Advanced analysis methods: today and tomorrow

Marcin Wolter
Institute of Nuclear Physics PAN, Kraków

Tau Workshop
14-19 may 2012, Kraków

Why go multivariate?

In HEP we try to identify events that are both rare and obscured by the wide variety of backgrounds: “finding needles in a haystack”. The conventional selection of events by using cuts on individual variables can be far from optimal.

- **We want to be efficient:**

  - Use all available information (variables)

- **We are lazy** (or we do not know the functional dependence between variables and the desired output):

  - We want a system learning on examples

What do we use MVA for?

- Classification (select signal out of background)
- Function approximation (regression)
- Finding signal / background ratio.
- Probability density estimation (estimate the probability distribution)
- Variable selection (find most important variables)
- Many others...

Commonly used Multivariate Methods

- Random Grid Search
- Linear Discriminants
- Quadratic Discriminants
- Naive Bayes (Likelihood Discriminant)
- Kernel Density Estimation
- (Boosted/Bagged) Decision Trees
- Support Vector Machines
- Neural Networks
- Bayesian Neural Networks
- Genetic Algorithms

And many, many others......

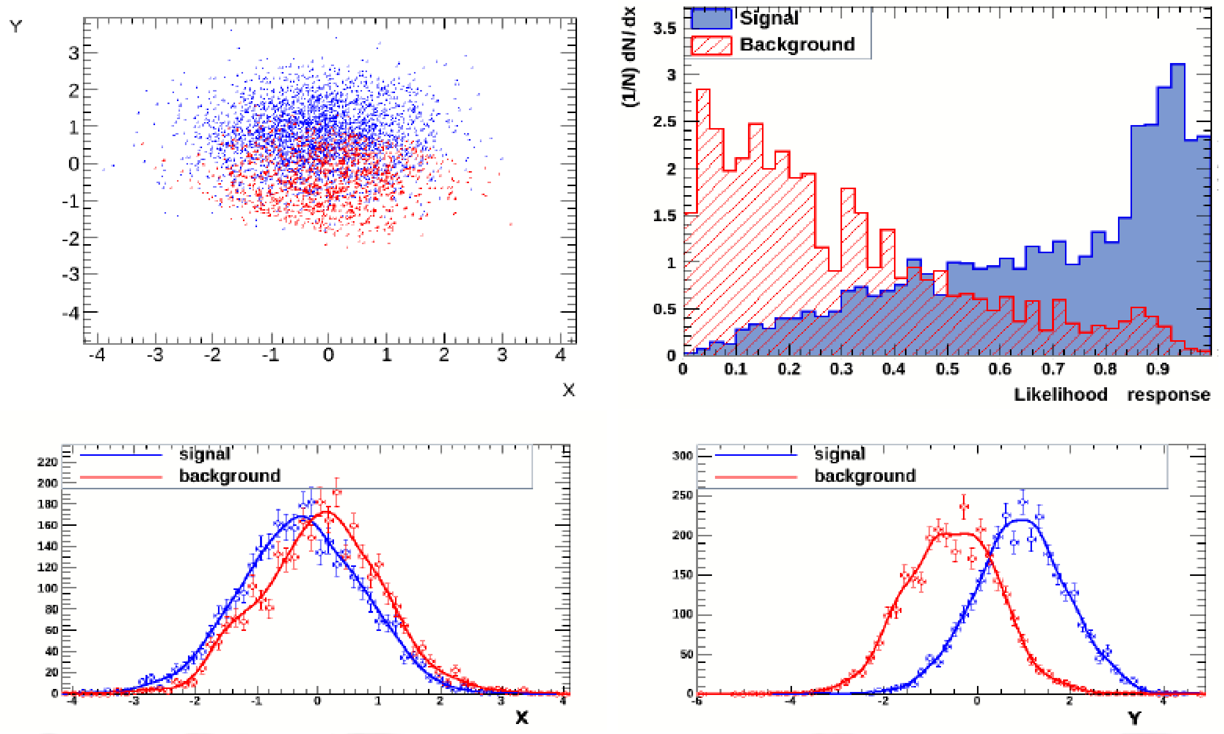
Why there are so many of them?

What do they have in common?

How much do they differ?

Do we need them all?

Naive Bayes – Projected Likelihood



We assume no correlation between variables:

$$P_i(S) = \frac{\mathcal{L}_i(S)}{\mathcal{L}_i(S) + \mathcal{L}_i(B)}$$

$$\mathcal{L}_i(S) = \prod_{k=1}^{N} p_k(S)(x_i)$$

$$\mathcal{L}_i(B) = \prod_{k=1}^{N} p_k(B)(x_i)$$

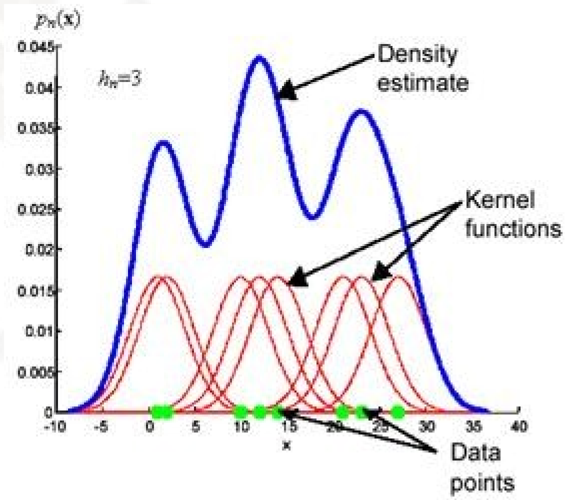
The method is simple, but in many cases very efficient (and fast). Used for tau identification at ATLAS experiment.

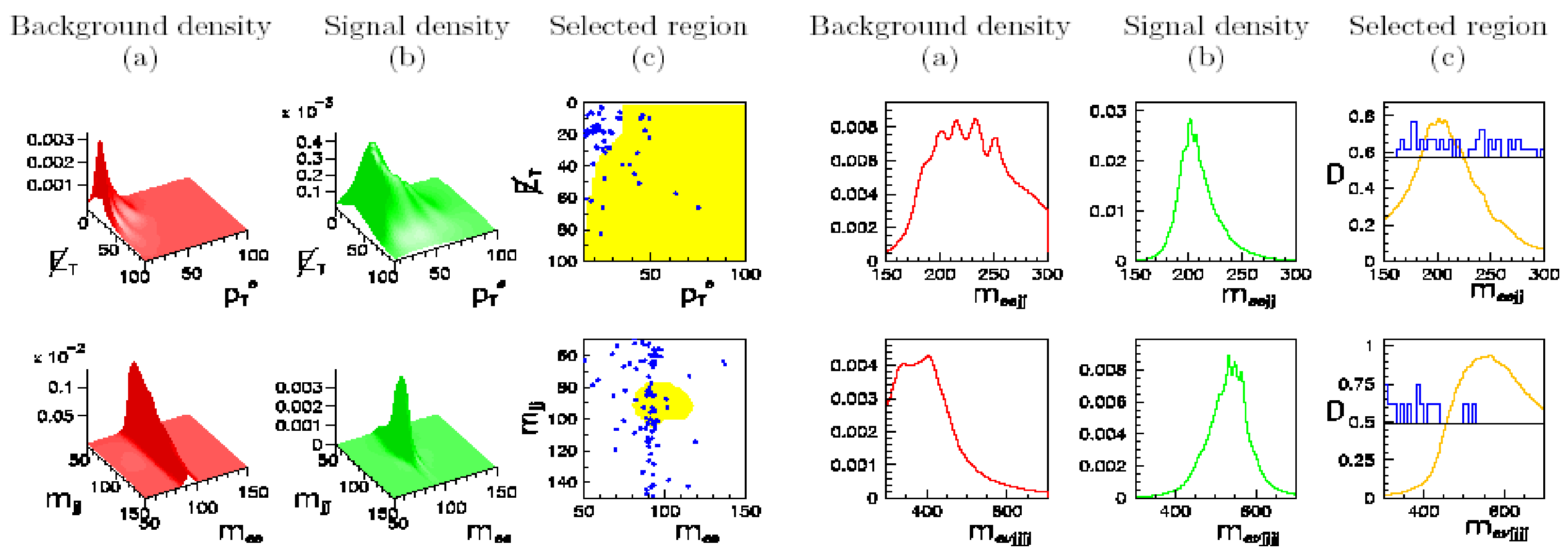
Probability Density Estimation

Approximation of the unknown probability as **a sum of kernel functions** placed at the points x_n of the training sample (Parzen 1960s).

Typical kernels: Gaussian, $1/x^n$, square etc.

Conceptually simple, no local minima, but CPU and memory demanding.



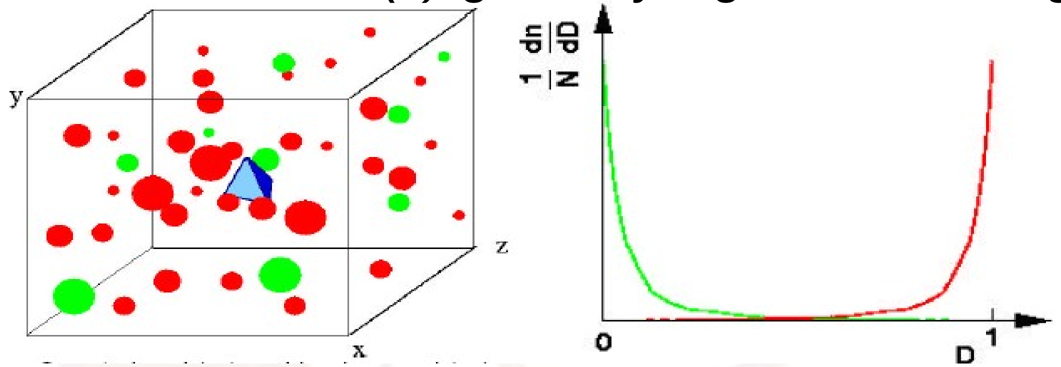


QUAERO package for D0 experiment.

How to speed it up? PDE_RS method

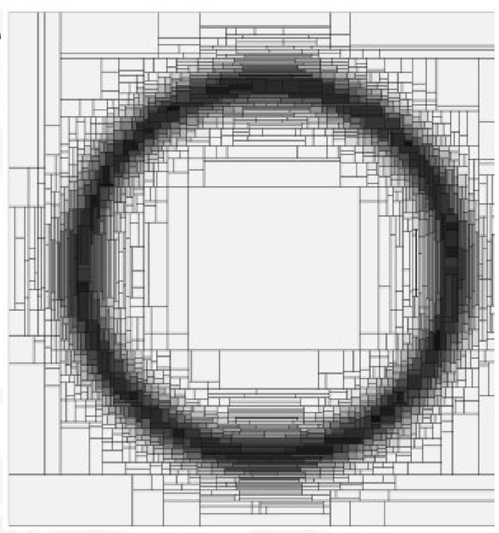
*method published by T. Carli, B. Koblitz, NIM A 501 (2003) 576-588, used by HERA

