Statistical Methods in Particle Physics

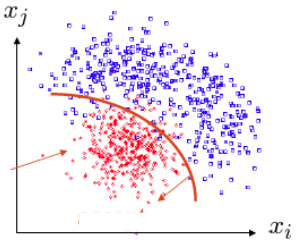
Lecture 8

December 3, 2012

Silvia Masciocchi, GSI Darmstadt
s.masciocchi@gsi.de

Winter Semester 2012 / 13

Opening

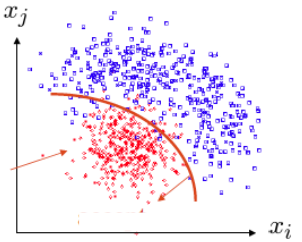


Today we talk of **STATISTICAL TESTS**

→ a way to find criteria to select candidate events (or particles, …) for further analysis (e.g. signal vs background)

The goal of test statistics is to make a statement about how well the observed data stand in agreement with given predicted probabilities, i.e. with *hypotheses*

Hypotheses



A ***hypothesis H*** specifies the probability for the data

i.e., the outcome of the observation, here symbolically “x”

We can write:

$$x \sim f(x|H)$$

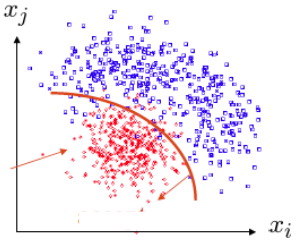
x can be uni-/multivariate, continuous or discrete

x could represent for example the observation of a single particle, a single event, or an entire “experiment”

Possible values of x form the **sample space S** (or “data space”)

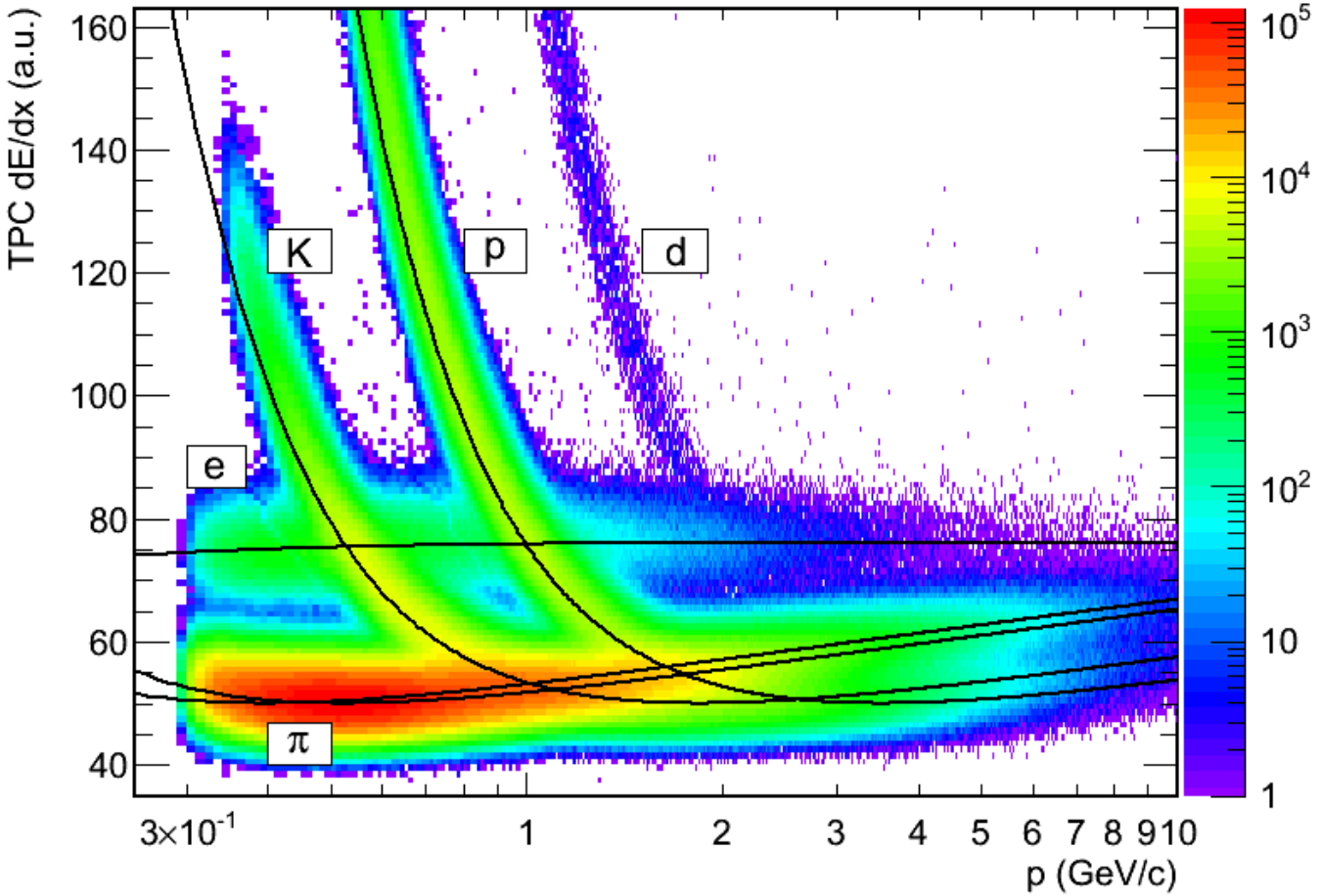
The probability for x given H is also called the **likelihood of the hypothesis,** written $L(x|H)$

Example: particle identification



Use the ALICE Time Projection Chamber to identify the particle species: electron, muon, pion, kaon, proton, deuteron

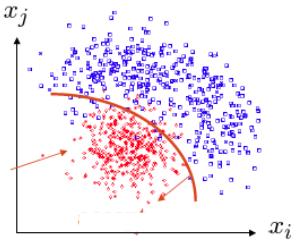
"x" = particle momentum (p), specific energy loss in TPC (dE/dx) (and more)



Example:
I want to select electrons (hypothesis H_1) from all other particles (hypothesis H_0)

In Bayesian approach: Can add prior hypotheses on the relative particle abundances (e.g. you see that pions are many more!)

Rejection / acceptance regions



Goal is to make some statement based on the observed data x, as to the validity of the possible hypotheses.

A test of hypothesis H_0 is defined by specifying a **critical region** W (also called **rejection** region) of the data space S, such that there is no more than some (small) probability α, assuming H_0 is correct, to observe the data there:

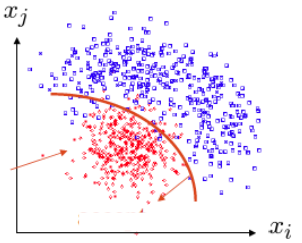
$$P(x \in W | H_0) \leq \alpha$$

If x is observed there, reject H_0.

α is called the **size or significance level** of the test.

The complementary region is called **acceptance region**.

