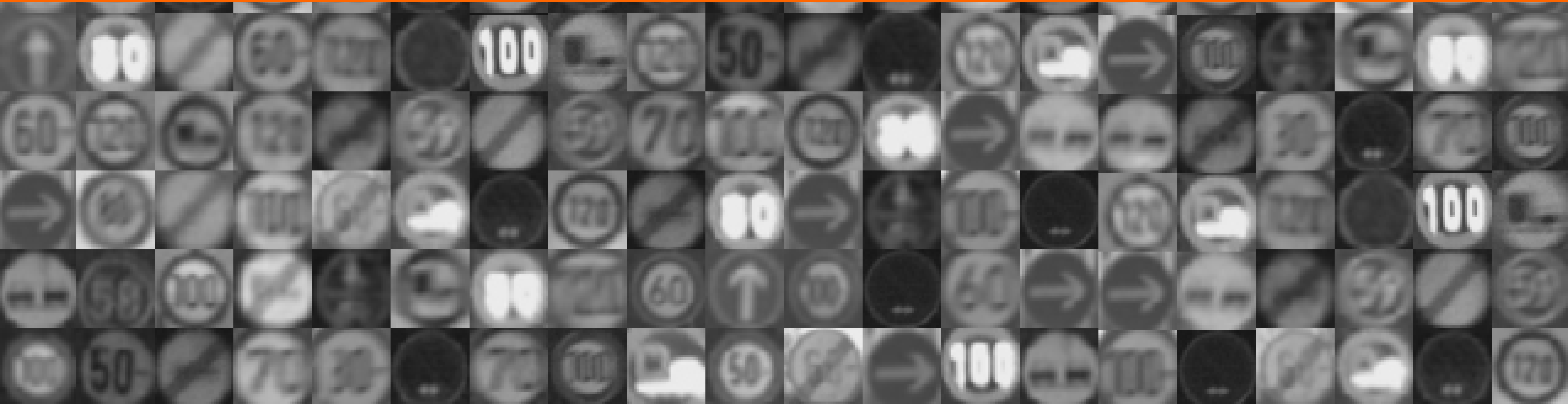
Multivariate Data Analysis and Machine Learning in High Energy Physics (V)

Helge Voss (MPI–K, Heidelberg)

Graduierten-Kolleg , Freiburg, 11.5-15.5, 2009



Outline

- last lecture
- Rule Fitting
- Support Vector Machines
- Some toy demonstations
- General considerations + Systematic uncertainties

Last Lecture

- (Boosted-) Decision trees
  - Decision Tree: sequential application of splits (cuts) → cut out an arbitrary volume formed out of little cubes in your parameter space
    - a test event is classified as either signal or background depending on which “cube” (leaf node) it falls into
  - “average” over many of those sets of such decision trees: → Boosting
    - create different trees using modifications of event weighs in training data
      - AdaBoost, ε-Boost, logit-Boost…
      - Bagging, Random Forest
  - the brute force method that often prooves very effective and robust although it has hundreds, thousands of “free” parameters.
    - simple construction rules → little chance to do things “wrong” (falling in local minima, be become confused by high dimensional problem with little data)

Learning with Rule Ensembles

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

- Following RuleFit approach by Friedman-Popescu
- Model is linear combination of *rules*, where a rule is a sequence of cuts

RuleFit classifier

rules (cut sequence → r_m=1 if all cuts satisfied, =0 otherwise)

normalised discriminating event variables

$$y_{\mathrm{RF}}(\vec{x}) = a_0 + \underbrace{\sum_{m=1}^{M_R} a_m r_m(\hat{\vec{x}})}_{\text{Sum of rules}} + \underbrace{\sum_{k=1}^{n_R} b_k \hat{x}_k}_{\text{Linear Fisher term}}$$

- 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 Machines

- If Neural Networks are complicated by finding the proper optimum “weights” for best separation power by “wiggly” functional behaviour of the piecewise defined separation hyperplane
- If KNN (multidimensional likelihood) suffers disadvantage that calculating the MVA-output of each test event involves evaluation of ALL training events
- If Boosted Decision Trees <u>in theory</u> are always weaker than a perfect Neural Network
- → Try to get the best of all worlds…