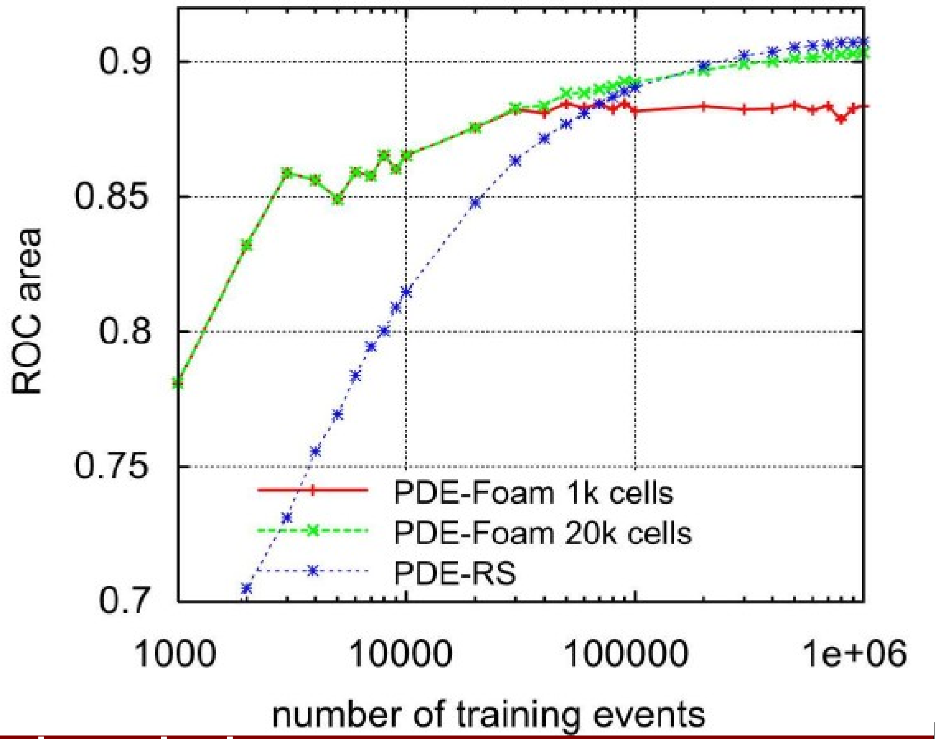
- Count signal (n_s) and background (n_b) events in N-dim hypercube around the event classified – only few events from the training sample needed.
- Hypercube dimensions are free parameters to be tuned.
- Discriminator *D(x)* given by signal and background event densities:



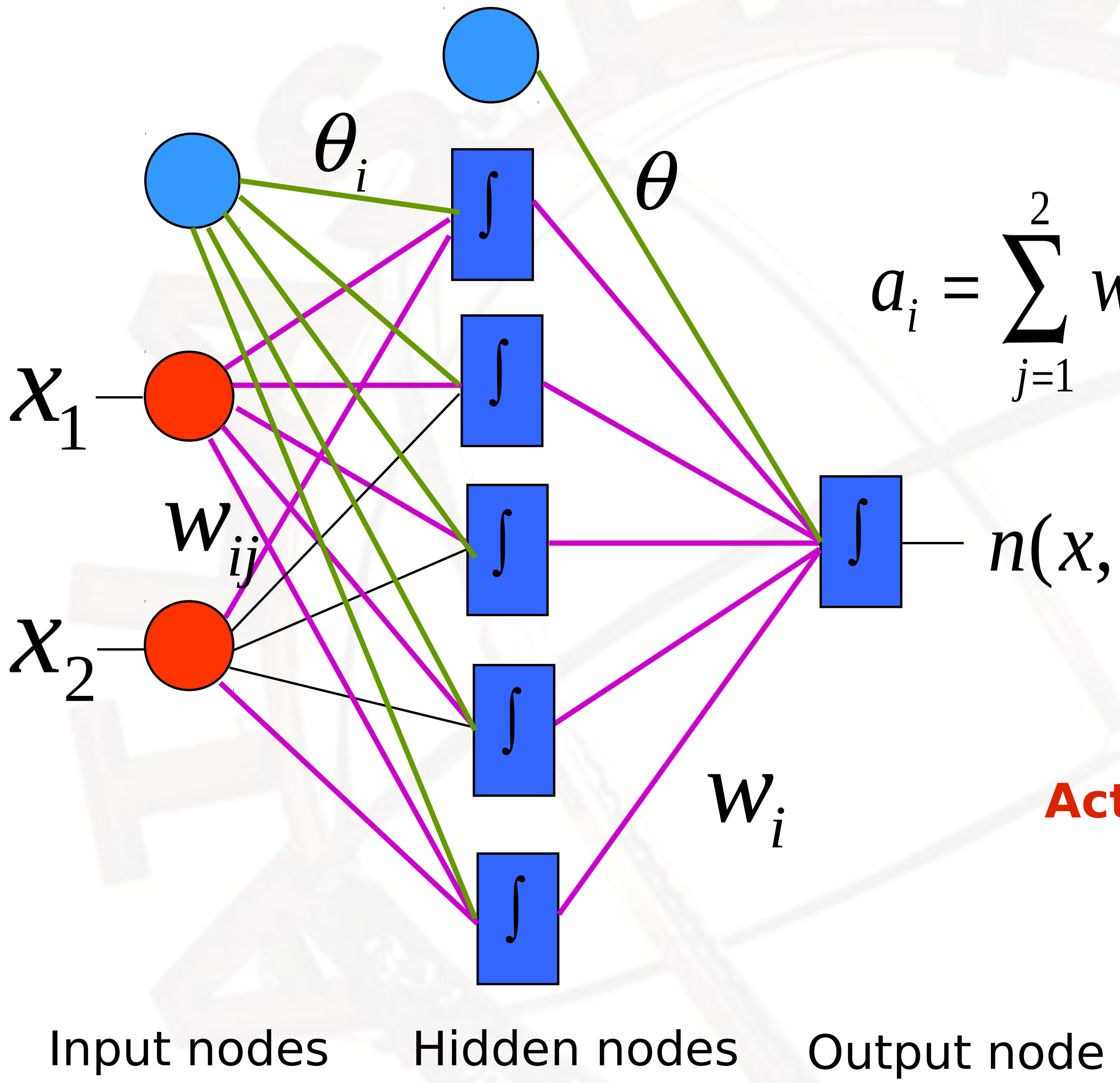
$$D(x) = \frac{n_S}{n_S + n_b}$$

- Improvement – subdivide space using the FOAM algorithm by S. Jadach:





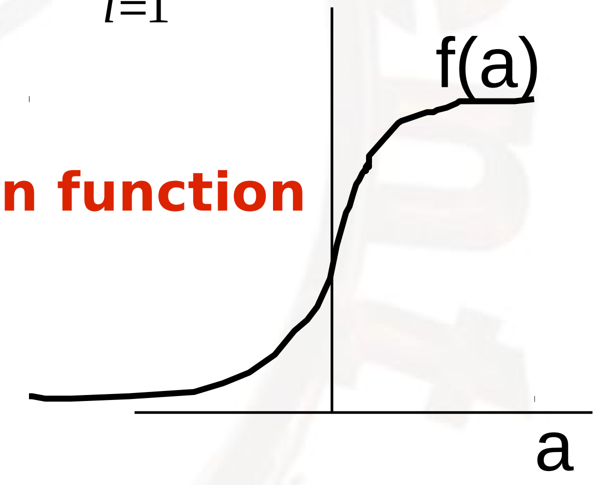
Neural Network



$$a_i = \sum_{j=1}^{2} w_{ij} x_j + \theta_i \quad \rightarrow \quad f(a_i)$$

$$n(x,w) = f\left(\sum_{i=1}^{5} w_i f(a_i) + \theta\right)$$

Activation function



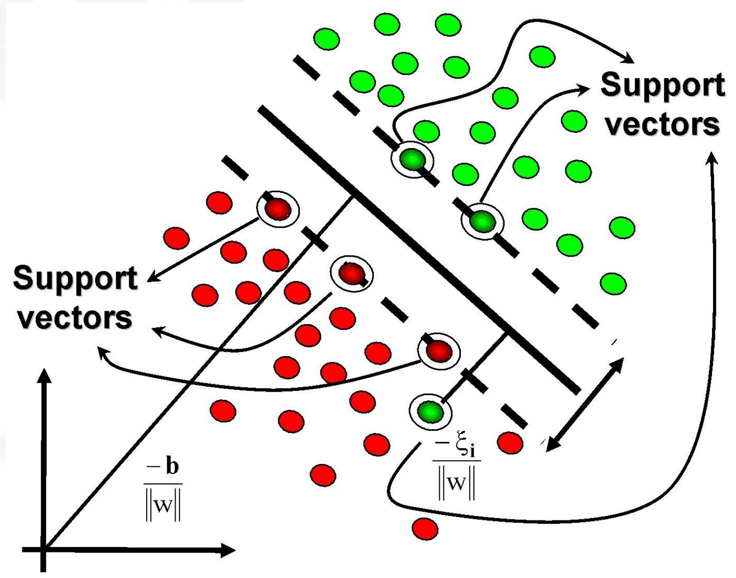
Support Vector Machine

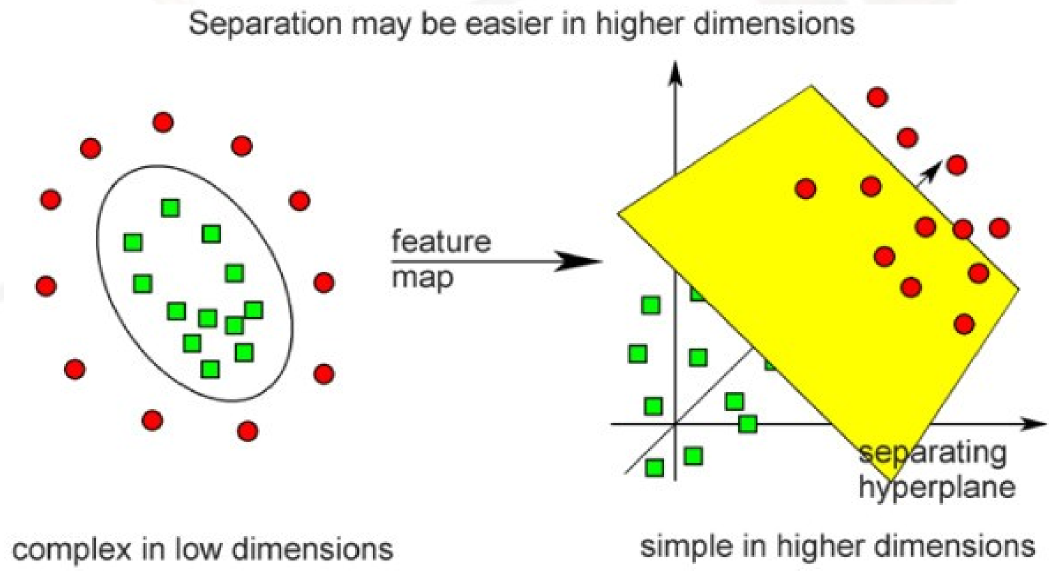
Basic idea:

- build a separating hyperplane using the minimal number of vectors from the training sample **(Support Vectors).**
- For non-linear case go to higher dimension feature space, where events are linearly separable - kernel trick.
- should be able to model any arbitrary function.

Nice mathematics, limited number of parameters, quadratic minimization.

But relatively slow and memory consuming.

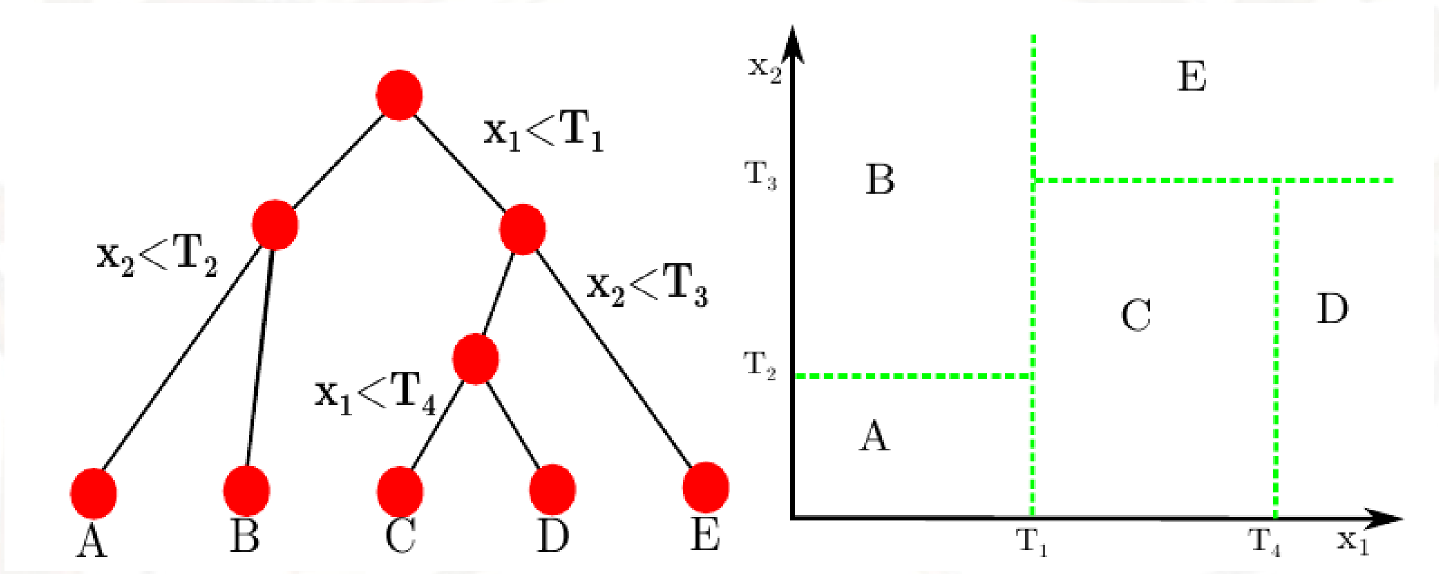




Separation may be easier in higher dimensions
feature map
separating hyperplane
complex in low dimensions
simple in higher dimensions

Decision trees and ensemble learning

Decision tree – a sequence of cuts, each leaf has an associated class (“signal” or “background”)



Simple and very fast, but sensitive for fluctuations and unstable.

