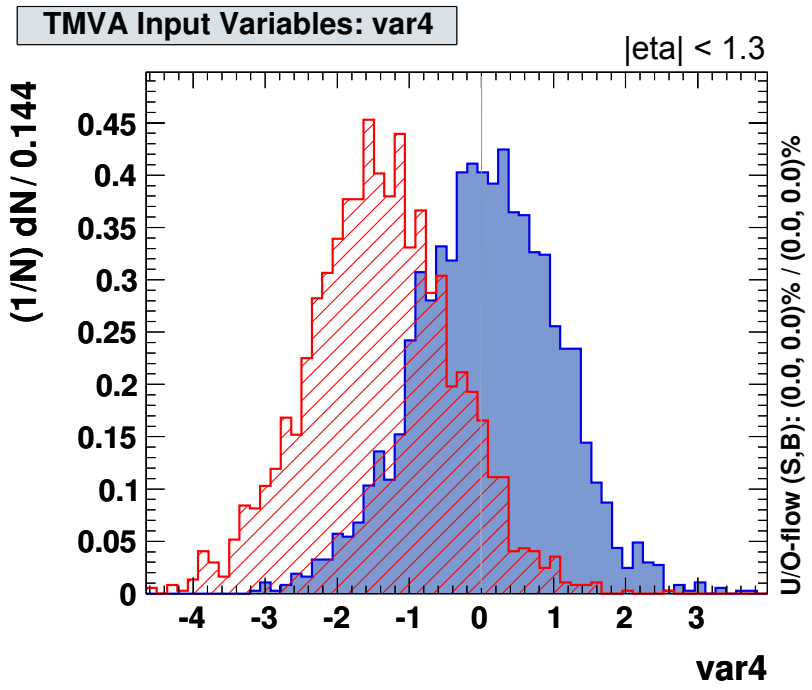
- Multivariate training samples often have distinct sub-populations of data
  - A detector element may only exist in the barrel, but not in the endcaps
  - A variable may have different distributions in barrel, overlap, endcap regions
- Ignoring this dependence creates correlations between variables, which must be learned by the classifier
  - Classifiers such as the projective likelihood, which do not account for correlations, significantly loose performance if the sub-populations are not separated
- Categorisation means splitting the data sample into categories defining disjoint data samples with the following (idealised) properties:
  - Events belonging to the same category are statistically indistinguishable
  - Events belonging to different categories have different properties
- All categories are treated independently for training and application (transparent for user), but evaluation is done for the whole data sample

Categorising Classifiers

- Let's try our standard example of 4 Gaussian-distributed input variables:
  - Now, "var4" depends on a new variable "eta" (which may not be used for classification)

→ for |eta| > 1.3 the Signal and Background Gaussian means are shifted w.r.t. |eta| < 1.3