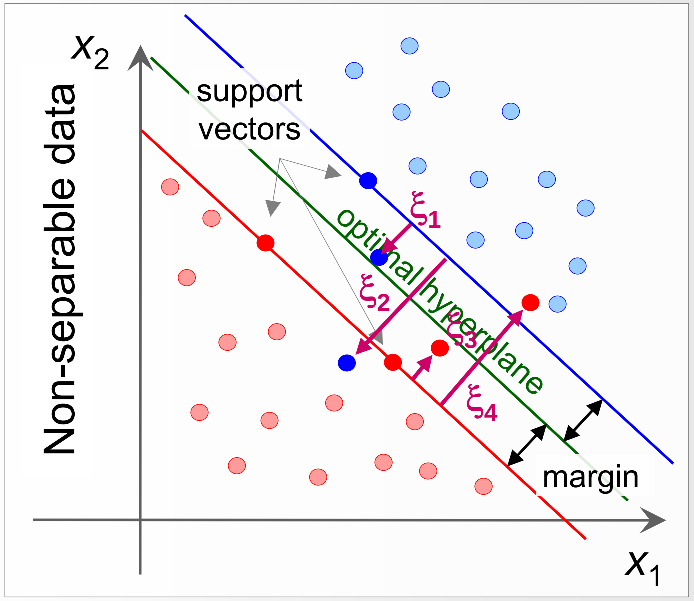
Support Vector Machine

- There are methods to create linear decision boundaries using only measures of distances
  - → leads to quadratic optimisation processs
- The decision boundary in the end is defined only by training events that are closest to the boundary
- We've seen that variable transformations, i.e moving into a higher dimensional space (i.e. using var1*var1 in Fisher Discriminant) can allow you to separate with linear decision boundaries non linear problems
- →Support Vector Machine

Support Vector Machines

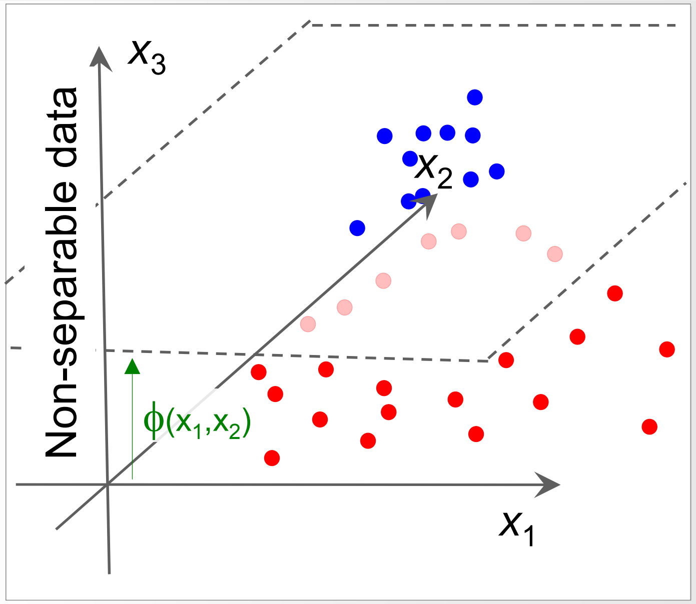
- Find hyperplane that best separates signal from background
  - Best separation: maximum distance (margin) between closest events (*support*) to hyperplane
  - Linear decision boundary
  - If data non-separable add *misclassification cost* parameter $C\cdot\Sigma_i\xi_i$ to minimisation function
  - **Solution of largest margin depends only on inner product of support vectors (distances)**

  → quadratic minimisation problem



Support Vector Machines

- 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 $C \cdot \Sigma_i \xi_i$ to minimisation function
  - **Solution depends only on inner product of support vectors → quadratic minimisation problem**