Solution:

Ensemble learning – boosting, bagging, random forests...

These procedures can be applied to other algorithms, not only trees!

Boosting - AdaBoost

Problem: can we improve a weak classifier?

Answer: yes, applying it many times.

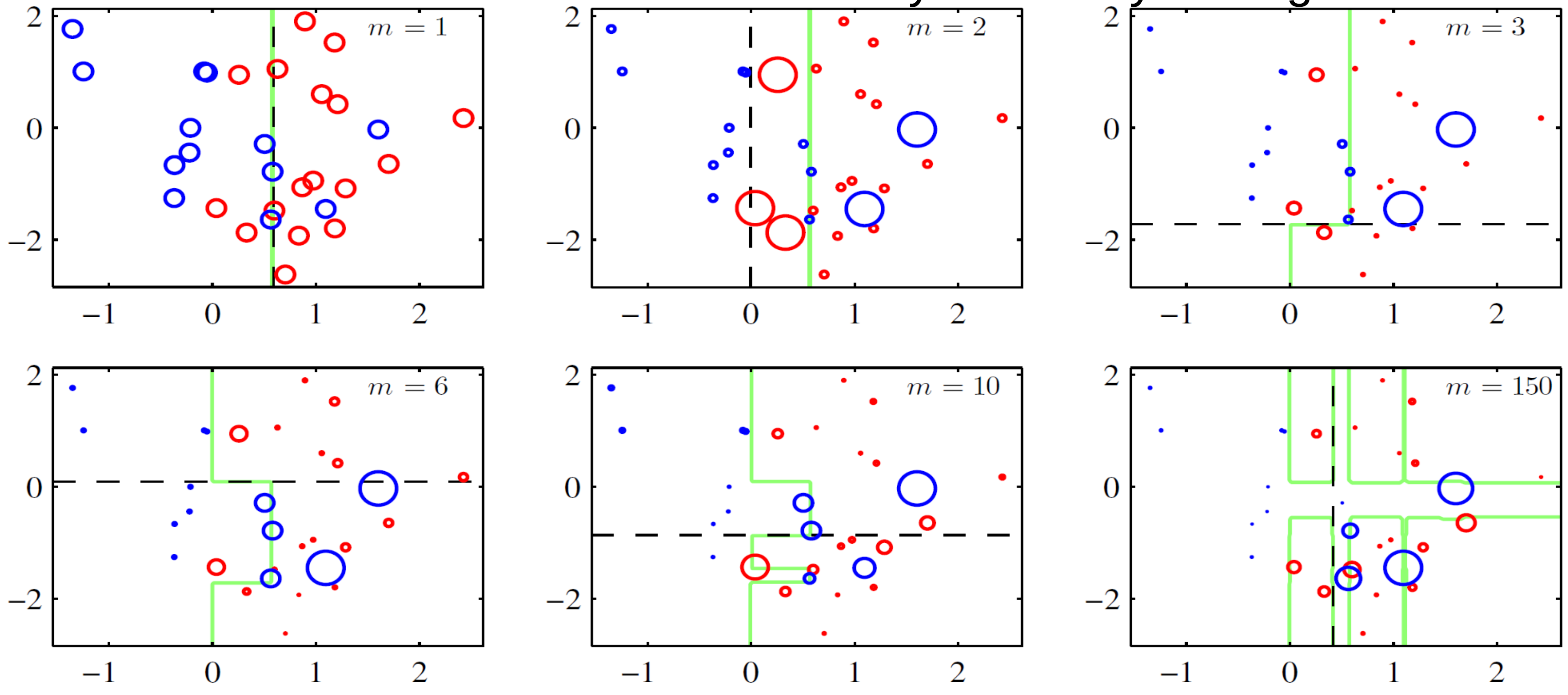
Most frequently used algorithm: **AdaBoost** (Freund & Schapire 1996 – Gödel prize)

Build a decision tree.

Increase the weights of wrongly classified events.

Repeat many times – get many trees.

Classify events by “voting” of all trees.



Boosting, bagging

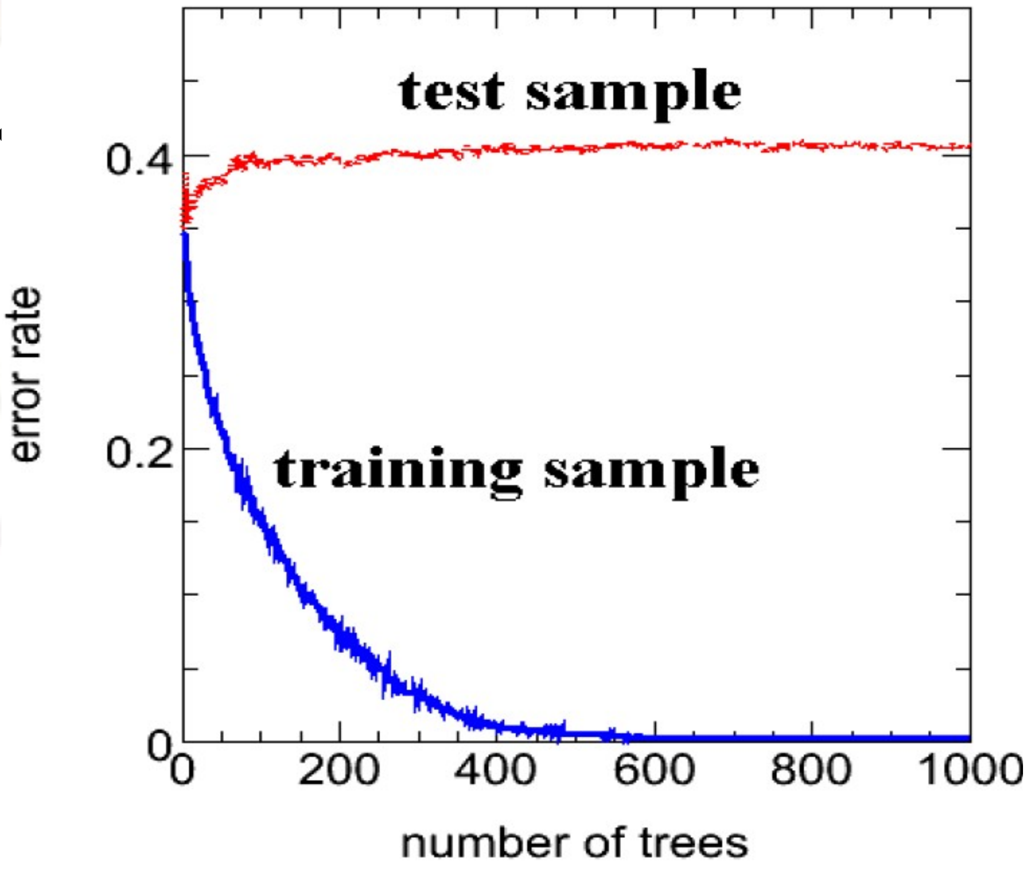
Boosting, bagging – in a “magic way” we get a strong classifier out of a weak one.

Commonly used to improve decision trees.

Good results without fine tuning of parameters - „the best out-of-box classification algorithm”.

<u>Resistant for overtraining.</u>

Became very popular in HEP. Or just trendy...



Results obtained with AdaBoost used to separate simulated mSUGRA out of tt events at LHC energies.

Bayesian learning - Bayesian probability

Bayesian probability:

degree of belief (Bayes, Laplace, Gauss, Jeffreys, de Finetti).

Frequentist probability:

relative frequency (Venn, Fisher, Neyman, von Mises).

Bayesian approach:

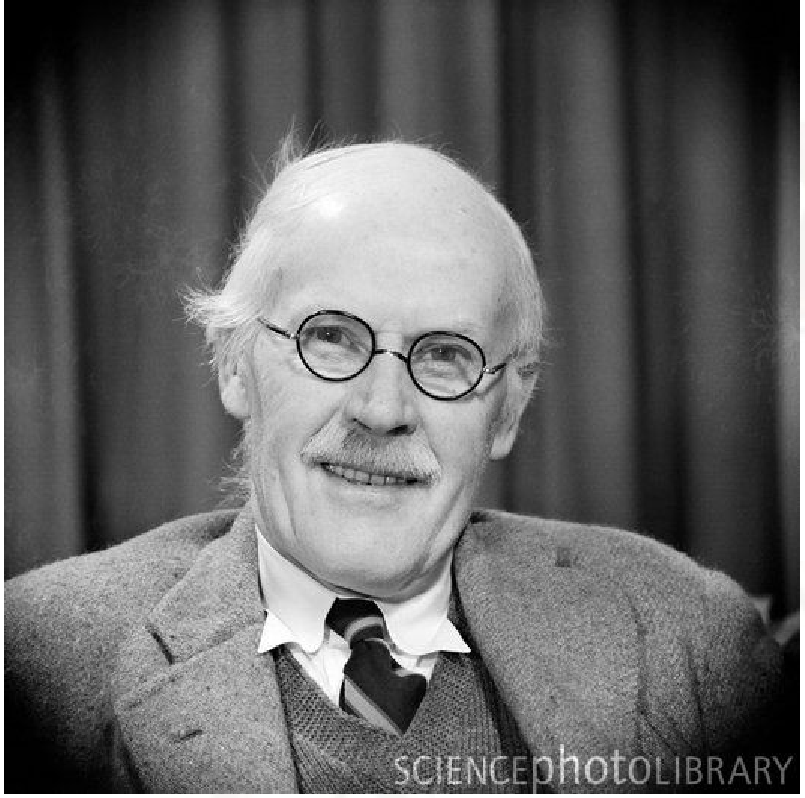
The probability ***p*** is our assessment of the probability of success at each trial, based on our current state of knowledge.

If our assessment, initially, is incorrect? As our state of knowledge changes, our assessment of the probability of success changes accordingly.

Rev. Thomas Bayes
(1702 - 1761)

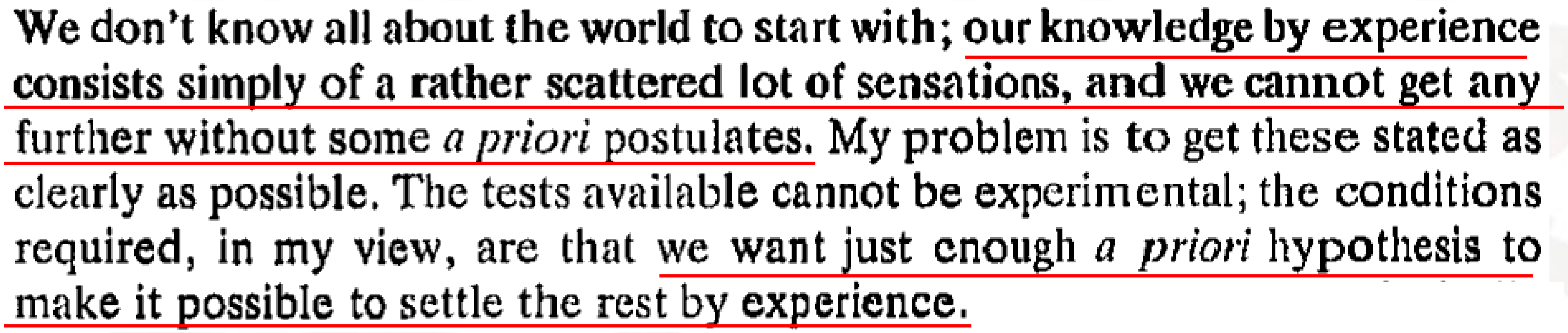
The probability of any event is the ratio between the value at which an expectation depending on the happening of the event ought to be computed, and the value of the thing expected upon its happening.