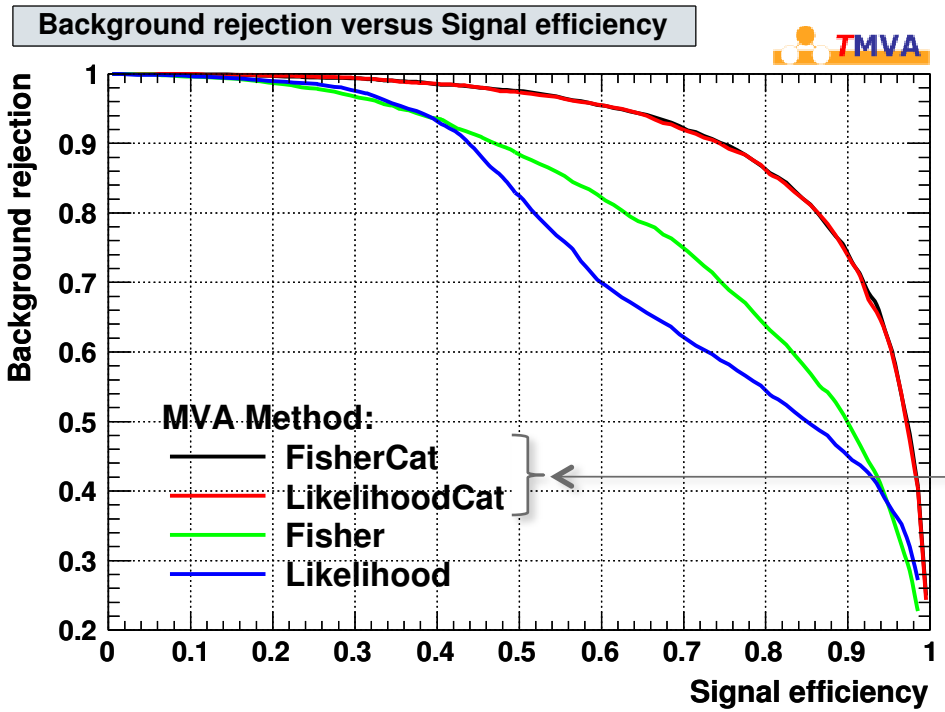




Categorising Classifiers

- Let's try our standard example of 4 Gaussian-distributed input variables:
  - Now, "var4" depends on a new variable "eta" (which may not be used for classification)

→ for |eta| > 1.3 the Signal and Background Gaussian means are shifted w.r.t. |eta| < 1.3



Recover optimal performance after splitting into categories

The category technique is heavily used in multivariate likelihood fits, eg, RooFit (RooSimultaneousPdf)

Wrap up

No Single Best Classifier …

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		☹	🙂	☹	🙂	🙂	😐	🙂	😐	😐
Transparency		🙂	🙂	😐	🙂	🙂	☹	☹	☹	☹

The properties of the Function discriminant (FDA) depend on the chosen function

The HEP community has already a lot of experience with MVA classification

- In particular for rare decay searches, *O*(all) mature experiments use it
- They increase the experiment's sensitivity, and may reduce systematic errors due to a smaller background component
- MVAs are **not** black boxes, but (possibly involved) $R^n \rightarrow R$ mapping functions

---

Need to acquire more experience in HEP with multivariate regression

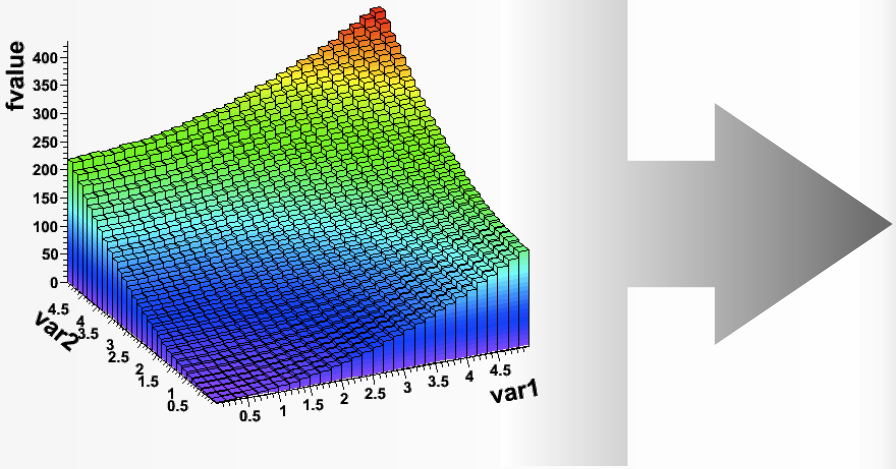
- Our calibration schemes are often still quite simple: linear or simple functions, look-up-table based, mostly depending on few variables (e.g., η, p_T)
- Non-linear multivariate regression may significantly boost calibration and corrections applied, in particular if it is possible to train from data
- Available since *T*MVA 4 for: LD, FDA, k-NN, PDERS, PDEFoam, MLP, BDT