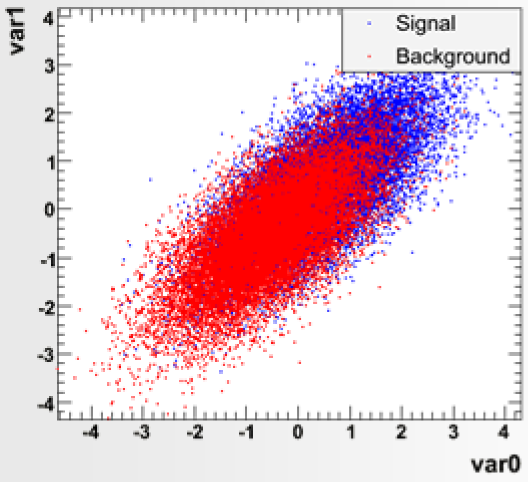
- Non-linear cases:
  - Transform variables into higher dimensional feature space where again a linear boundary (hyperplane) can separate the data
  - Explicit transformation doesn't need to be specified. Only need the "scalar product" (inner product) $x \cdot x \rightarrow \Phi(x) \cdot \Phi(x)$.
    - certain *Kernel Functions* can be interpreted as scalar products between transformed vectors in the higher dimensional feature space. e.g.: Gaussian, Polynomial, Sigmoid
  - Choose Kernel and fit the hyperplane using the linear techniques developed above
  - ➡ Kernel size paramter typically needs careful tuning! (Overtraining!)

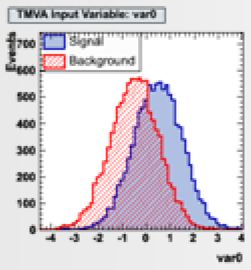
See some Classifiers at Work

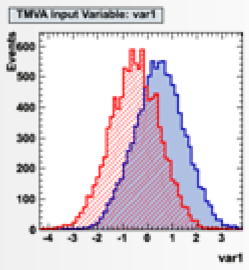
Toy Examples: Linear-, Cross-, Circular Correlations

- Illustrate the behaviour of linear and nonlinear classifiers

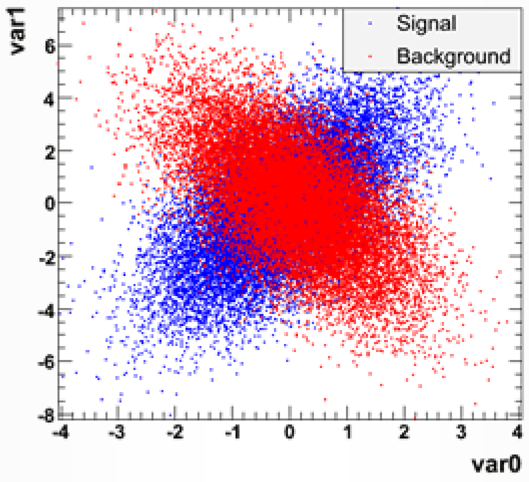
Linear correlations
(same for signal and background)

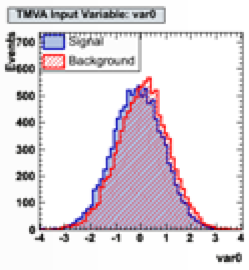


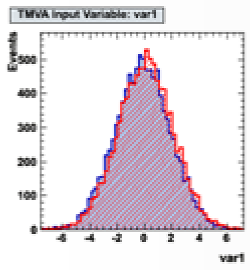




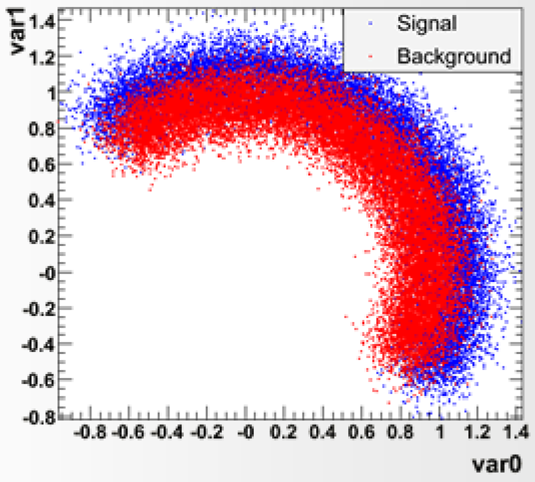
Linear correlations
(opposite for signal and background)