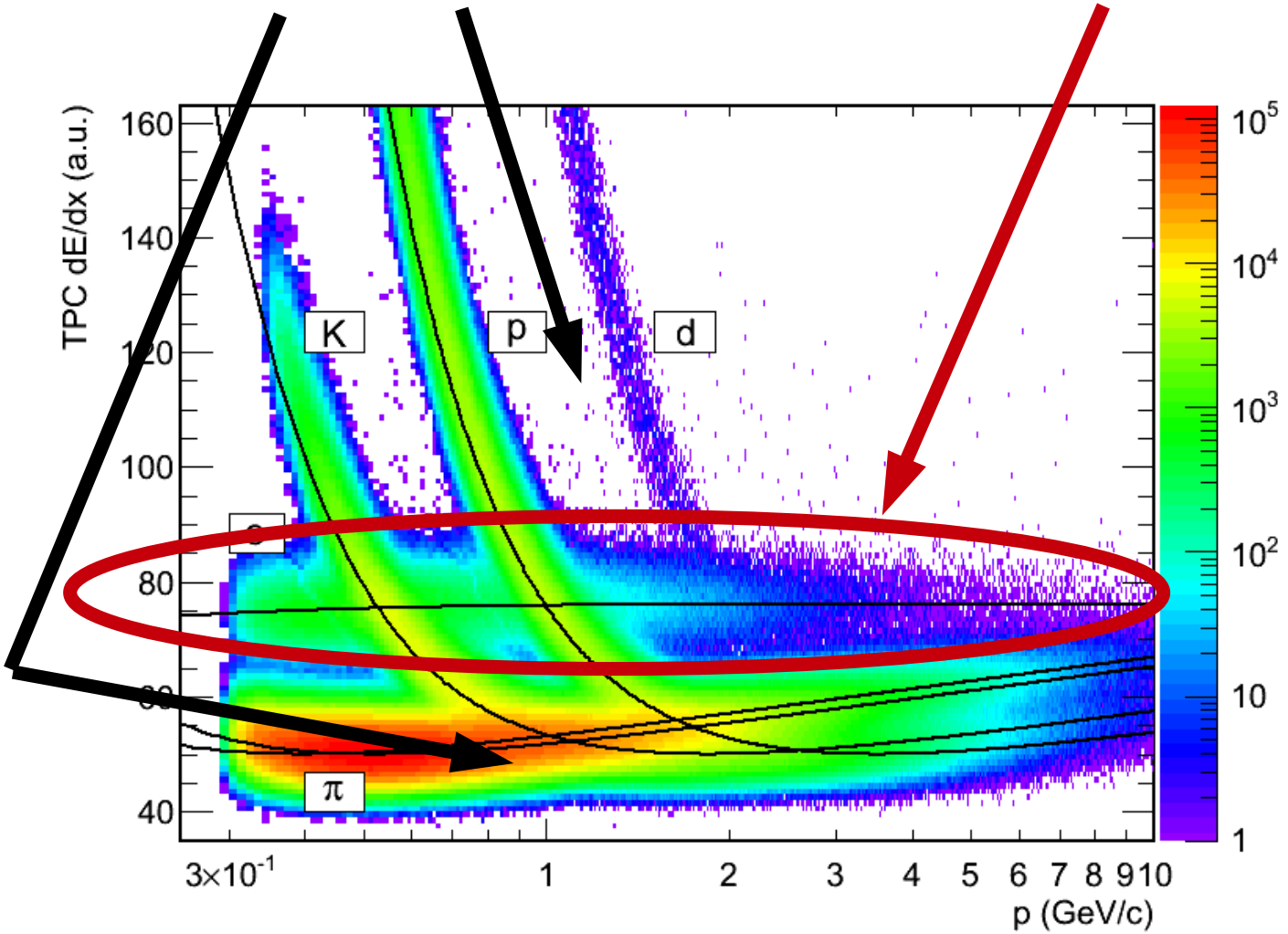
Again the example



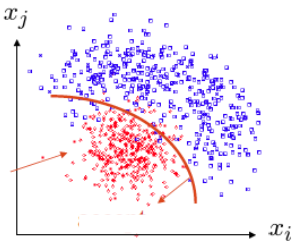
Looking for electrons: null hypothesis H_0 is to be a hadron

Acceptance region here:

Rejection region here (for bkg = hadrons)



Any decision involves a certain risk ...

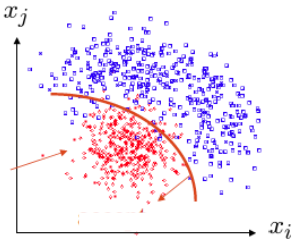


Event by event, we need to decide whether to take it as signal or as background

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 II error
back-ground	Type I error	☺

Type-I, Type-II errors



Rejecting the hypothesis H_0 when it is true is a Type-I error.

The maximum probability for this is the **size of the test**:

$$P(x \in W | H_0) \leq \alpha$$

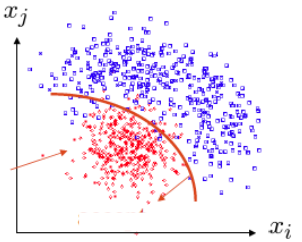
But we might also accept H_0 when it is false and an alternative H_1 is true.
This is called Type-II error, and occurs with probability:

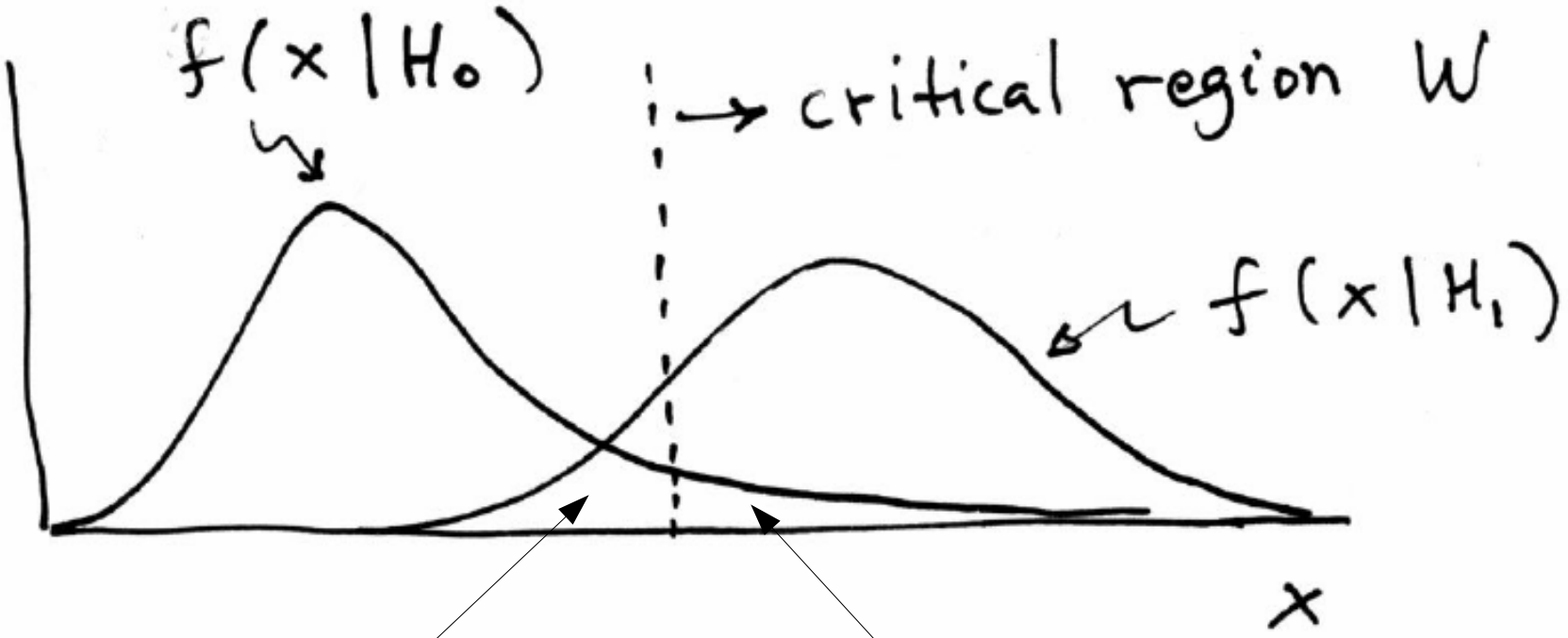
$$P(x \in S - W | H_1) = \beta$$

One minus this is called the power of the test with respect to the alternative hypothesis H_1:

Power = 1 - β

Type-I, Type-II errors

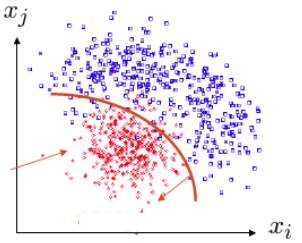




$$P(x \in S - W | H_1) = \beta$$

$$P(x \in W | H_0) \leq \alpha$$

Selecting events



We have a data sample with two kinds of events, corresponding to hypotheses H_0 (background) and H_1 (signal).

We want to select those of type H_1.

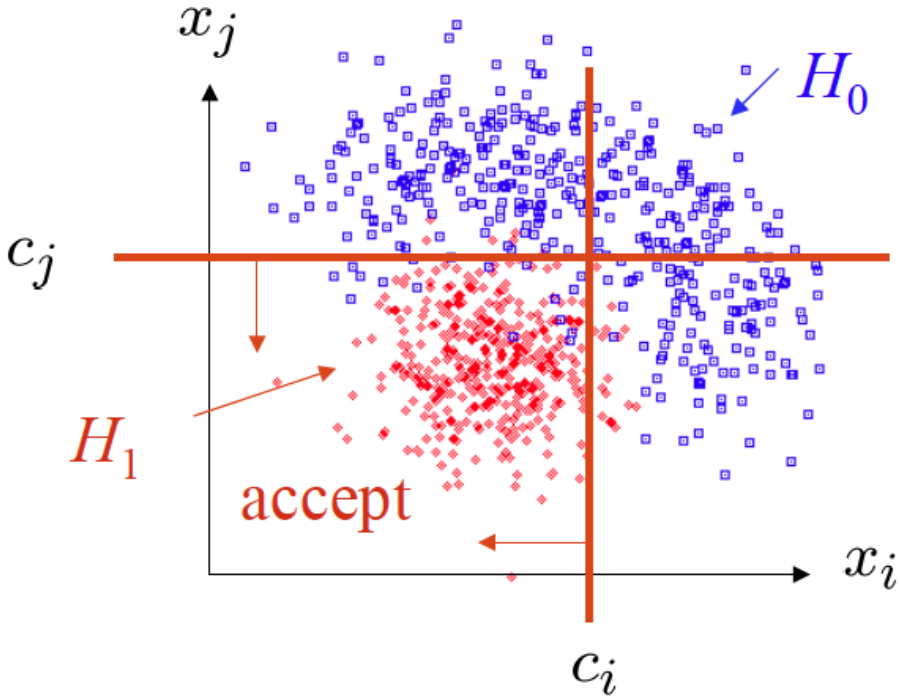
Each event is a point in $\vec{x}$ space (n dimensions).