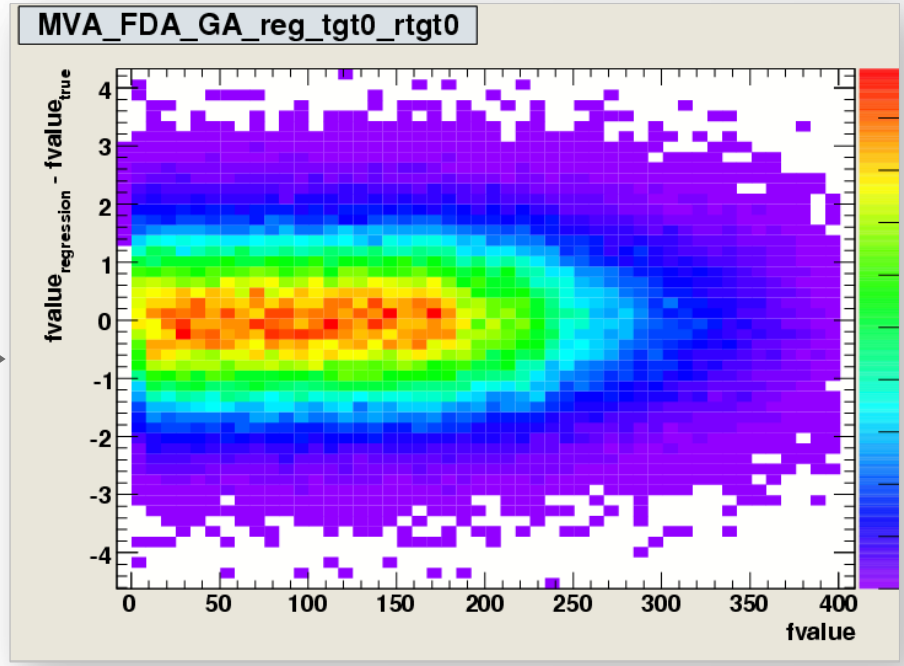
Extra Slides

Regression Analysis

- Regression approximates the functional dependence of a target on $(x_1,\ldots,x_N)$
  - Example: predict the energy correction of jet clusters in calorimeter
- Training: instead of specifying sig/bkgr, provide a regression target
  - Multi-dim target space possible
- Not yet implemented for all methods (so far: LD, PDERS, PDEFoam, MLP, SVM)

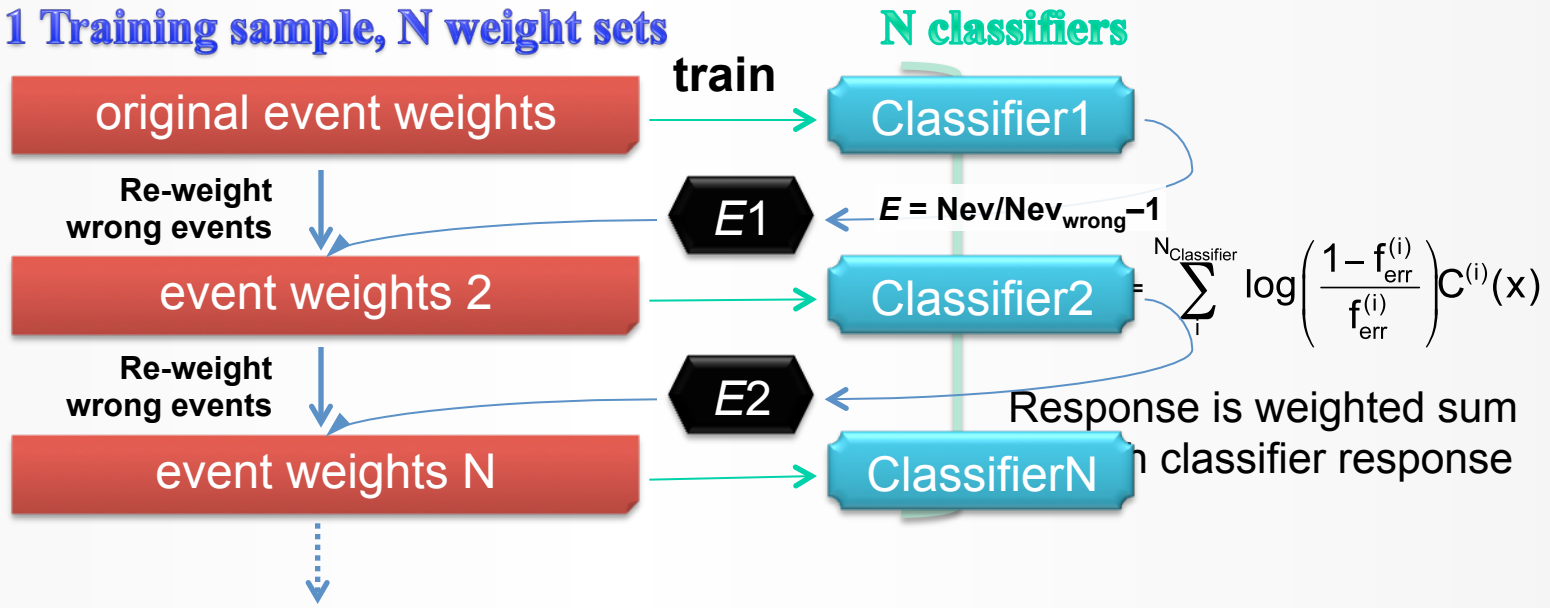
Example:
Target as function of 2 input variables