T. Bayes.



Sir Harold Jeffreys, (1891–1989) was a mathematician, statistician, geophysicist, and astronomer. His book "Theory of Probability", played an important role in the revival of the Bayesian view of probability

Sir Ronald Aylmer Fisher (1890 - 1962) was an English statistician, evolutionary biologist, eugenicist and geneticist.



We don't know all about the world to start with; our knowledge by experience consists simply of a rather scattered lot of sensations, and we cannot get any further without some *a priori* postulates. My problem is to get these stated as clearly as possible. The tests available cannot be experimental; the conditions required, in my view, are that we want just enough *a priori* hypothesis to make it possible to settle the rest by experience.

Sir Harold Jeffreys, in a letter to Sir Ronald Fisher dated 1 March, 1934

Machine and Bayesian Learning

Machine Learning

Teach a machine a dependence $y = f(x)$ by feeding it with **training data** ***T*** = (***x***, ***y***) = $(x,y)_1, (x,y)_2, \ldots (x,y)_N$ and a **constraint** on the class of functions.

Bayesian Learning

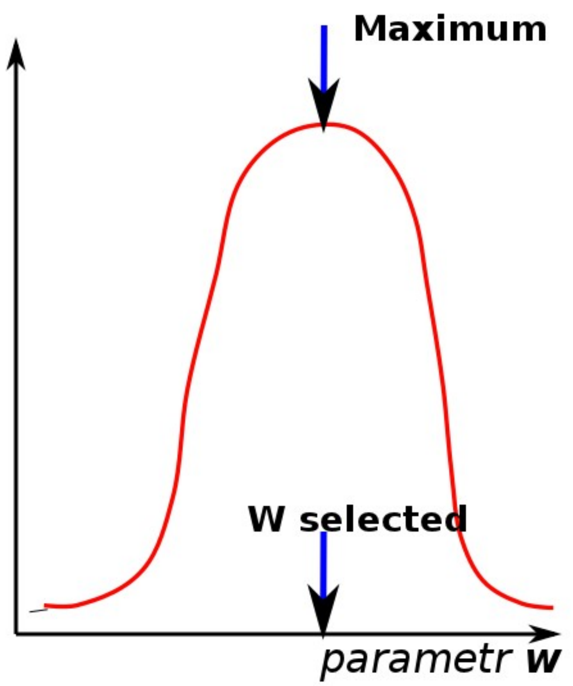
For each function f(x) in the function space F calculate the posterior probability *p*(***f*** |***T***) using a given training sample T= (***x***, ***y***).

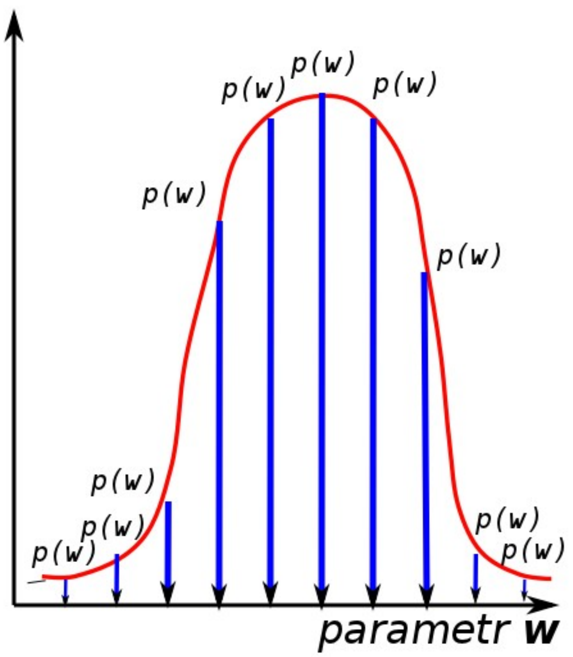
In bayesian learning we DO NOT use a single function, we use a bunch of functions weighted by their probabilities

The posterior probability is the conditional probability that is assigned after the relevant evidence is taken into account.

Training sample: ***T*** = (***x***, ***y***) set of input vectors ***x*** and desired outputs ***y.***

Machine and Bayesian learning





Machine learning

Just one function (parameter) selected

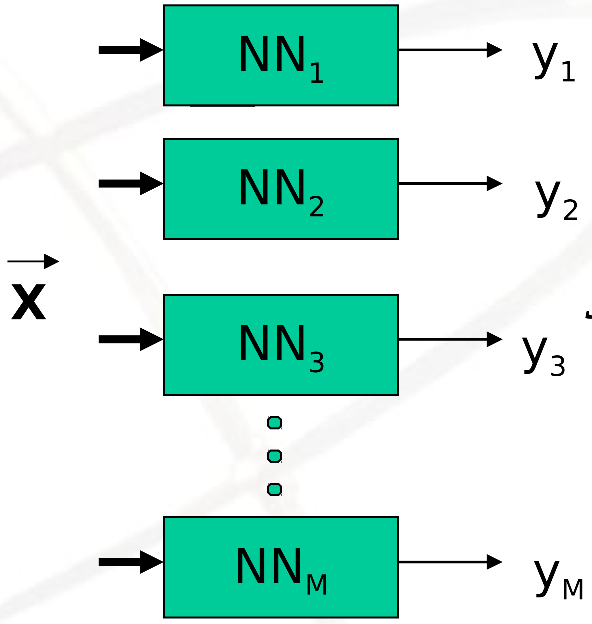
Bayesian learning

Each function (parameter) gets its probability / weight.

Practical implementation: Bayesian Neural Networks

Instead of choosing a single set of weights in NN – find posterior probability density over all possible weights (and many network architectures).

Average over many networks weighted by the probability of each network given the training data



$$\bar{y} = \sum_i a_i\, y_i(x)$$

Well described in C.M. Bishop "Neural Networks for Pattern Recognition", Oxford 1995

Free software (used by D0 Collaboration):
Radford Neal, http://www.cs.toronto.edu/~radford/fbm.software.html

Bayesian Neural Network

Each network is bdescribed by a vector of parameters **w.**

For training data T= {y,x}, the probability density at **w** is given by:

$$p(w,T)=\frac{p(T,w)\,p(w)}{p(T)}$$

Bayes theorem

p(w) – prior probability, must be chosen beforehand

The answer is a weighted average over all networks (values of **w**)**:**

$$\bar{y}(x)=\int f(x,w)\,p(w,T)\,dw \qquad f(x,w)-\mathit{neural\ network}$$

Calculation of integral: sampling using Markov Chain Monte Carlo.

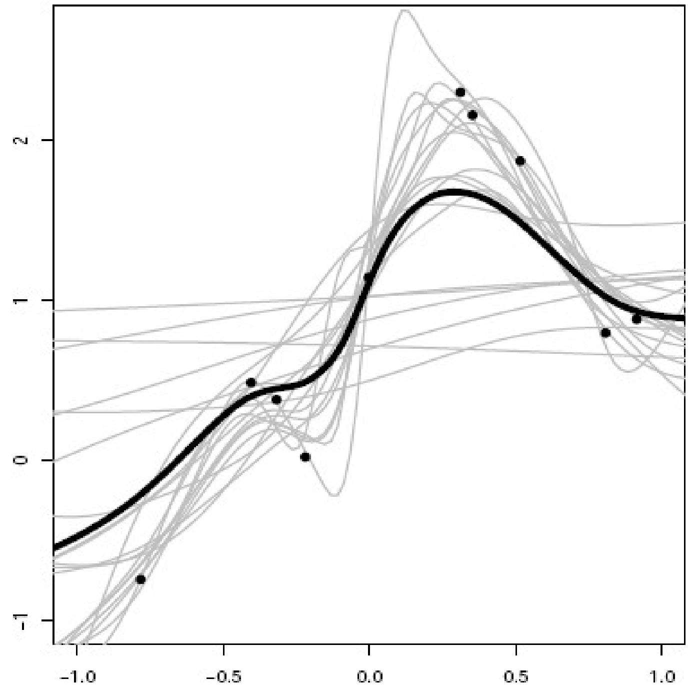
Advantages:

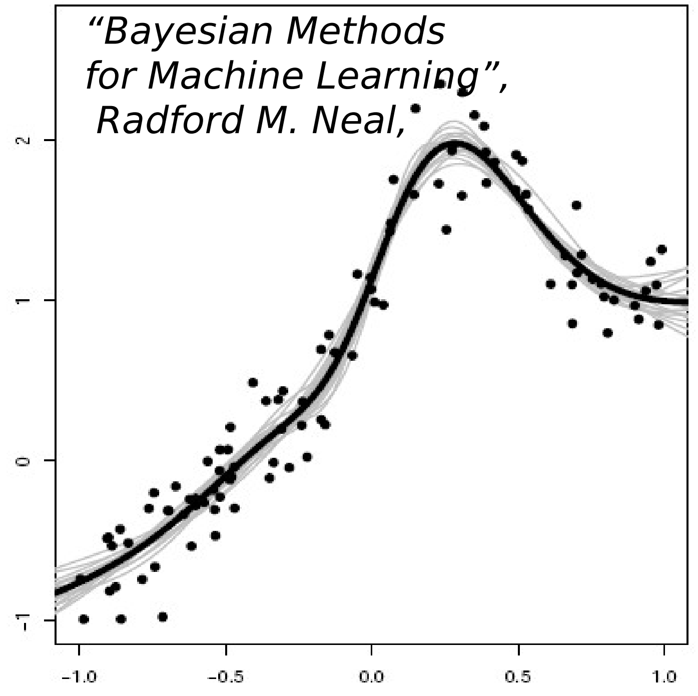
- We get an error of estimated function
- Less prone for overtraining and fluctuations.

Bayesian Neural Network

How the complex network with 100 hidden units performs with little data.

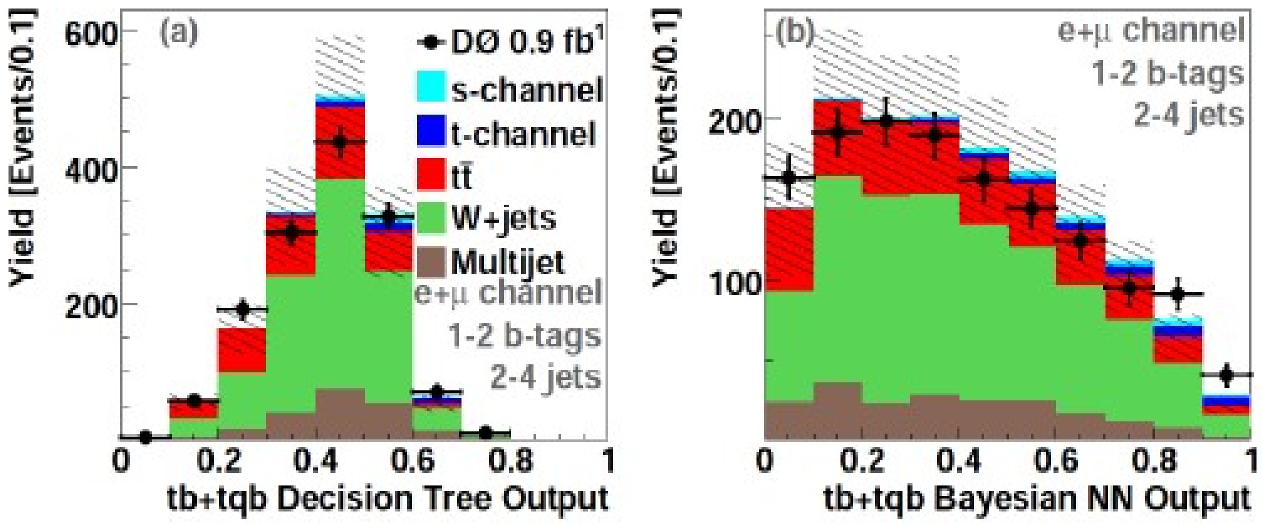
The data is from the function $x^3 + 2\exp(-6(x-0.3)^2)$, with Gaussian noise with standard deviation 0.2 added. The plots show show 20 functions from the posterior distributions given 10 data points (left) and 100 data points (right):