What 'decision boundary' should we use to accept/reject events as belonging to event types H_0 or H_1?

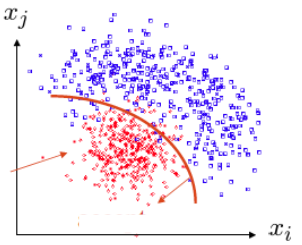
One possibility is to select events with several 'cuts': e.g.

$x_i < c_i$

$x_j < c_j$

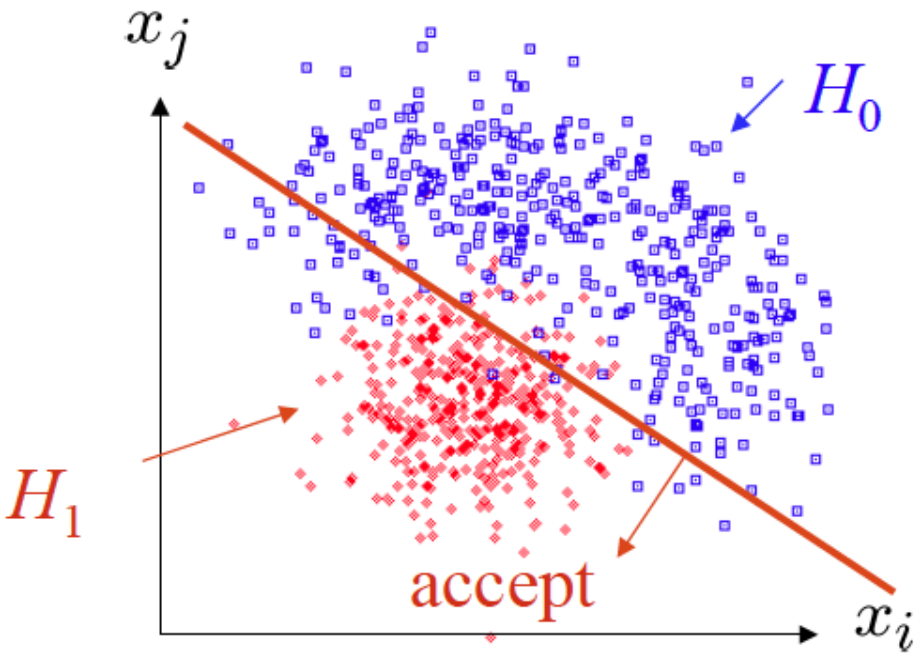


Other selections

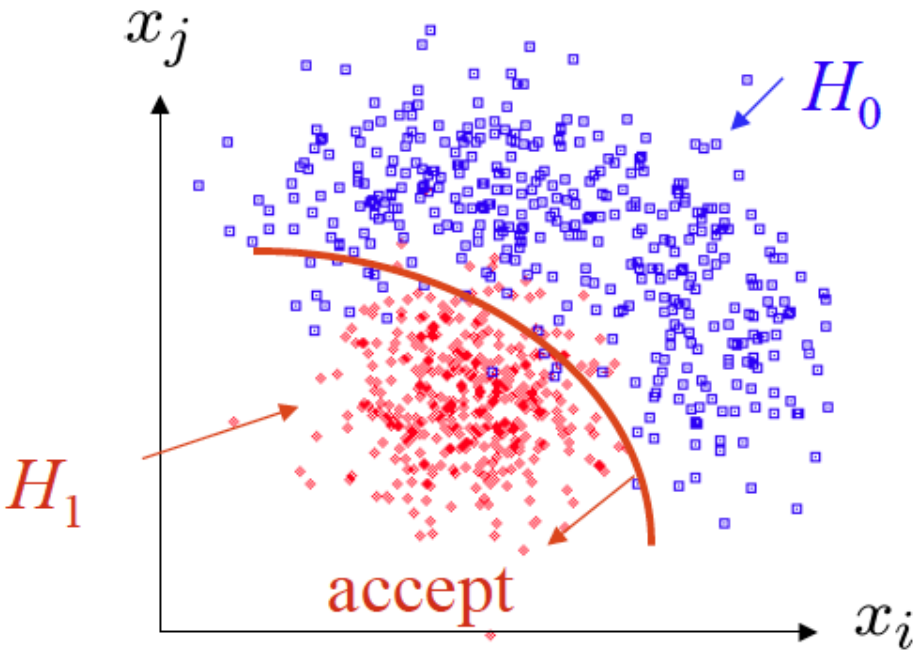


But we can also use some other sort of decision boundary !!

linear

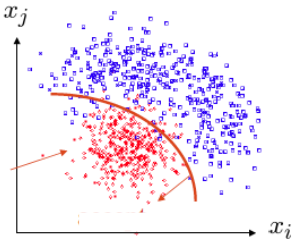


or nonlinear



How can we formalize this to choose the boundary in an 'optimal' way?

Scalar / multidimensional



In addition:

$\vec{x}$ is the result of the measurements, n can be large

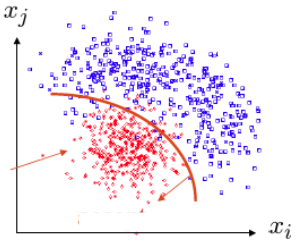
$\vec{x}$ follows some joint pdf in an n-dimensional space

Usually it is awkward to work with multidimensions!

At first we try to construct a test statistic of lower dimension (e.g. scalar):

- Compactify the data
- Try not to loose the ability to discriminate between hypotheses

Multivariate analysis (MVA)



- Map the n-dimensional space of the observable variables ("feature" space of our measurements) to one dimensional output

$$\mathbb{R}^n \rightarrow \mathbb{R}$$

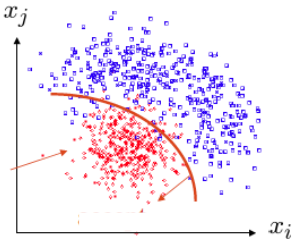
Test statistic

$$(x_1, \ldots, x_n) \rightarrow t(\vec{x})$$

- There are model classes for this
  - Various types: linear, non-linear, flexible, less flexible
- We can use previous knowledge, "known" or "previously solved" cases
- The resulting class (description) should have good generalization properties

Often associated with the term of "machine learning"

Test statistics



The decision boundary can be defined by an equation of the form:

$$t(x_1, ..., x_n) = \text{constant} = t_{cut}$$

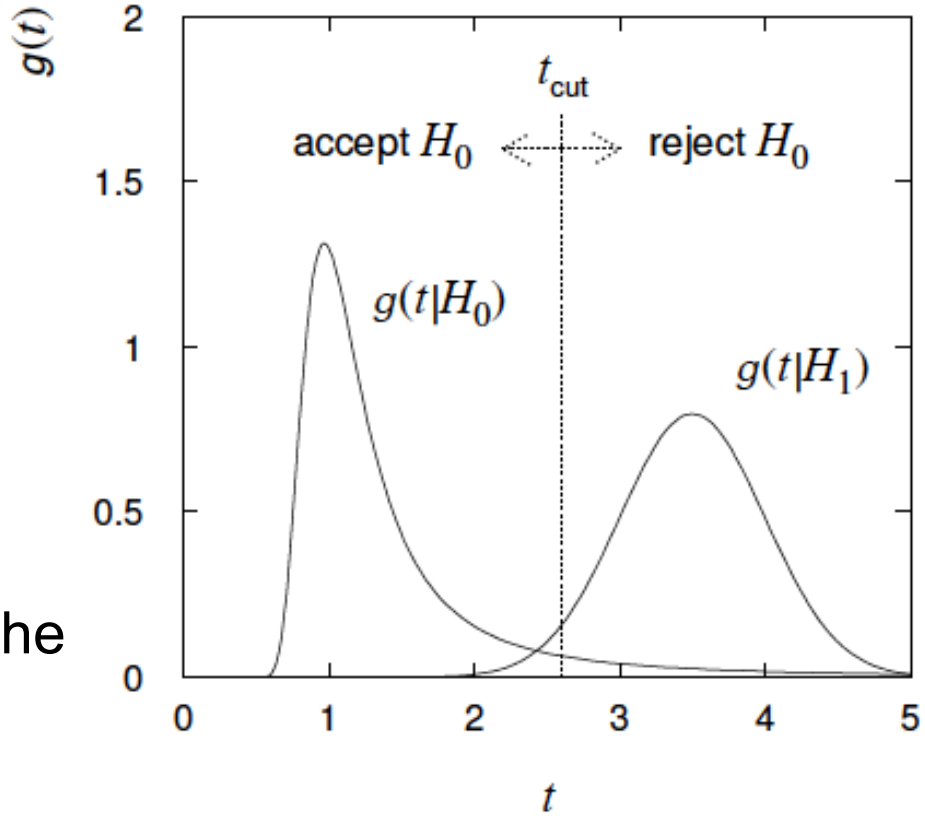
where $t(x_1, ..., x_n)$ is a scalar test statistic