Generalised Classifier Boosting

- Principle (just as in BDT): multiple training cycles, each time wrongly classified events get a higher event weight



Boosting will be interesting especially for Methods like Cuts, MLP, and SVM

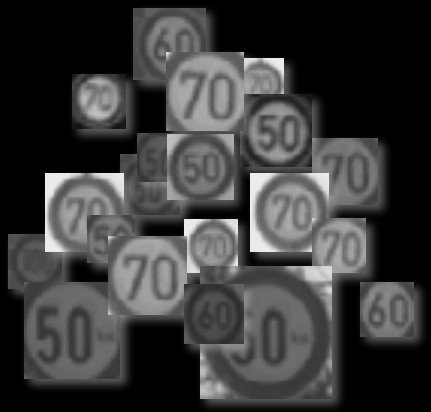
Automated Classifier Tuning via Cross Validation

- Many classifiers have parameters that, being tuned, improves performance
- Method for automated parameter tuning: Cross-Validation
- Special choice of *K*-fold cross-validation:
  - Divide the data sample into *K* sub-sets
  - For set of parameters α train *K* classifiers $C_i(\alpha)$, i=1..K, omitting each time the i-th subset from the training to use as test sample

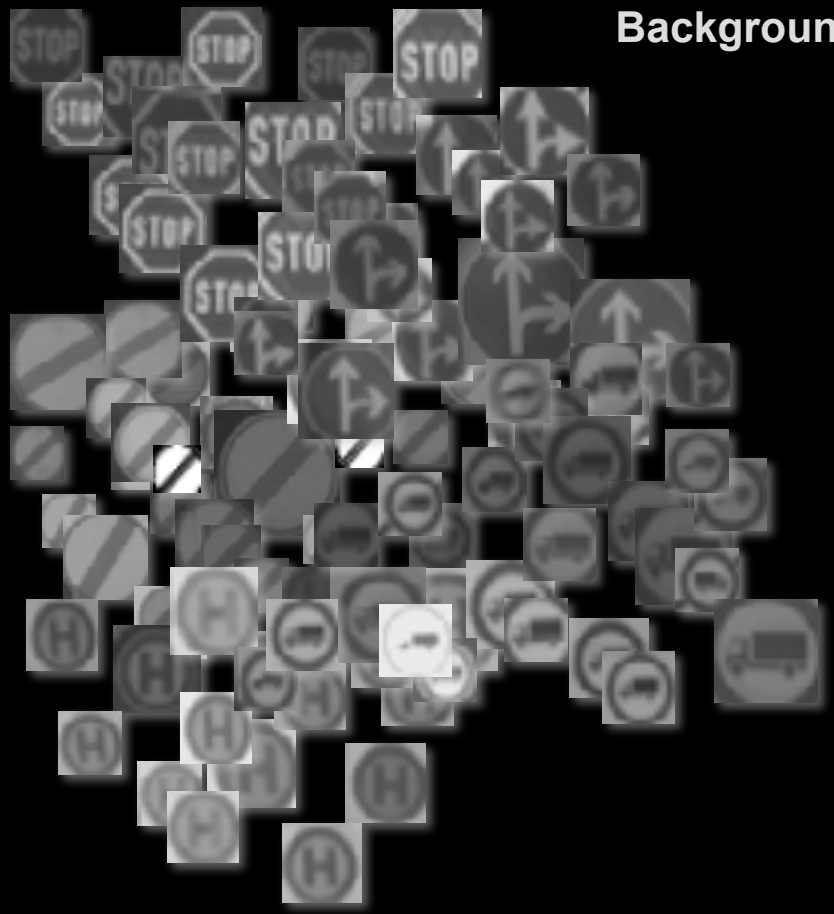
  - Compute performance estimator for each $C_i(\alpha)$ and average among all *K*
  - Choose parameter set α providing the best average estimator