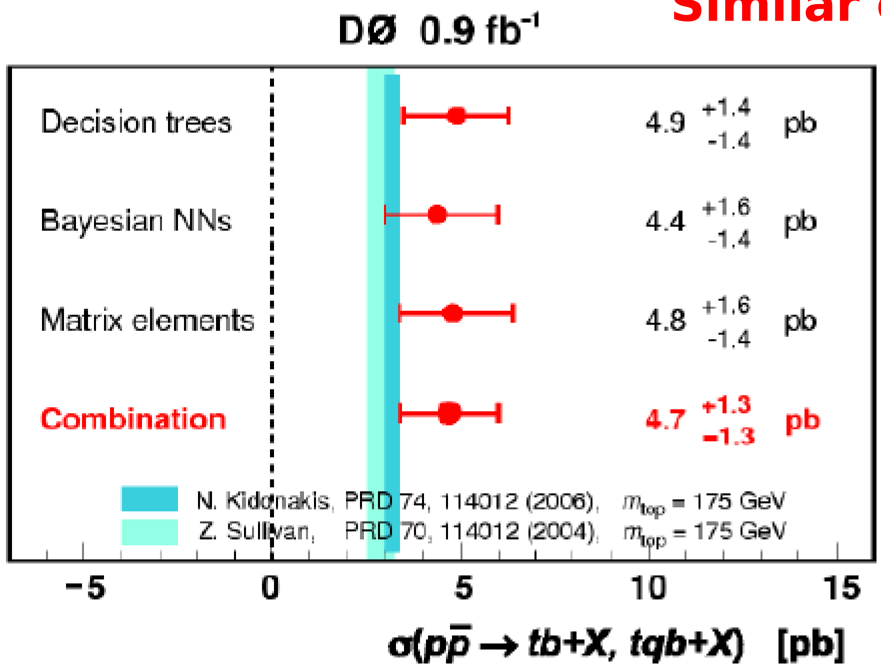


The bold line is the average of these functions, the Bayesian prediction for minimizing squared error, which is smoother than some individual functions.

Example – Single Top Search



Similar errors



Single top quark search using
boosted decision trees
Bayesian neural networks

D0 Collaboration,
PRD 78 012005, 2008

Do we always need the most advanced multivariate algorithm?

A Bayes classifier:

$$p(S|x)=\frac{p(x|S)\,p(S)}{p(x|S)\,p(S)+p(x|B)\,p(B)}$$

where ***S*** is associated with *y* = **1** and ***B*** with *y* = **0**. **Bayes classifier** *accepts events x if p(**S**|x) **> cut** as belonging to **S***.

Our method needs to approximate probability distributions *P(**x**|**S**)* and *P(**x**|**B**)*.

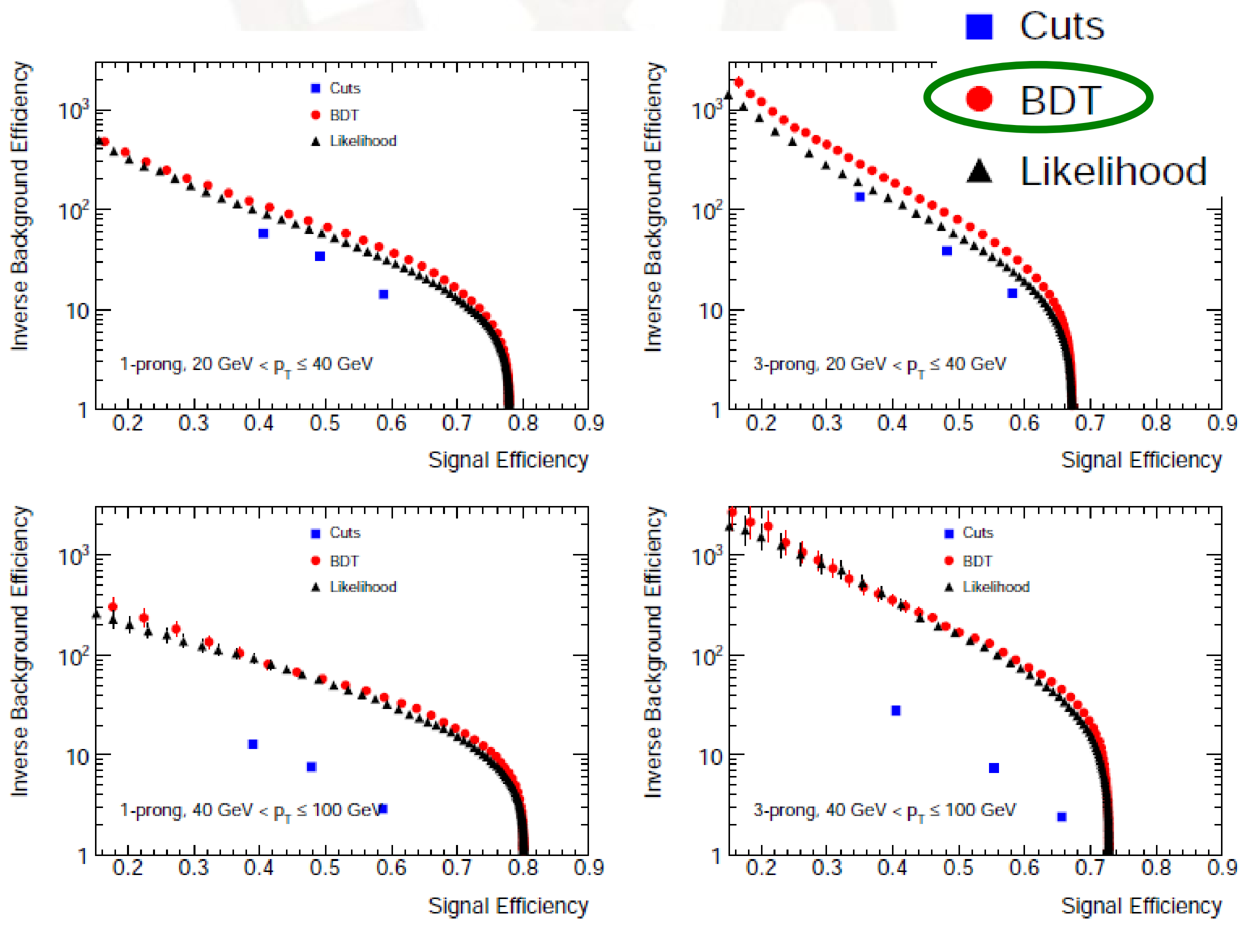
*If your goal is to **classify objects** with the fewest errors, then the **Bayes classifier** is the **optimal** solution.*

Consequently, if you have a classifier known to be ***close*** to the **Bayes limit**, then *any* other classifier, *however sophisticated*, can ***at best*** be only marginally better than the one you have.

- =>If your problem is **linear** you don't gain anything by using more sophisticated **Neural Network or BDT**

***All* classification methods, such as the ones in TMVA, are different numerical approximations of the Bayes classifier.**

Tau identification at ATLAS



See the talk of Stan: multivariate approach improves significantly tau identification.

In the past we used also: Neural Network, PDE_RS, SVM. They were also giving good results, but they are currently abandoned due to the lack of manpower.

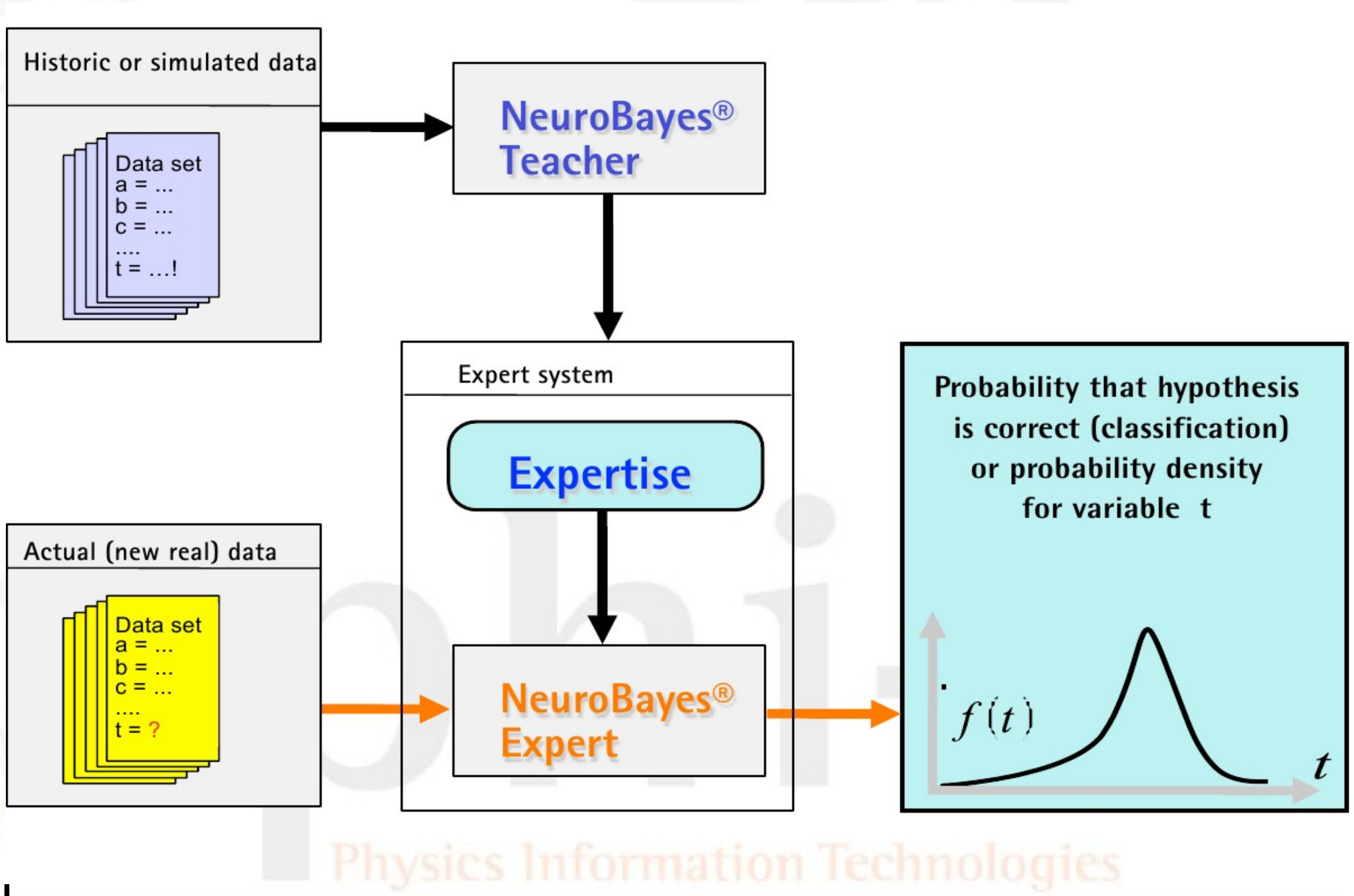
NeuroBayes package used by BELLE experiment

Commercial package – special rates for physics.

Neural Network + developed preprocessing tools.

Analysis tool: neural network combined with bayesian statistics -> well performing, fast and overtraining-save algorithm for data analysis (even boosting included).

TMVA/root interface.