We can work out the pdf's:

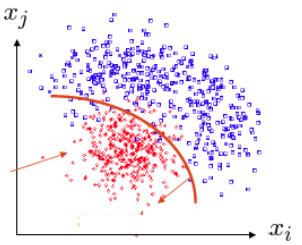
$$g(t|H_0), \quad g(t|H_1)$$

Decision boundary is now a single 'cut' on t, which divides the space into the critical (rejection region) and the acceptance region.

This defines a TEST: if the data fall in the critical region, we reject H_0



Signal / background efficiency

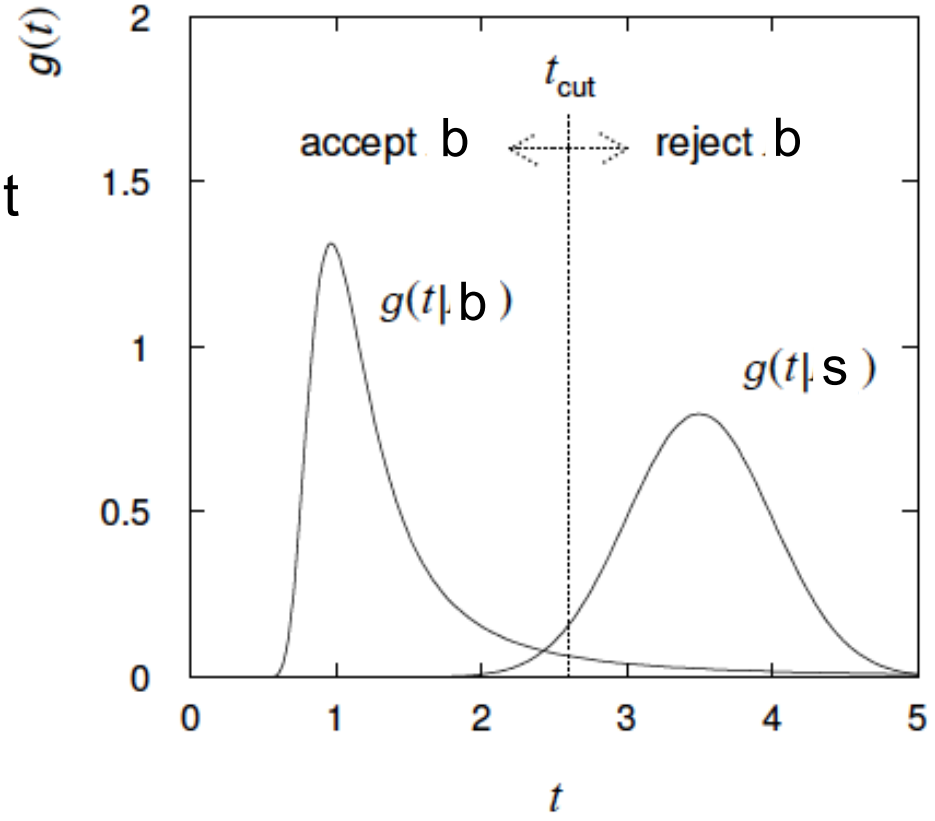


The probability to reject background hypothesis for a background event (background efficiency) is:

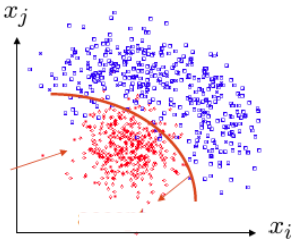
$$\epsilon_b = \int_{t_{cut}}^{\infty} g(t|b)\, dt = \alpha$$

The probability to accept a signal event as signal (signal efficiency) is:

$$\epsilon_s = \int_{t_{cut}}^{\infty} g(t|s)\, dt = 1-\beta$$



Purity of event selection



Suppose only one type of background b.
Overall fractions of signal and background events are π_s and π_b (prior probabilities).

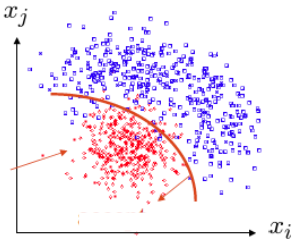
Suppose we select signal events with $t>t_{cut}$. What is the PURITY of the selected sample?

PURITY means the probability to be signal given that the event was accepted. Using Bayes' theorem we find:

$$P(s|t>t_{cut}) = \frac{P(t>t_{cut}|s)\pi_s}{P(t>t_{cut}|s)\pi_s + P(t>t_{cut}|b)\pi_b} = \frac{\epsilon_s \pi_s}{\epsilon_s \pi_s + \epsilon_b \pi_b}$$

→ the purity depends on the prior probabilities as well as on the signal and background efficiencies !!

How to optmize the choice?



How can we choose a test's critical region in an “optimal way”?

Neyman-Pearson lemma states:

To get the highest power for a given significance level (or highest purity for a given efficiency) in a test of H_0 (background) versus H_1 (signal), the critical region should have:

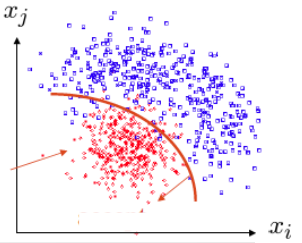
$$\frac{P(x|H_1)}{P(x|H_0)} > c$$

inside the region, and ≤c outside, where c is a constant which determines the power.

Equivalently, optimal scalar test statistics is: $t(x) = \frac{P(x|H_1)}{P(x|H_0)}$

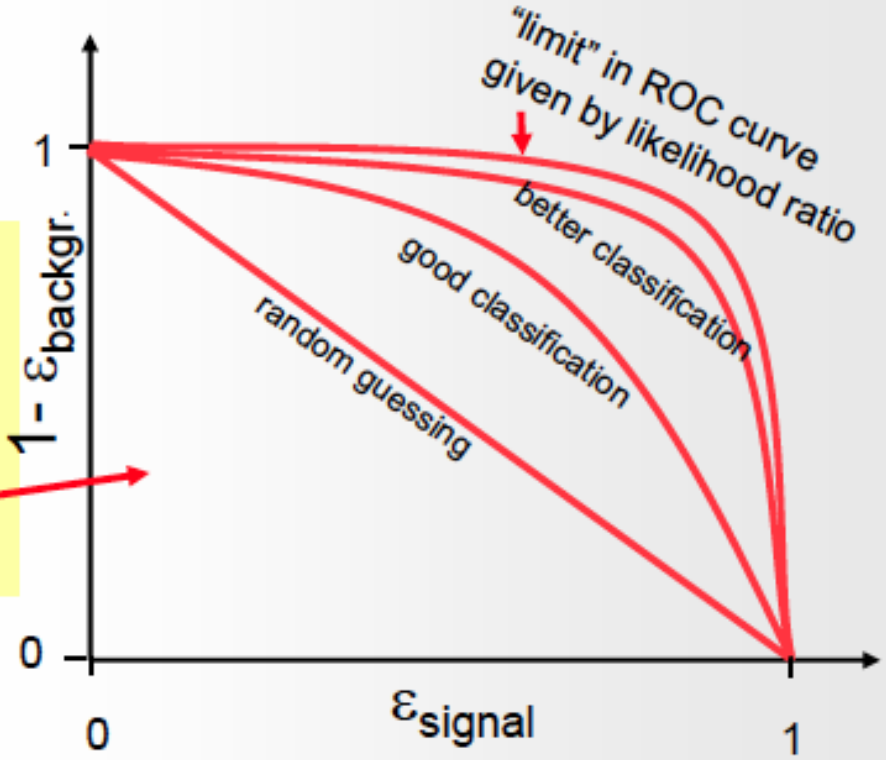
Likelihood ratio

Neyman-Pearson lemma



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

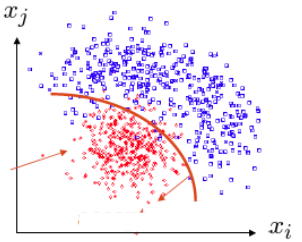
Neyman-Peason:
The Likelihood ratio used as "selection criterium" y(x) gives for each selection efficiency the best possible background rejection.
i.e. it maximises the area under the "Receiver Operation Characteristics" (ROC) curve