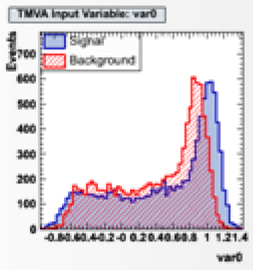


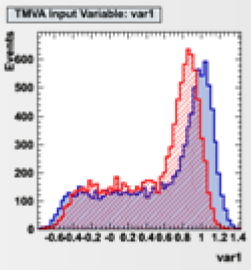




Circular correlations
(same for signal and background)



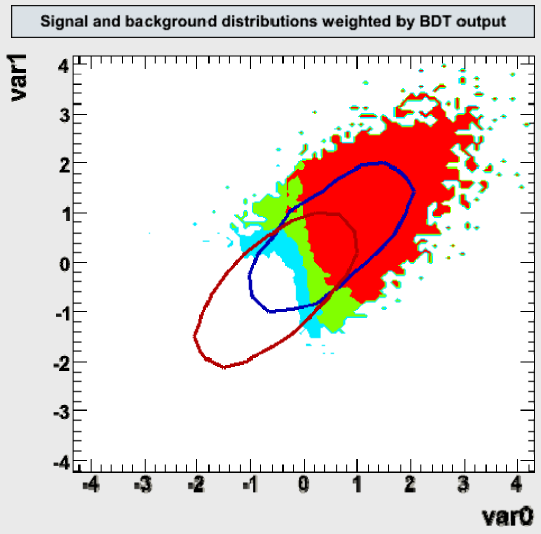




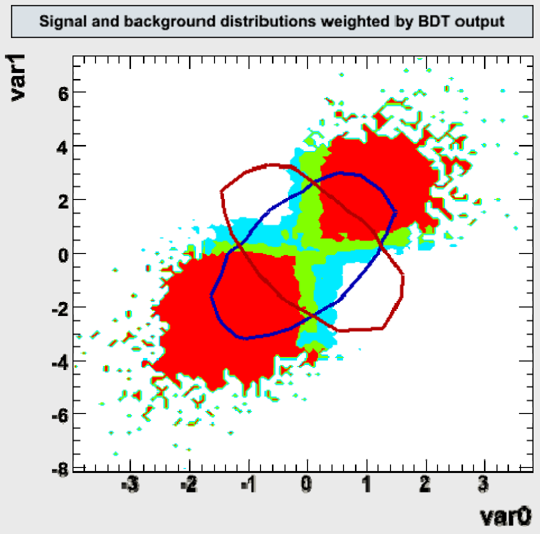
Weight Variables by Classifier Output

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

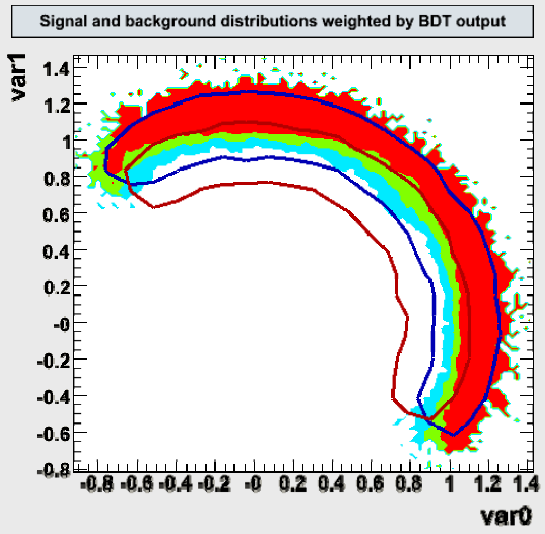
Linear correlations
(same for signal and background)



Cross-linear correlations
(opposite for signal and background)



Circular correlations
(same for signal and background)



BDT

General Advice for (MVA) Analyses

- There is no magic in MVA-Methods:
  - no need to be too afraid of “black boxes”
  - you typically still need to make careful tuning and do some “hard work”
- The most important thing at the start is finding good observables
  - good separation power between S and B
  - little correlations amongst each other
  - no correlation with the parameters you try to measure in your signal sample!
- Think also about possible combination of variables
  - can this eliminate correlation
- Always apply straightforward preselection cuts and let the MVA only do the rest.
- “Sharp features should be avoided” → numerical problems, loss of information when binning is applied
  - simple variable transformations (i.e. log(var1)) can often smooth out these areas and allow signal and background differences to appear in a clearer way

Some Words about Systematic Errors

- Typical worries are:
  - What happens if the estimated "Probability Density" is wrong ?
  - Can the Classifier, i.e. the discrimination function y(x), introduce systematic uncertainties?
  - What happens if the training data do not match "reality"

→Any wrong PDF leads to imperfect discrimination function

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

→Imperfect (calling it "wrong" isn't "right") y(x) → loss of discrimination power

that's all!