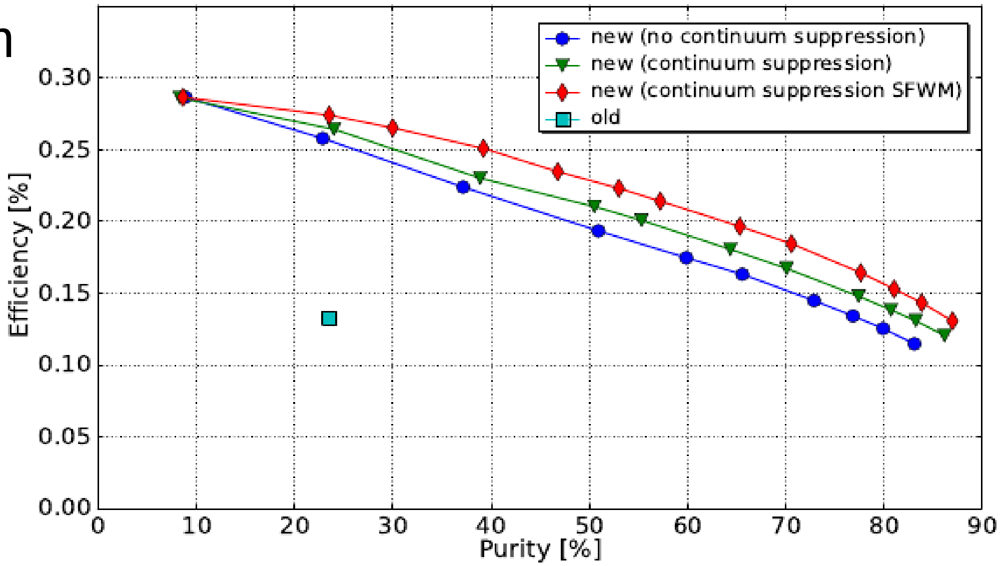


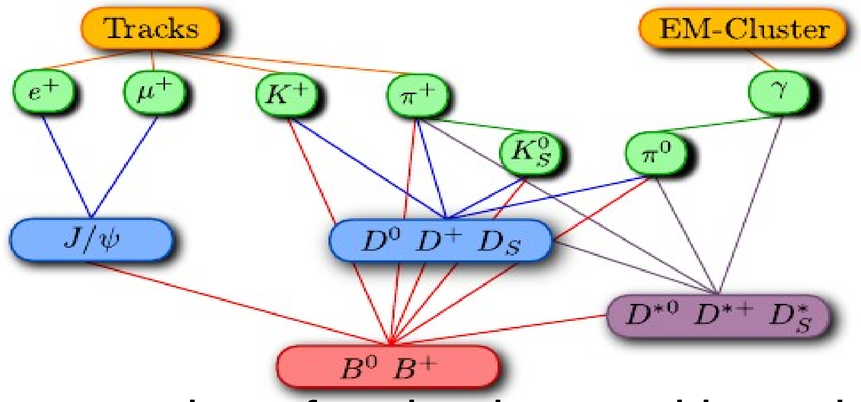
This is not a Bayesian Neural Network.

Belle analyses – two examples

•A Hierarchical NeuroBayes-based Algorithm for Full Reconstruction of B Mesons

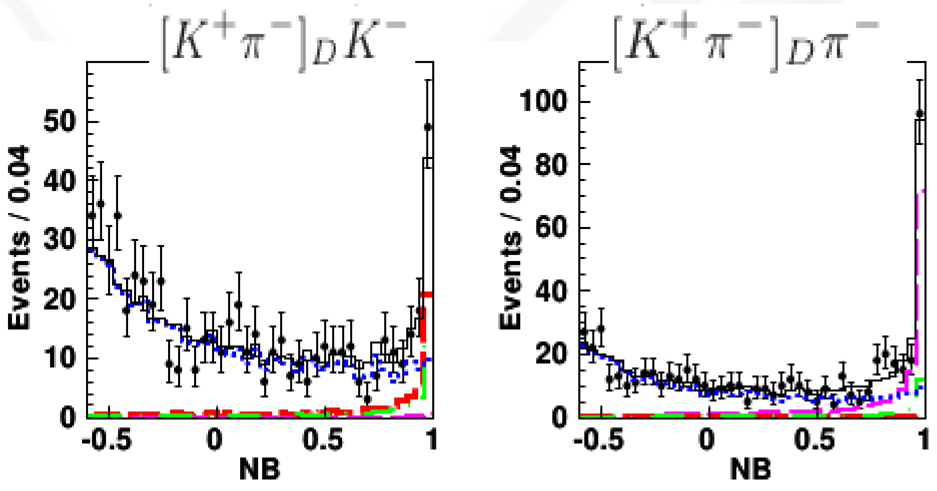
Hierarchical structure and multivariate classification – the user can choose the desired purity or efficiency.





•Suppression of qq background based on NeuroBayes (in analysis of the Suppressed Decay $B^- \to D K^-$, $D \to K^+\pi^-$).

Multivariate analysis with six inputs, 96% of signal kept, 74% of background rejected.



NN output, blue is qq background, red and magenta are the signal components.

$_s$Plot – training MVA on data?

(implemented in NeuroBayes, used by Belle)

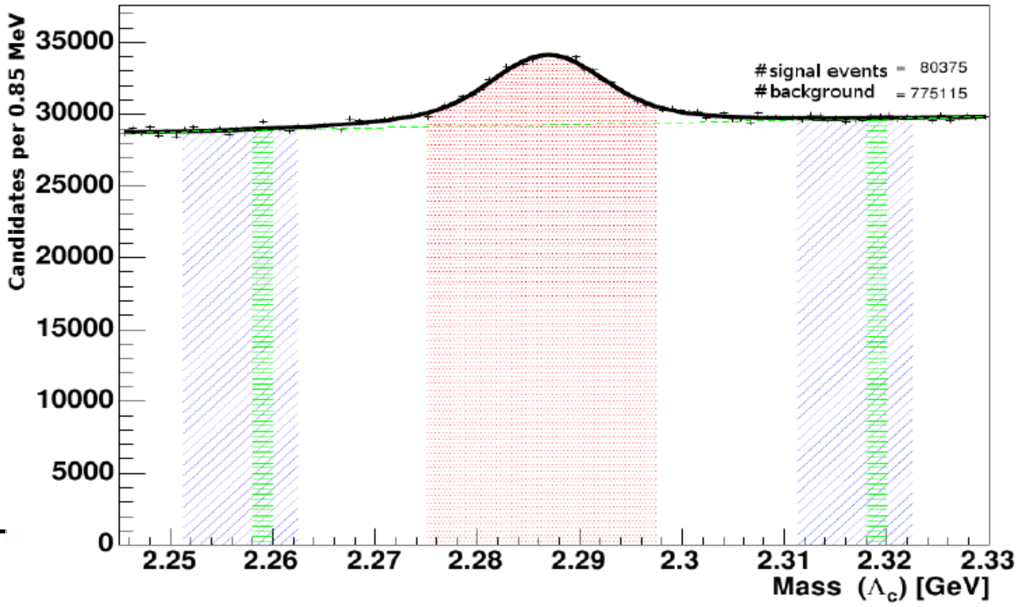
MVA methods are usually trained on Monte Carlo samples: “signal" and “background" events, applied to data to select real events of signal type.

MC could be inadequate => propose solution: training on reweighted data samples http://arxiv.org/abs/physics/0402083

Simple Side-Band subtraction:

- Determine signal/bckg ratio and find background only regions.
- Train MVA using data from signal region with background added with negative weigh.

region	amount of events	target type	weight
red	$N_{sig} + N_{back}$	signal	1
green	N_{sig}	background	1
blue	N_{back}	signal	-1



We assume: other variables are not correlated with invariant mass.

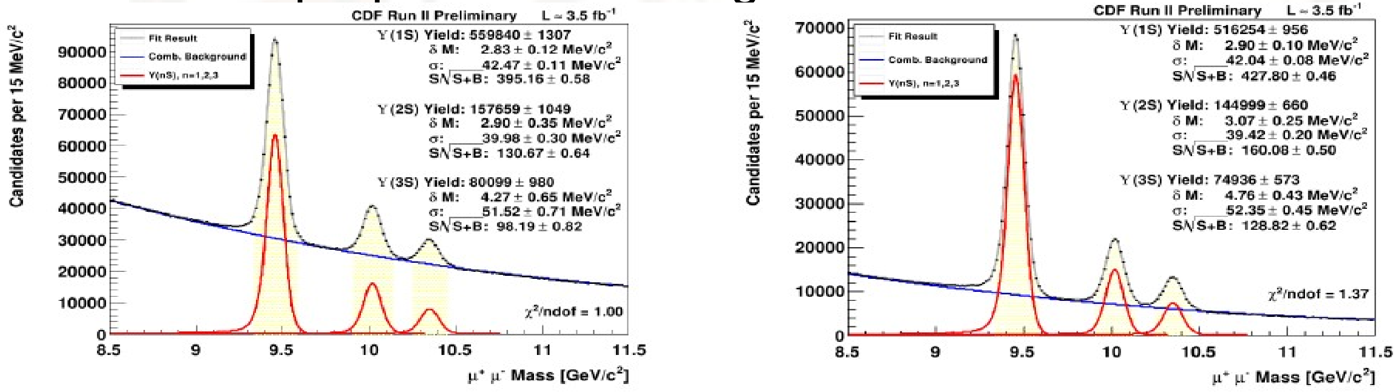
$_s$Plot method

Deduce the probability density function for the two event types signal and background, e.g. using function fits to a mass peak.

Transform pdfs into event weights. By applying the signal/background $_s$Plot weights to data, we will get the pdf of any variable independent of the one (mass) we used to deduce the pdfs in the first place.

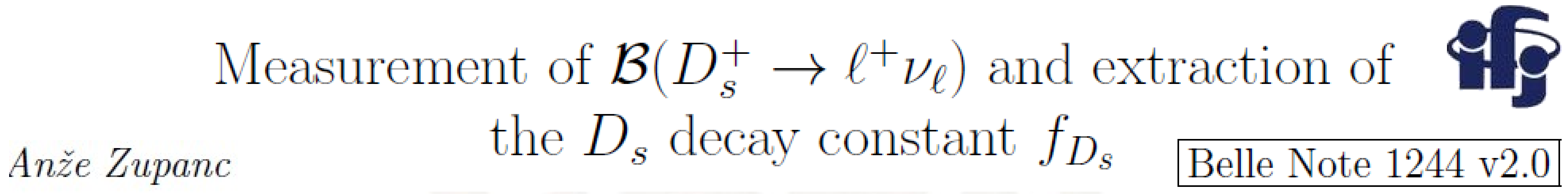
We use each event twice, once as signal (weight w_s) and once as background (weight w_{bg}). MVA will learn that there are certain events in the training sample which have a high signal weight and a lower background weight, and reside in a different phase space than the rest of the sample and will learn their properties to distinguish them.