- y(x) is the discriminating function given by your estimator (i.e. the likelihood ratio)
- varying y(x)>"cut" moves the working point (efficiency and purity) along the ROC curve
- where to choose your working point? → need to know prior probabilities (abundances)

measurement of signal cross section:	maximum of $S/\sqrt{(S+B)}$ or equiv. $\sqrt{(\varepsilon \cdot p)}$
discovery of a signal (typically: S<<B):	maximum of $S/\sqrt{(B)}$
precision measurement:	high purity (p)
trigger selection:	high efficiency (ε)

BUT !!!



Usually we do ***NOT*** have explicit formulae for the pdfs $P(x|H_0), P(x|H_1)$

What we usually have are Monte Carlo models for signal and background processes, so we can produce simulated data, and enter each event into an n-dimensional histogram.

But then we need M bins for each of the n dimensions

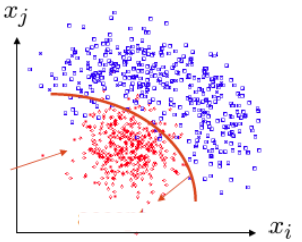
→ ***total of M^n cells !!***

If n is large, then we end up with a prohibitively large number of cells to populate with Monte Carlo data !!!

Compromise solution:

Make Ansatz for form of the test statistic t(x) with fewer parameters; determine them (e.g. using MC) to give best discrimination between signal and background!

Example



Distinguish between 2 processes:

H_0: $e^+e^- \rightarrow W\,W \rightarrow$ hadrons (usually 4 jets)

H_1: $e^+e^- \rightarrow q\bar{q} \rightarrow$ hadrons (usually 2 jets)

For each event we measure $\vec{x}$ (n. of hadrons, their momenta, jets, missing energy, angles between jets, etc etc)

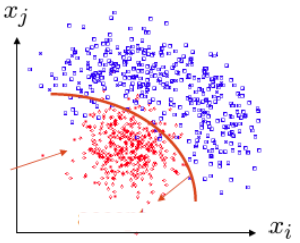
According to Neyman-Pearson, to select WW's we should cut on

$$t(\vec{x}) = \frac{f(\vec{x}|H_0)}{f(\vec{x}|H_1)}$$

But we do not know entirely these pdf's !!!

Partly help with MC, partly simplify / transform the description of t

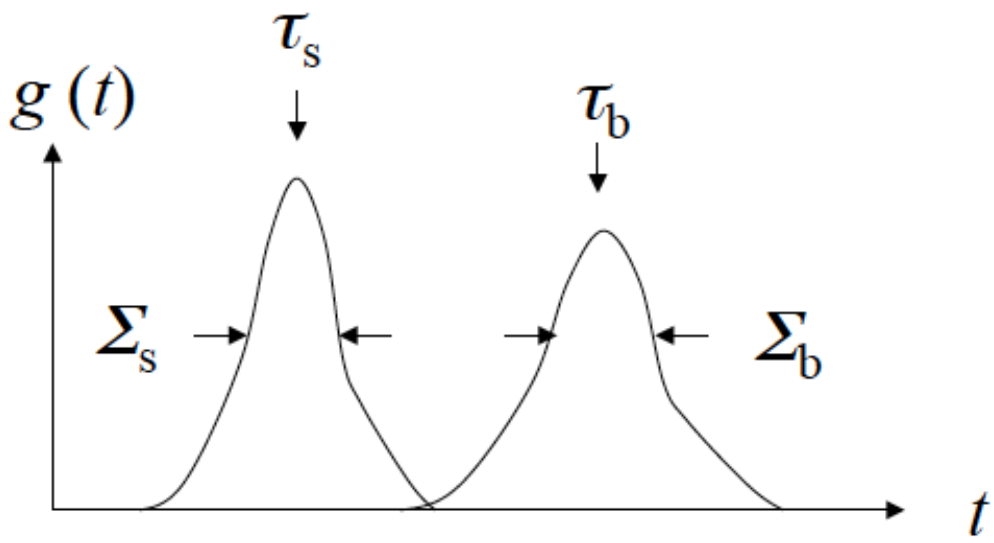
Linear test statistic



Ansatz: $$t(\vec{x}) = \sum_{i=1}^{n} a_i x_i = \vec{a}^T \vec{x}$$

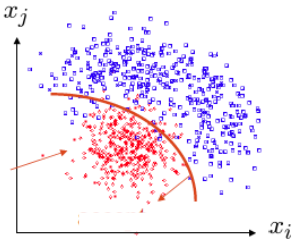
Choose the parameters a_1, .. a_n so that the pdf's $g(t|s), g(t|b)$ have maximum SEPARATION:

We want large distance between the mean values and small widths



Fisher: maximize $$J(\vec{a}) = \frac{(\tau_s - \tau_b)^2}{\Sigma_s^2 + \Sigma_b^2}$$

Maximum separation - 1



Reasoning to come
to Fisher's statement

Hypotheses: k = 0,1

Measurement: $\vec{x}$ i, j = 1, ..., n (components)

Means and variances for the x_i:

$$(\mu_k)_i = \int x_i \, f(\vec{x}|H_k) \, d\vec{x}$$

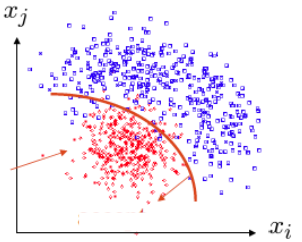
$$(V_k)_{ij} = \int (x-\mu_k)_i \, (x-\mu_k)_j \, f(\vec{x}|H_k) \, d\vec{x}$$

In terms of mean and variance of $t(\vec{x})$ this becomes:

$$\tau_k = \int t(\vec{x}) \, f(\vec{x}|H_k) \, d\vec{x} = \vec{a}^T \vec{\mu}_k$$

$$\Sigma_k^2 = \int (t(\vec{x}) - \tau_k)^2 \, f(\vec{x}|H_k) \, d\vec{x} = \vec{a}^T \, V_k \, \vec{a}$$

Maximum separation - 2



The numerator of $J(\vec{a})$ is:

$$(\tau_0 - \tau_1)^2 = \sum_{i,j=1}^{n} a_i a_j (\mu_0 - \mu_1)_i (\mu_0 - \mu_1)_j$$

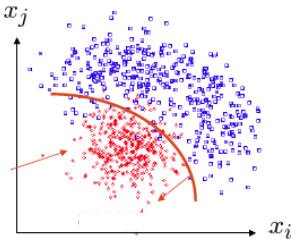
$$= \sum_{i,j=1}^{n} a_i a_j B_{ij} = \vec{a}^T B \vec{a}$$

The denominator:

$$\Sigma_0^2 + \Sigma_1^2 = \sum_{i,j=1}^{n} a_i a_j (V_0 + V_1)_{ij} = \vec{a}^T W \vec{a}$$