→classical cuts face exactly the same problem, however:

in addition to cutting on features that are not correct, now you can also "exploit" correlations that are in fact not correct

- Systematic error are only introduced once "Monte Carlo events" with imperfect modeling are used for
  - efficiency; purity
  - #expected events

- same problem with classical "cut" analysis
- use control samples to test MVA-output distribution (y(x))

- Combined variable (MVA-output, y(x)) might "hide" problems in ONE individual variable more than if looked at alone → train classifier with few variables only and compare with data

Treatment of Systematic Uncertainties

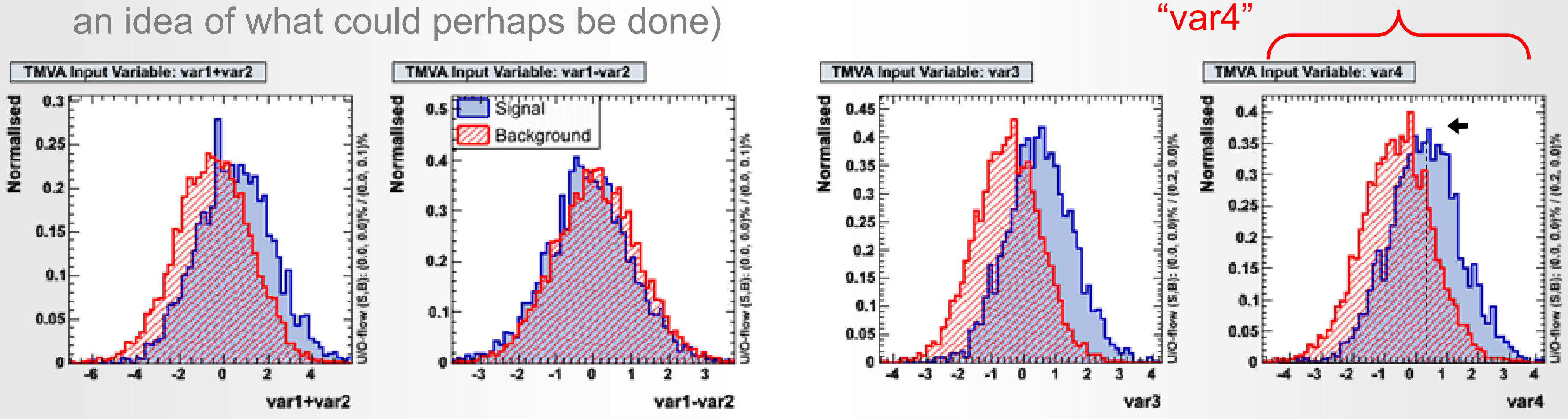
- Is there a strategy however to become 'less sensitive' to possible systematic uncertainties
  - i.e. classically: variable that is prone to uncertainties → do not cut in the region of steepest gradient
  - classically one would not choose the most important cut on an uncertain variable

- Try to make classifier less sensitive to "uncertain variables"
  - i.e. re-weight events in training to decrease separation