Baron Műnchausen



CDF example

Figure 6. Υ resonances before and after cutting on a $_s$Plot trained NeuroBayes network. Extracted from Phd thesis Claudia Marino figure 5.8



Measurement of $\mathcal{B}(D_s^+ \to \ell^+\nu_\ell)$ and extraction of the D_s decay constant f_{D_s}

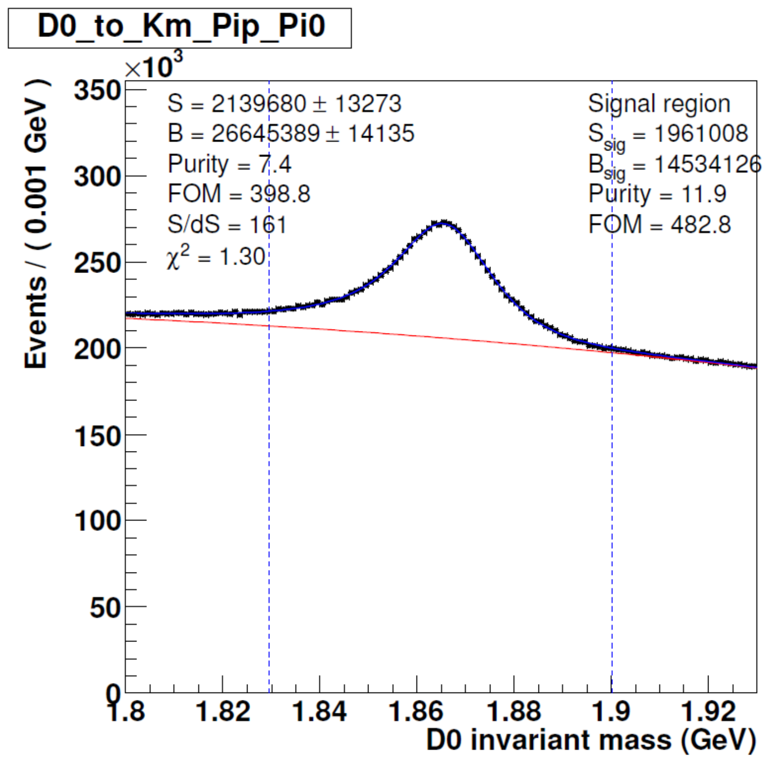
Anže Zupanc

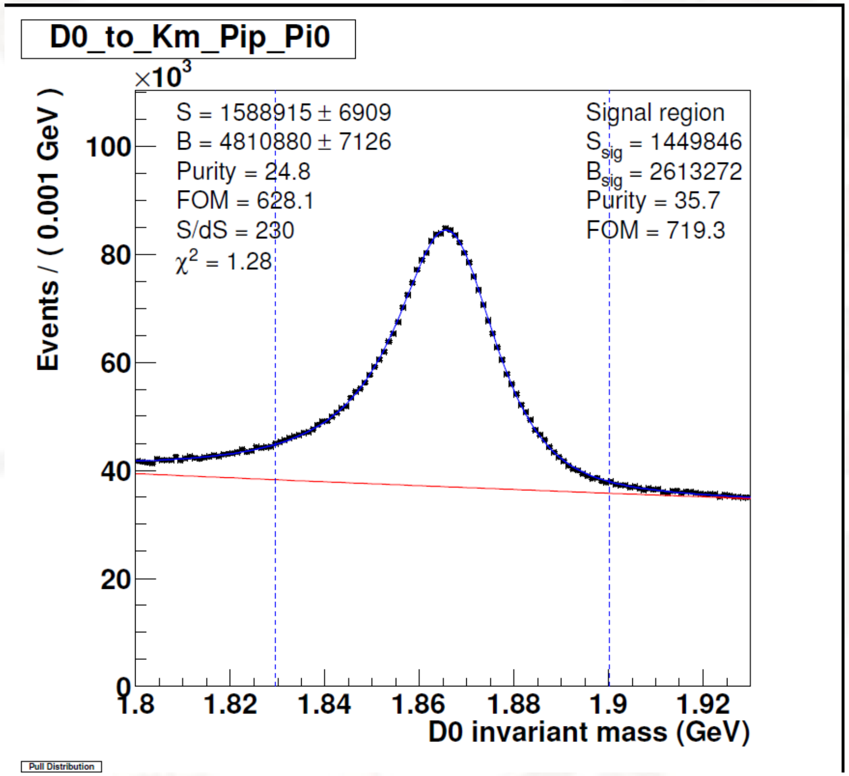
Belle Note 1244 v2.0

For signal selection the $_s$plot technique from NeuroBayes package is used.

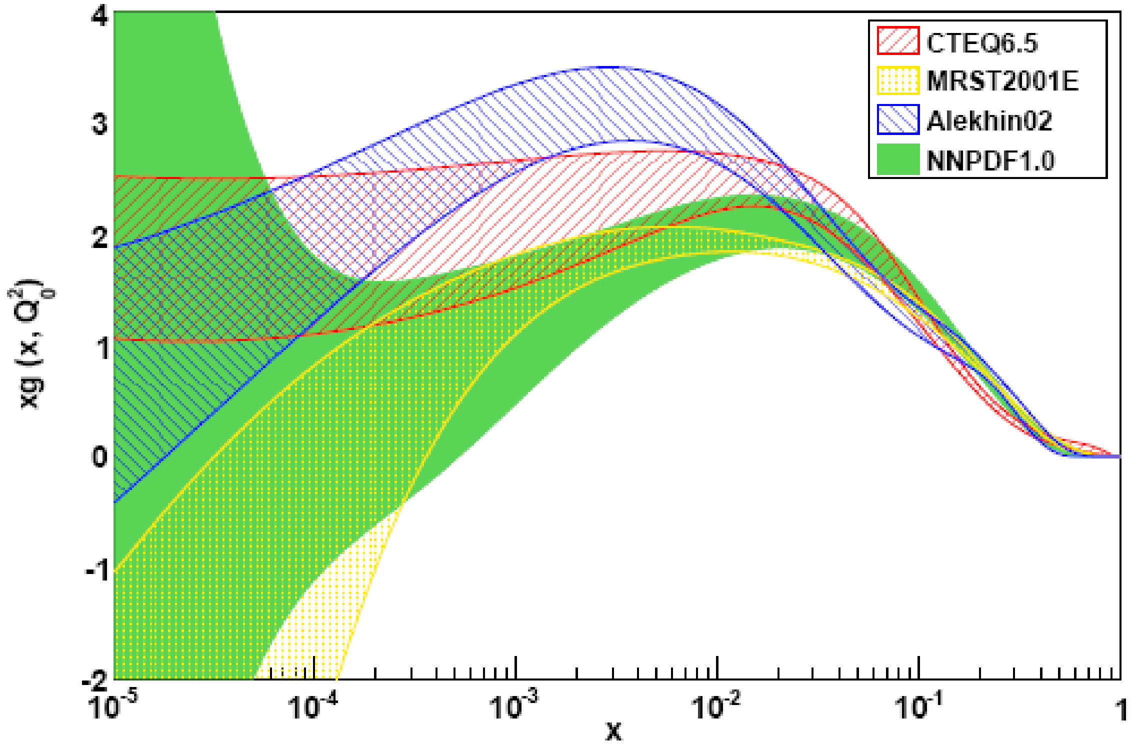
The neural network is trained on real, not simulated data, while the pdfs are found using a set of control variables (uncorrelated with discriminating variables). Signal and background are fitted with analytic functions (Gaussian, Student's, polynomial).

Signal efficiency - 84%, background rejection - 75%





Unbinned multidimensional fit – Parton Distributions



Gluon distribution

PDFs modeled with **neural networks.**

“We present the determination of a **set of parton distributions of the nucleon**, at next to leading order, from a global set of deep-inelastic scattering data: NNPDF1.0 The determination is based on a Monte Carlo approach, **with neural networks used as unbiased interpolants.** This method [...] is designed to provide a faithful and statistically sound representation of the uncertainty on parton distributions.”

The NNPDF Collaboration, R.D. Ball et al., arXiv: 0808.1231v2

NNPDF approach to parton distribution functions

Monte Carlo generation of replica

All errors as given by experimental collaborations are translated into a Monte Carlo set of artificial data. This set does reproduce all the means, errors, correlations. Many replicas are created.

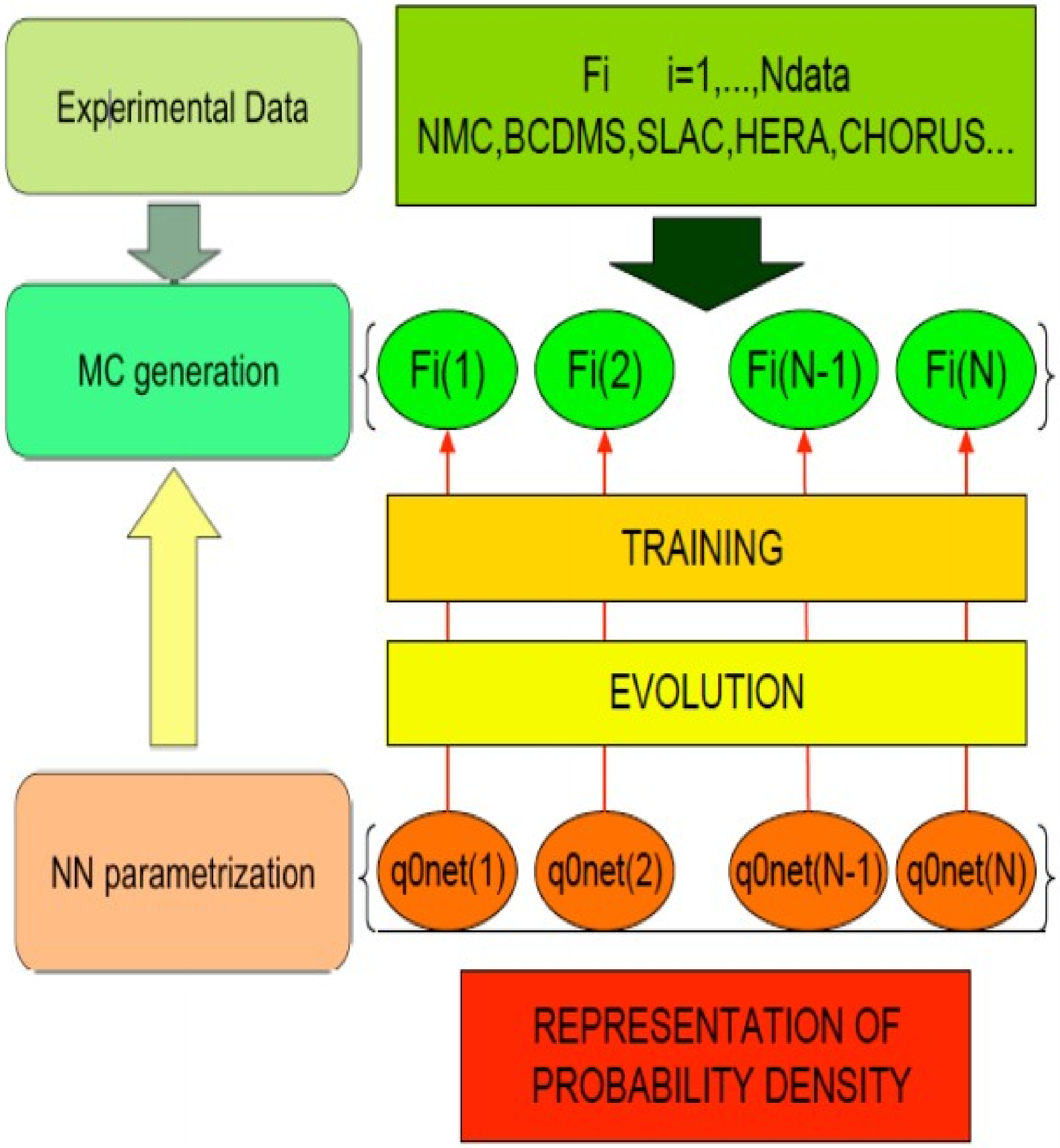
Construction and evolution of parton distributions

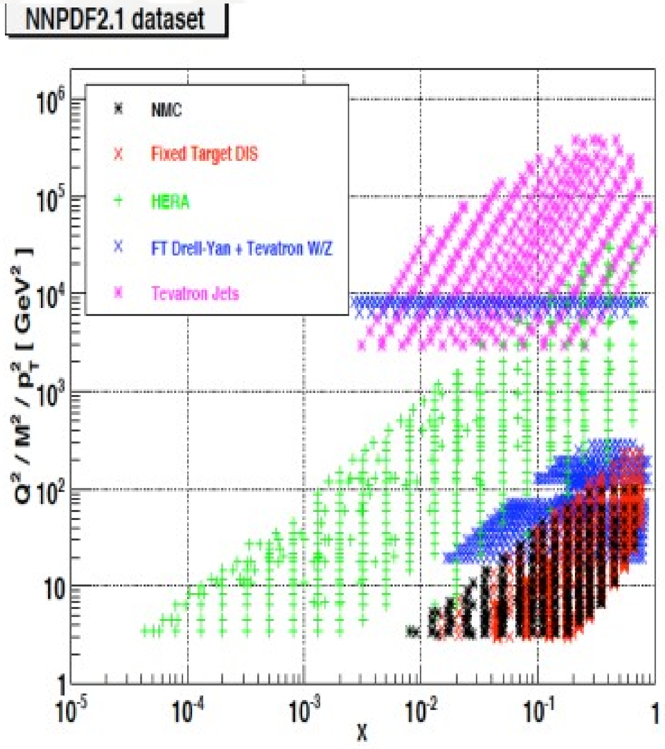
PDFs are constructed as Neural Networks in real space for each replica. We obtain many different PDFs.