Multi-Class Classification

Signal

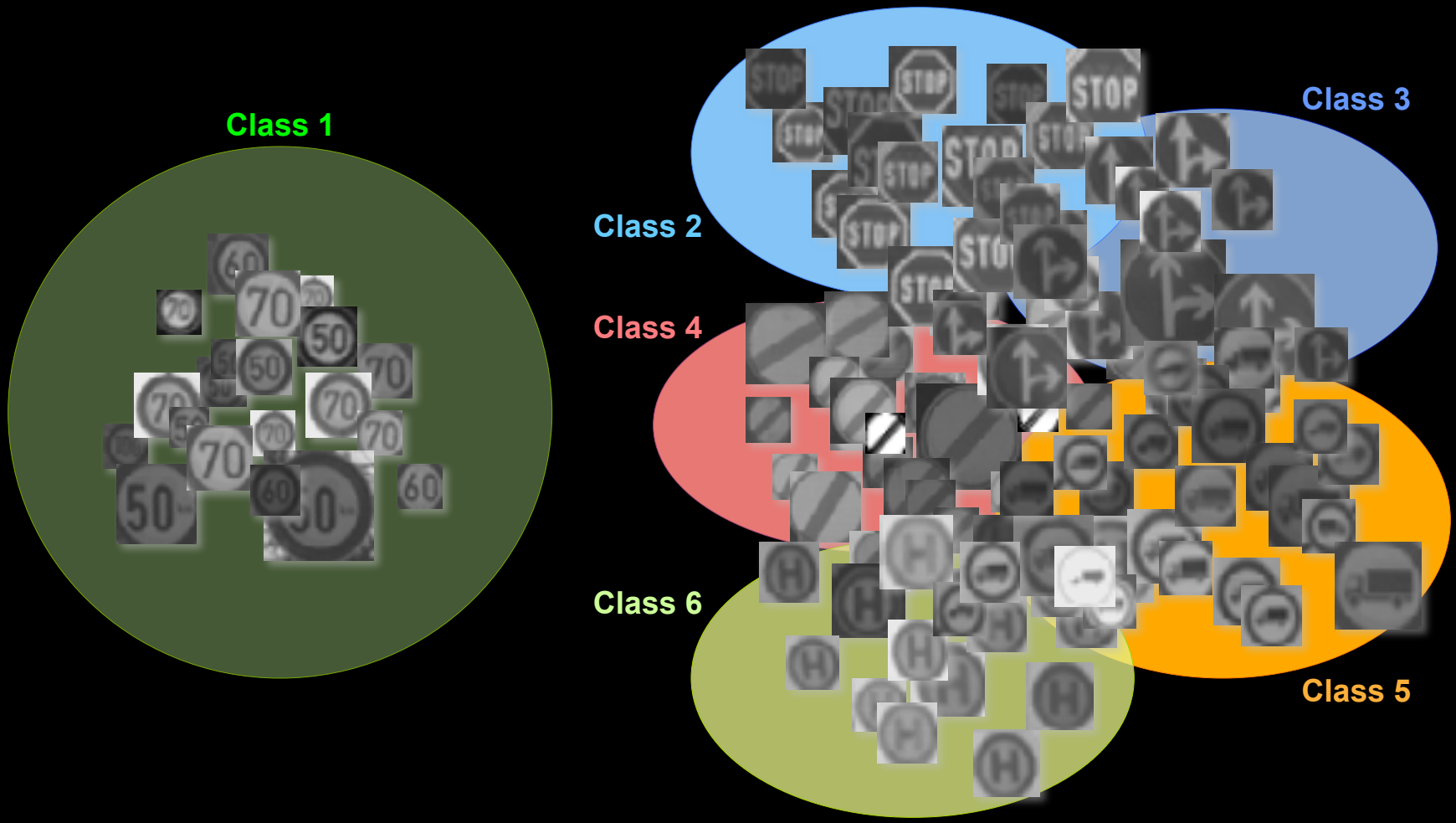


Background



Binary classification: two classes, “signal” and “background”