    in variables with large systematic uncertainty

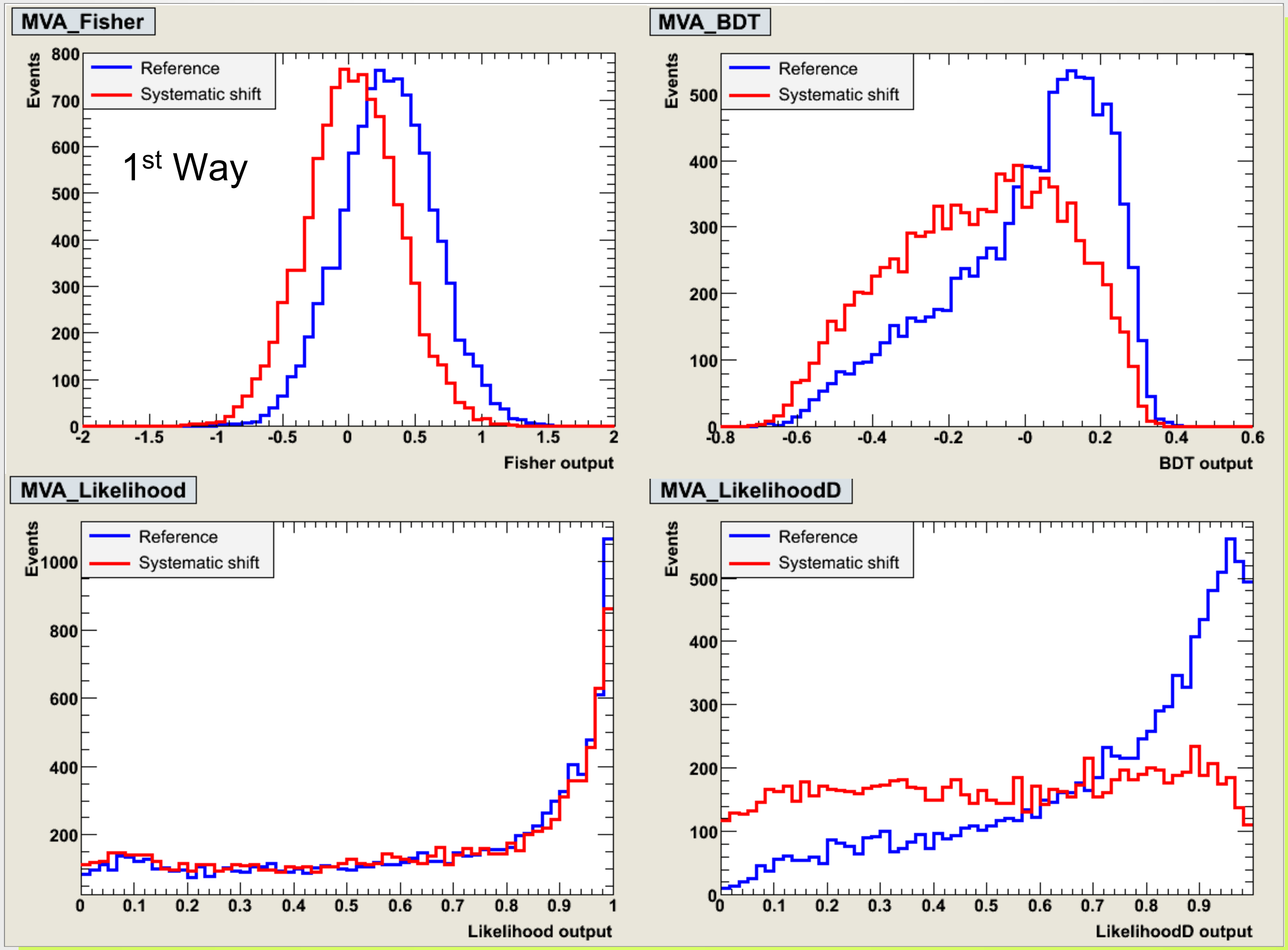
(certainly not yet a recipe that can strictly be followed, more an idea of what could perhaps be done)

"Calibration uncertainty" may shift the central value and hence worsen (or increase) the discrimination power of "var4"