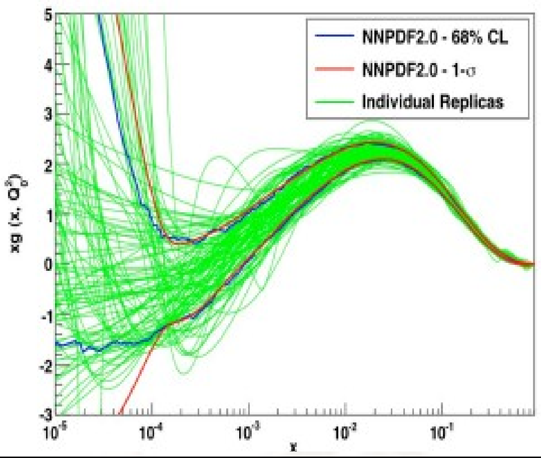
Statistical faithfulness

The final set of neural PDFs can then be used to reproduce observables, including errors and their correlations.

NNPDF


Experimental Data
Fi i=1,...,Ndata
NMC,BCDMS,SLAC,HERA,CHORUS...
MC generation
Fi(1)
Fi(2)
Fi(N-1)
Fi(N)
TRAINING
EVOLUTION
NN parametrization
q0net(1)
q0net(2)
q0net(N-1)
q0net(N)
REPRESENTATION OF PROBABILITY DENSITY


NNPDF2.1 dataset
NMC
Fixed Target DIS
HERA
FT Drell-Yan + Tevatron W/Z
Tevatron Jets
$Q^2 / M^2 / p_T^2$ [GeV^2]
x


NNPDF2.0 - 68% CL
NNPDF2.0 - 1-σ
Individual Replicas
$xg(x, Q_0^2)$
x

Spin in tau decays into 3 pions
(see Zbyszek talk)

Important for spin measurements – information about decaying particle

Difficult problem – combinatorics with many .particles, each described by a couple of variables.

There was an attempt to attack the problem with Neural Network – failed.

Is there enough information in data? If we try all the combinations and choose the one with highest probability, does it help?

Maybe one should find an error on reconstructed "variables" and reject/supress events with large errors?

It is not an easy task...

Summary

Multivariate methods can be applied to many aspects of data analysis, not only for signal/background selection. Many practical methods, and convenient tools such as TMVA, are available for regression and classification.

All methods approximate the same mathematical entities, but no one is guaranteed to be the best in all circumstances. Simplicity, speed of learning and robustness also matters. So, experiment with a few of them!

General questions

We need methods, and convenient tools, to explore and quantify the quality of modeling of n-dimensional data.

Is there a sensible way to use multivariate methods when one does not know for certain where to look for signals?

Additional slides

Baron Münchausen

Opened issues

Verification

• How can one confirm that an n-dimensional density is well-modeled?

• How can one find, characterize, and exclude, discrepant domains in n-dimensions automatically?

• How can one automate re-weighting of model data, event-by-event, in order to improve the match between real data and the model?

• How can one verify that a classifier function is close to the Bayes limit?

Looking Beyond the Lamppost

• Is there a robust and useful way to quantify the information content of a sample of n-dimensional signal and background data so that when it is compressed, say to 1-dimension, one is able to assess how much information has been lost?

• Is there a sensible way to use multivariate methods when one does not know for certain where to look for signals? In particular, is there a useful way to compress data even if one does not know for certain what the signal should look like?

Systematics – some concepts

Same steps as using cut based analysis:

Systematic errors:

Vary all aspects of the model about which one is uncertain and run the analysis.

Get distribution of results (ex. Higgs masses)

Usual error propagation fails – non-linearity of the error propagation

Robustness (stability):

Different or differently trained MV leads in general to different results.

Solution should be unaffected by these variations.

“In summary, handling systematic errors with MV-based analysis is no different from the procedure for cut-based analysis 99% of the effort, as you know, is convincing yourself that your N-dimensional model agrees sufficiently well with the N-dimensional data to be trustworthy!”

H. B. Prosper

Event Weighting

The probability ***p(S|x)*** is optimal in another sense:

If one ***weights*** an admixture of **signal** and **background** events by the weight function

$$W(x) = p(S|x)$$

then the ***signal*** strength will be extracted with ***zero bias*** and the ***smallest possible variance***, provided that our models describe the signal and background densities accurately and the signal to background ratio $p(S)/p(B)$ is equal to the true value.

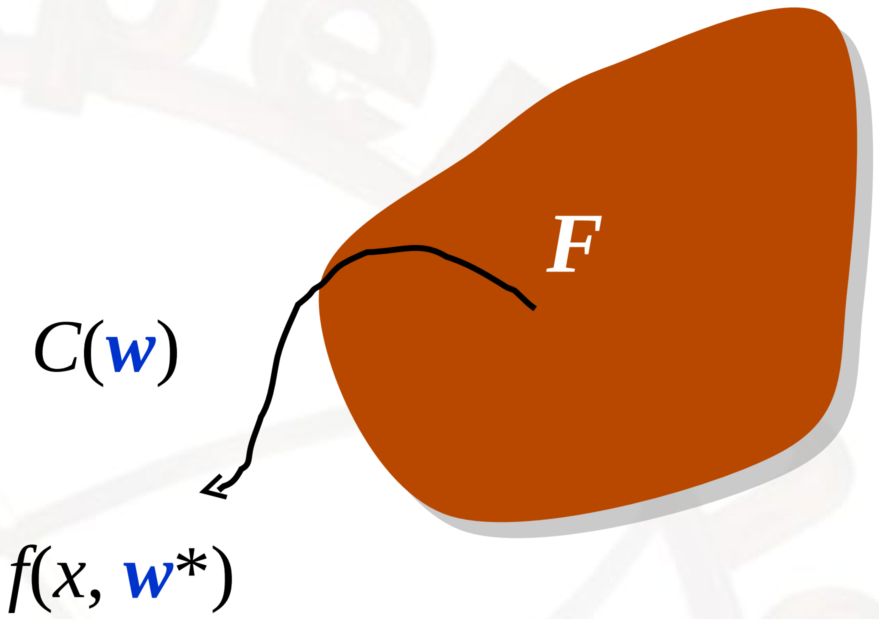
We can recover signal by event weighting with the discriminant output without cutting on it.

Roger Barlow, J. Comp. Phys. 72, 202 (1987)

Machine Learning

We have to chose:

Function class	***F*** **= {** $f(x, \mathbf{w})$ **}**
Constraint	***C***
Loss function	***L***
	example:
	$L = y_i^2 - f^2(x_i, w)$
Training data	$T(y,x)$



Method

Find $f(x, \mathbf{w})$ by minimizing the **empirical risk** ***R***

$$R(w) = \frac{1}{N} \sum_{i=1}^{N} L(y_i, f(x_i, w))$$

subject to the constraint $C(\mathbf{w})$

We choose at the end a single “best” function $f(x, \mathbf{w})$
(best single Neural Network, best likelihood etc.)

Bayesian Learning

Choose

Function class	$F = \{ f(x, \mathbf{w}) \}$
Prior	$p(\mathbf{w})$
Likelihood	$p(\mathbf{y}\|\mathbf{x}, \mathbf{w})$ *probability of obtaining an output **y** for given **x, w***
Training data	$T(\mathbf{y},\mathbf{x})$

Method

Use Bayes' theorem to infer the parameters:

$$p(\mathbf{w}|T) = \{p(\mathbf{y}|\mathbf{x}, \mathbf{w})\, p(\mathbf{x}|\mathbf{w})\, p(\mathbf{w})\}/\{p(\mathbf{y}|\mathbf{x})\, p(\mathbf{x})\}$$

because $p(w|y,x) = p(\mathbf{y}|\mathbf{x}, \mathbf{w})\, p(\mathbf{w}|\mathbf{x})/p(\mathbf{y}|\mathbf{x})$

$\sim p(\mathbf{y}|\mathbf{x}, \mathbf{w}) \; * \; p(\mathbf{w})$ (assume $p(\mathbf{x}|\mathbf{w}) = p(\mathbf{x})$)

~ Likelihood * Prior

We do not pick up a single function *f(x)*, instead $p(\mathbf{w}|T)$ assigns a probability density to each function in the function class.

Bayesian Learning: Why?

<u>Probabilistic learning</u>: Calculate explicit probabilities for hypothesis, among the most practical approaches to certain types of learning problems.

<u>Probabilistic prediction</u>: Predict multiple hypotheses, weighted by their probabilities.

<u>Incremental</u>: Each training example can incrementally increase or decrease the probability that a hypothesis is correct. Prior knowledge can be combined with observed data.

<u>Standard</u>: Even when Bayesian methods are computationally intractable, they can provide a standard of optimal decision making against which other methods can be measured.

Classification Example from D0 experiment

Dots

$D(x) = H_S/(H_S+H_B)$

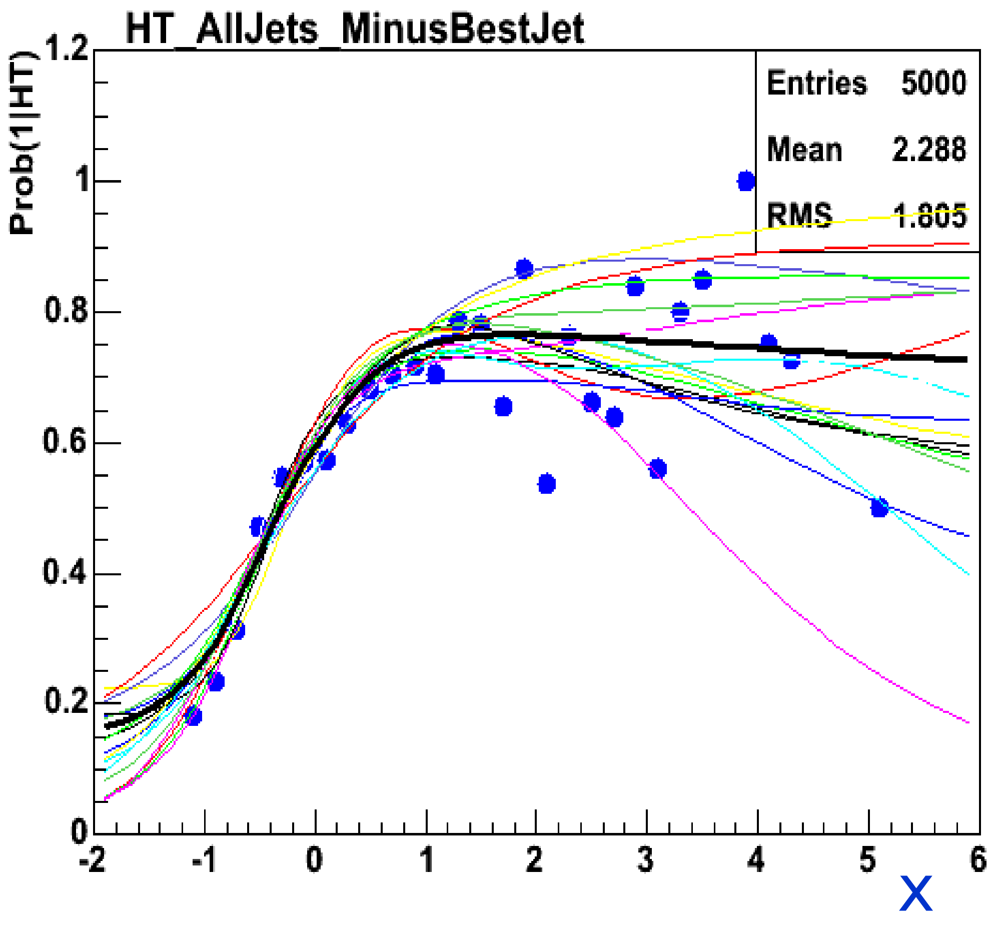
H_S signal histogram

H_B background histogram

Curves

Individual neural networks

$n(x, \mathbf{w}_k)$



Black curve

$D(x) = E[\, n(x, \mathbf{w})\,] = (1/N) \sum n(x, \mathbf{w}_k)$