Multi-Class Classification



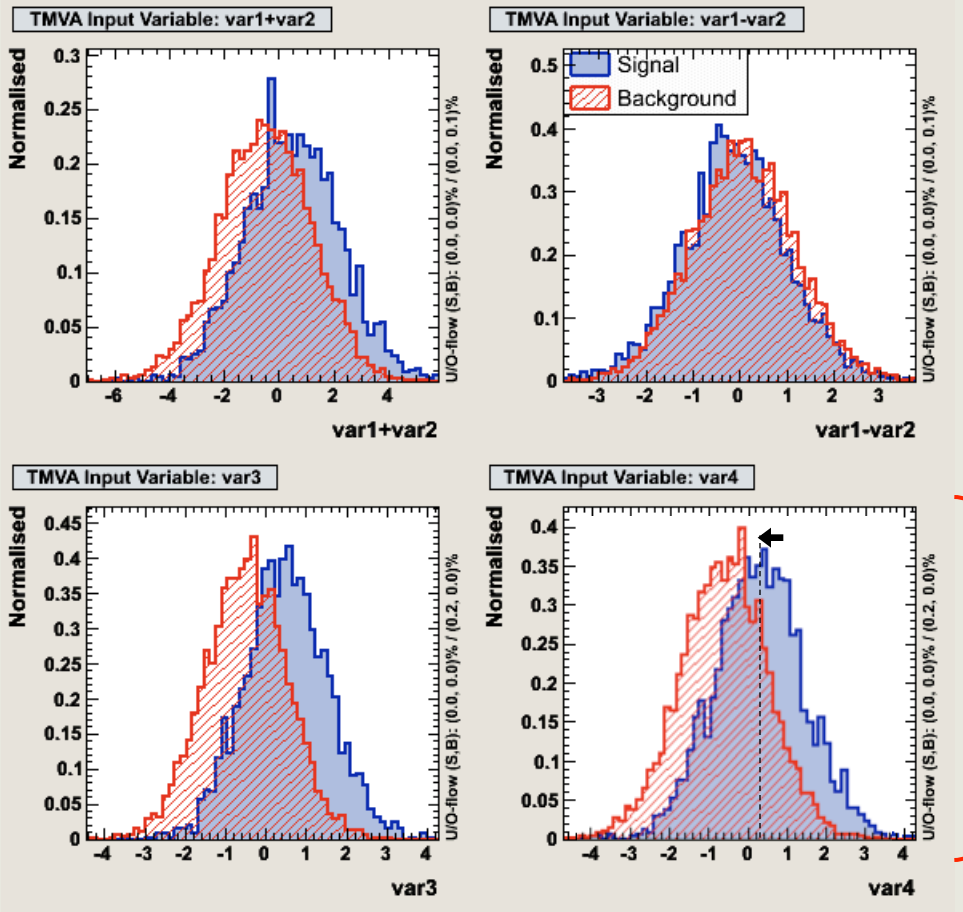
Multi-class classification – natural extension for many classifiers

A (brief) Word on

Systematics
&
Irrelevant Input Variables

Treatment of Systematic Uncertainties

- Assume strongest variable “var4” suffers from systematic uncertainty



“Calibration uncertainty” may shift the central value and hence worsen the discrimination power of “var4”

Treatment of Systematic Uncertainties

- Assume strongest variable “var4” suffers from systematic uncertainty
- (at least) Two ways to deal with it:

1. Ignore the systematic in the training, and evaluate systematic error on classifier output

– Drawbacks:

  - “var4” appears stronger in training than it might be → suboptimal performance
  - Classifier response will strongly depend on “var4”