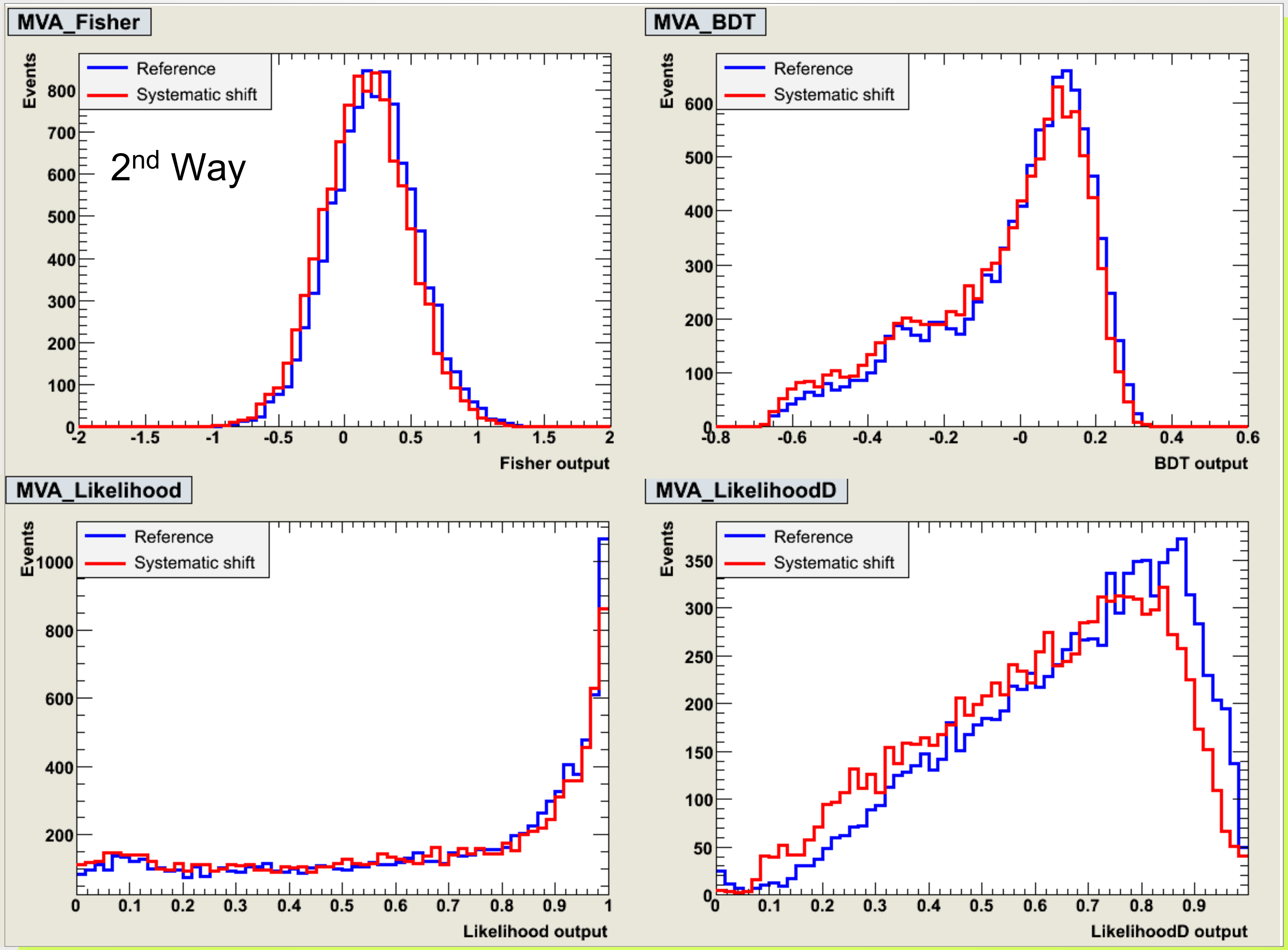
Treatment of Systematic Uncertainties

Classifier output distributions for signal only



Treatment of Systematic Uncertainties

Classifier output distributions for signal only



Summary of Classifiers and their Properties

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 A

Summary

- Traditional “cuts” are certainly not the most effective selection technique
- Multivariate Analysis → Combination of variables (taking into account variable correlations)
- Optimal event selection is based on the likelihood ratio
  - kNN and Kernal Methods are attempts to estimate the probability density in D-dimensions
  - “naïve Bayesian” or “projective Likelihood” ignores variable correlations
    - Use de-correlation pre-processing of the data if appropriate
- More Classifiers:
  - Fishers Linear discriminant → simple and robust
  - Neural networks → very powerful but difficult to train (multiple local minima)
  - Support Vector machines → one global minimum but needs careful tuning
  - Boosted Decision Trees → a “brute force method”
- Avoid overtraining
- Systematic errors
  - be as careful as with “cuts” and check against data