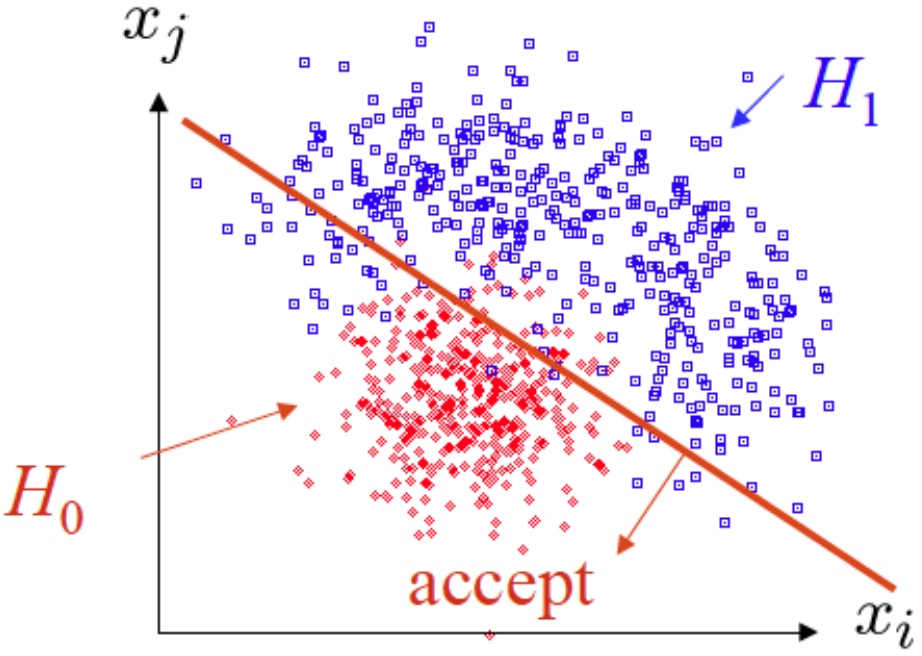
Maximize $$J(\vec{a}) = \frac{\vec{a}^T B \vec{a}}{\vec{a}^T W \vec{a}} = \frac{\text{separation between classes}}{\text{separation within classes}}$$

Fisher's linear discriminant



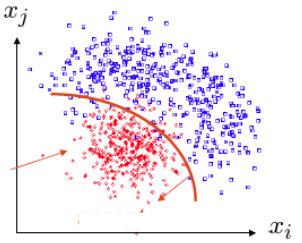
Setting: $\frac{\partial J}{\partial a_i} = 0$ gives Fisher's linear discriminant function:

$$t(\vec{x}) = \vec{a}^T \vec{x}, \quad \text{with} \quad \vec{a} \propto W^{-1}(\vec{\mu}_0 - \vec{\mu}_1)$$

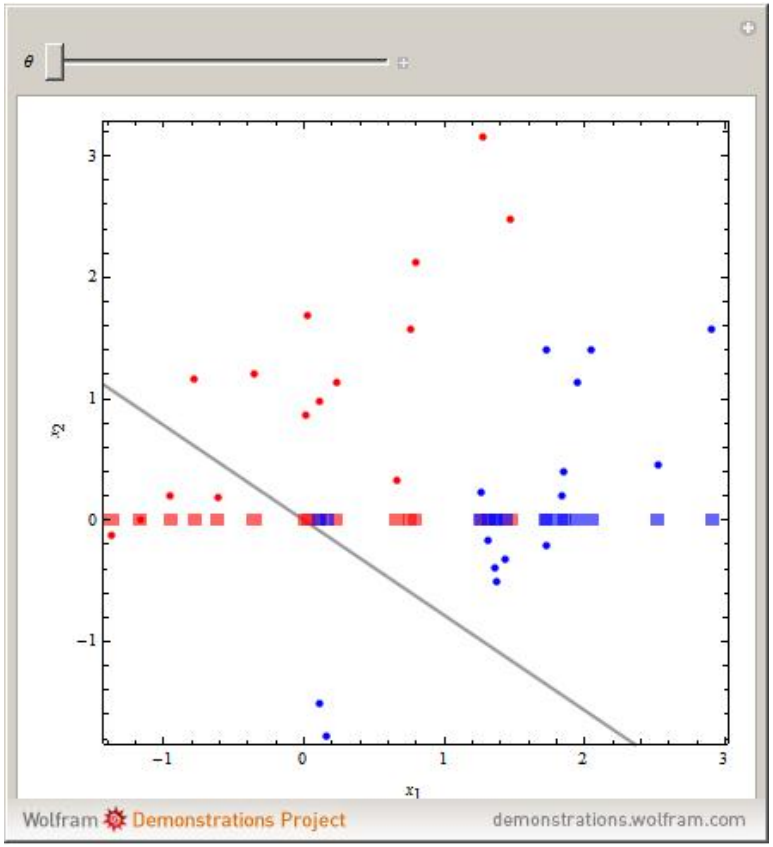


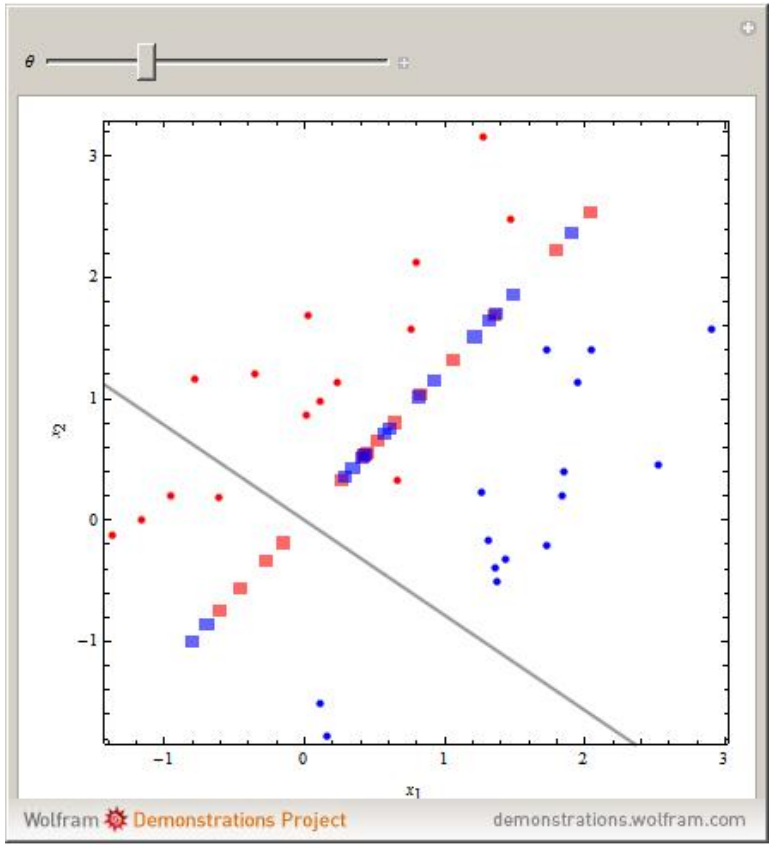
Corresponds to a linear decision boundary

Another illustration - 1

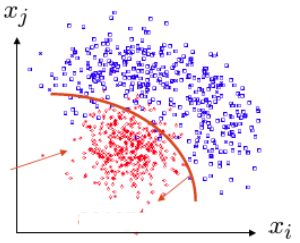


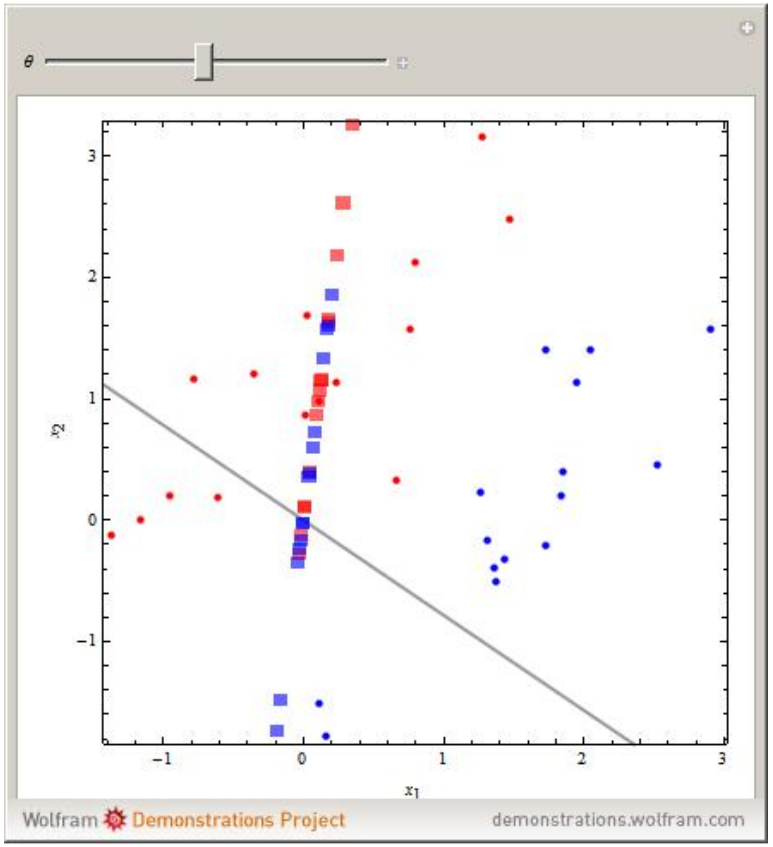
Fisher linear discriminant analysis determines a canonical direction for which the data is most separated when projected on a line in this direction. The solid gray line shows the canonical direction.