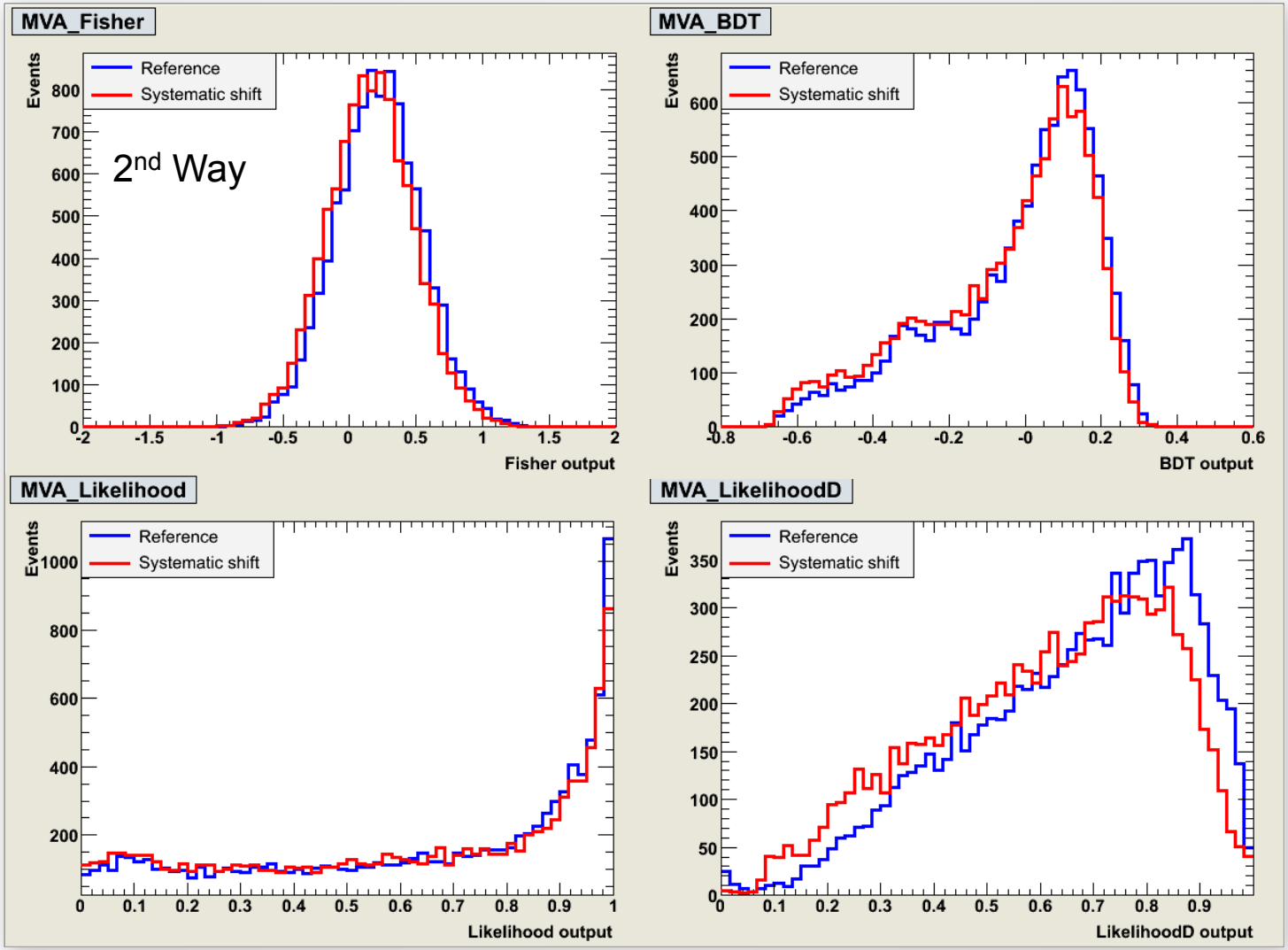
2. Train with shifted (= weakened) “var4”, and evaluate systematic error on classifier output

– Cures previous drawbacks

- If classifier output distributions can be validated with data control samples, the second drawback is mitigated, but not the first one (the performance loss) !

Treatment of Systematic Uncertainties

Classifier output distributions for signal only



Stability with Respect to Irrelevant Variables

- Toy example with 2 discriminating and 4 non-discriminating variables ?