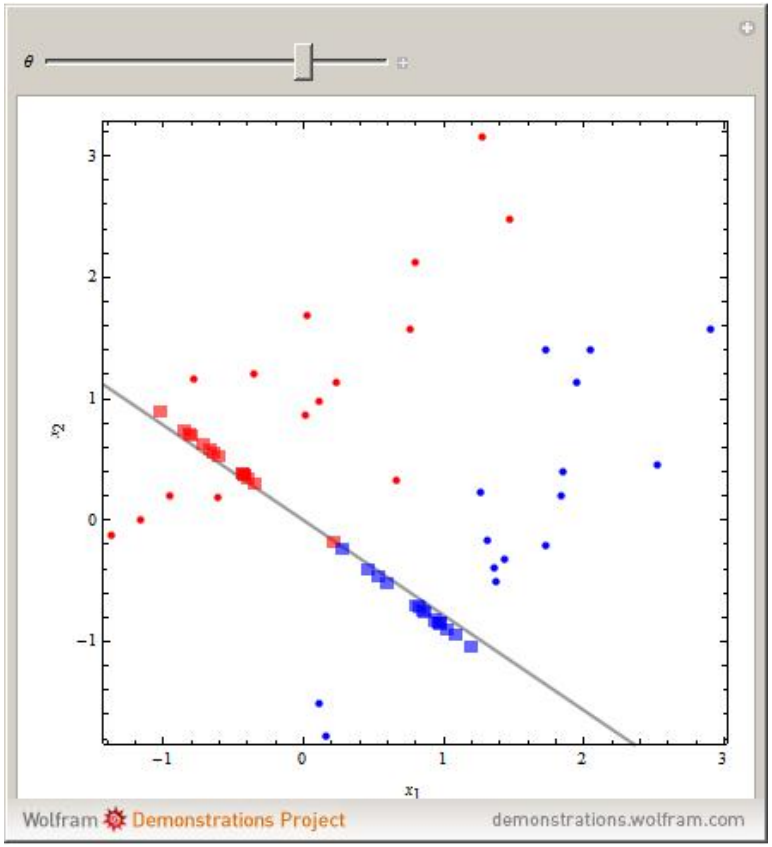




Another illustration - 2

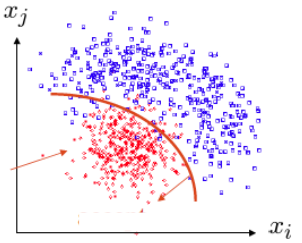






The squares are projected points on a line inclined at the angle θ with respect to the origin. When θ is adjusted so the projected points are aligned with the gray line, the points are maximally separated in the sense that the ratio of between-classes variances to within-classes variance is maximized.

Fisher: comment on least squares



We obtain equivalent separation between hypotheses if we multiply the a_i by a common scale factor and add an arbitrary offset a_0:

$$t(\vec{x}) = a_0 + \sum_{i=1}^{n} a_i x_i$$

Thus we can fix the mean values under the hypotheses H_0 and H_1 to arbitrary values as 0 and 1.

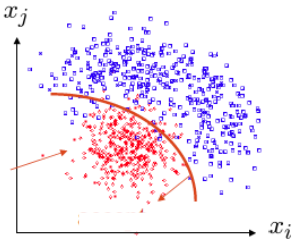
Then maximizing $$J(\vec{a}) = \frac{(\tau_s - \tau_b)^2}{\Sigma_s^2 + \Sigma_b^2}$$

Is equivalent to minimizing

$$\Sigma_0^2 + \Sigma_1^2 = E_0[(t-\tau_0)^2] + E_1[(t-\tau_1)^2]$$

A type of least squares principle !!!

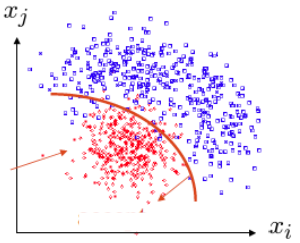
Multivariate methods



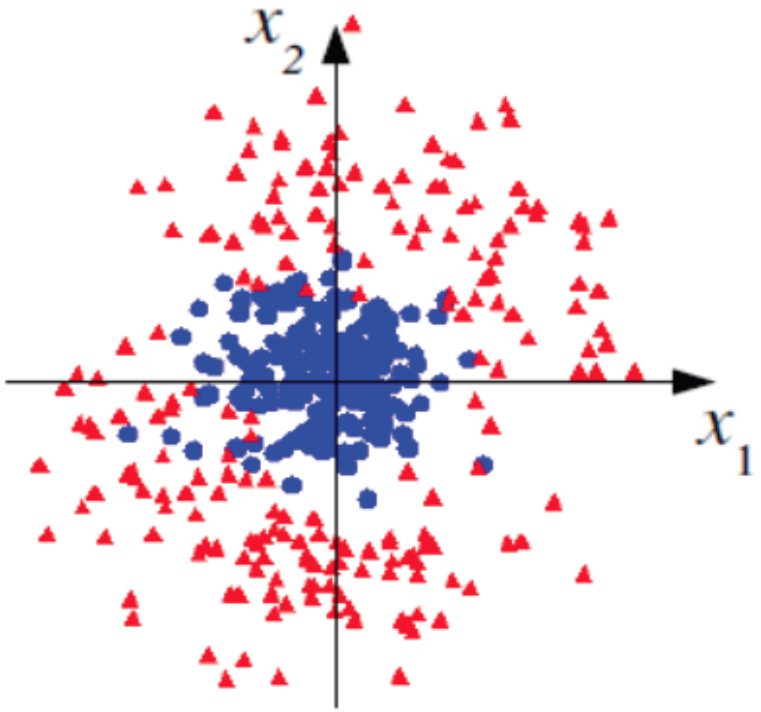
Many new (and some old) methods:

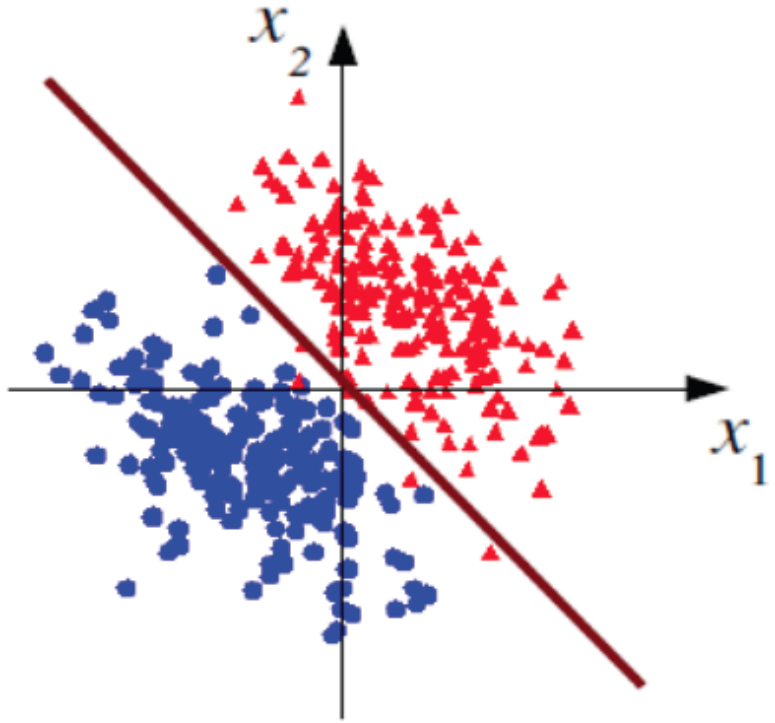
- Fisher discriminant *(linear decision boundary)*
- Neural networks
- Kernel density methods
- Support Vector Machines
- Decision trees:
  - Boosting
  - Bagging
- Toolkit for Multivariate Data Analysis: TMVA
  - Framework for “all” MVA-techniques, available in ROOT

Linear decision boundaries



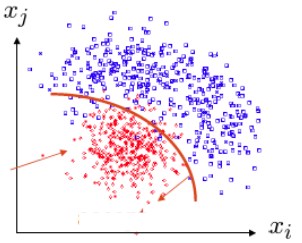
A linear decision boundary is only optimal when both classes follow multivariate Gaussians with equal covariances and different means





For other cases, a linear boundary is almost useless

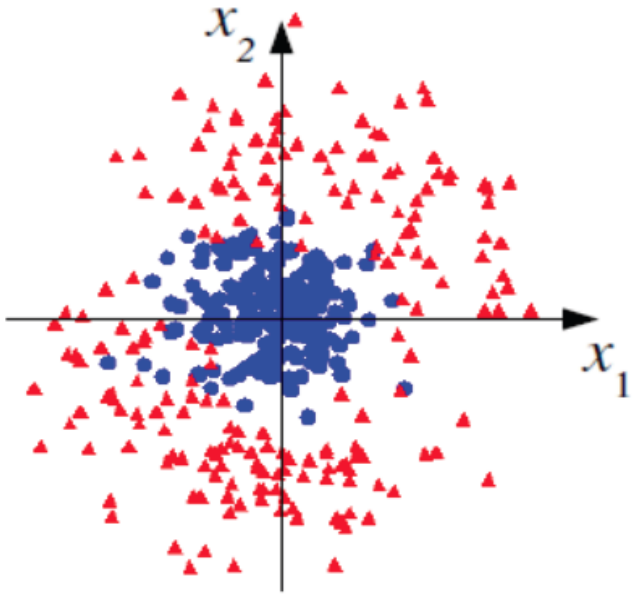
Non-linear transformation of inputs

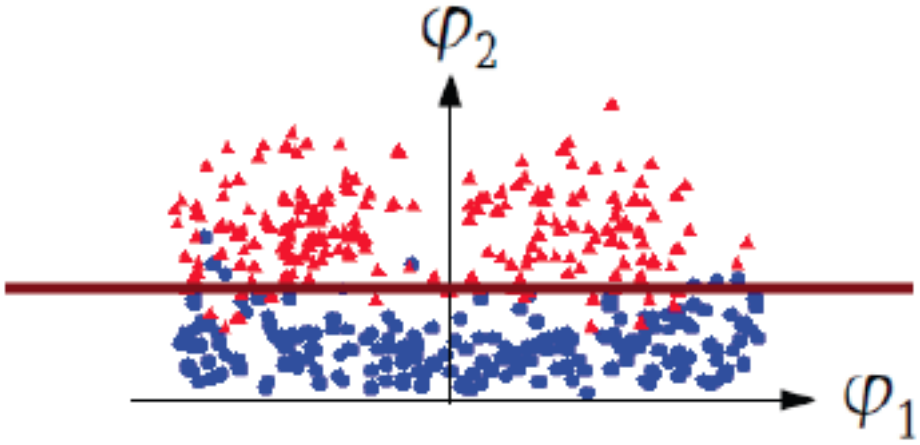


We can try to find a transformation $x_1, \ldots, x_n \rightarrow \phi_1(\vec{x}), \ldots, \phi_m(\vec{x})$
So that the transformed "feature space" variables can be separated better by a linear boundary:

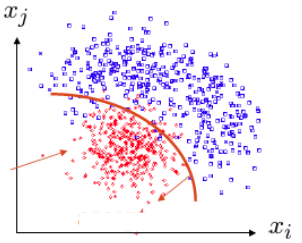
$$\phi_1 = \tan^{-1}(x_2/x_1)$$

$$\phi_2 = \sqrt{x_1^2 + x_2^2}$$





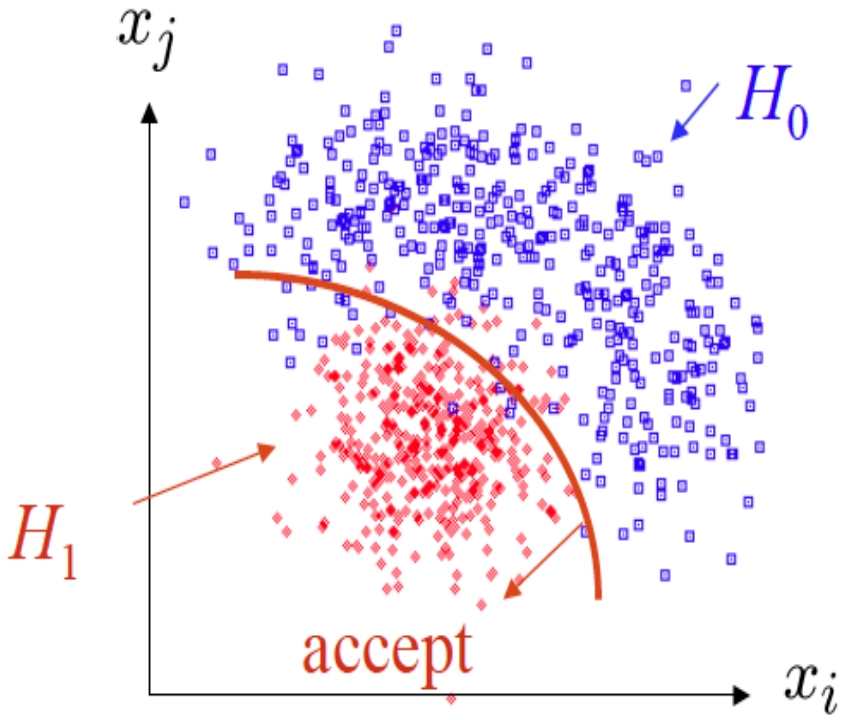
Non-linear test statistic



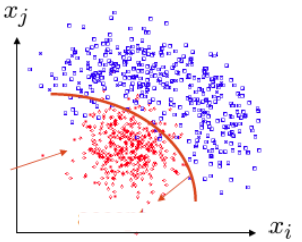
The optimal decision boundary may not be a hyperplane
→ non linear test statistic !!

Many methods of multivariate statistical methods:

- **Neural networks**
- Support vector machines
- Kernel density methods
- Decision trees
- TMVA



Neural networks: introduction

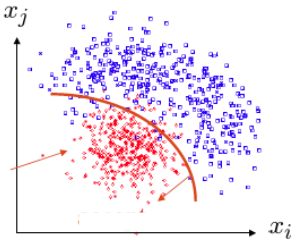


If we want to go to the “arbitrary” non-linear decision boundaries, t(x) needs to be constructed in “any” non-linear fashion

$$t(\vec{x}) = \sum_{i}^{M} (w_i \, h_i(\vec{x}))$$

- Think of $h_i(x)$ as a set of “basis” functions
- If h(x) is sufficiently general (i.e. non linear), a linear combination of “enough” basis functions (M) should allow to describe any possible discriminating function t(x)

Neural networks: single-layer perceptron



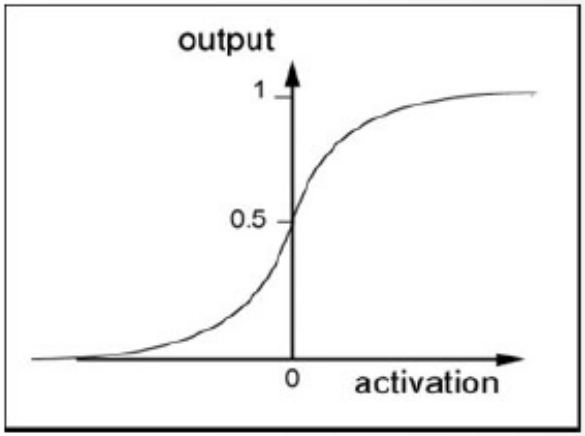
We take the $h_i(x)$ to be such that:

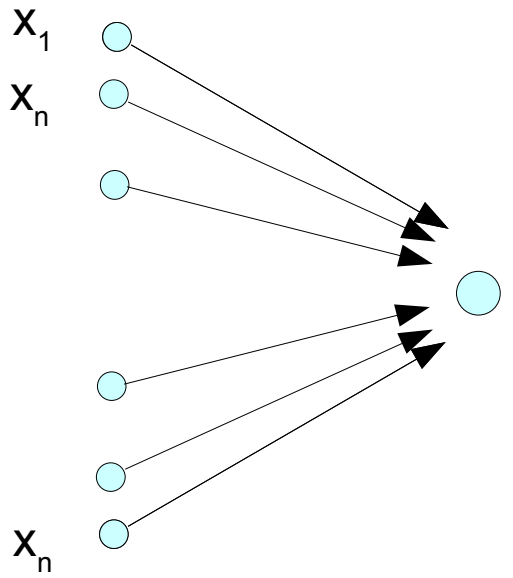
$$t(\vec{x}) = A\left(a_0 + \sum_{i=1}^{n} a_i x_i\right)$$

Where:

- a_0: threshold
- A(x): activation function

$$A(x)=\frac{1}{1-e^{x}}$$: the sigmoid function

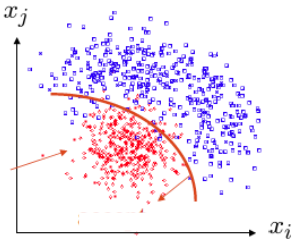




Output node

NODES = input layer

Neural networks: double-layer perceptron



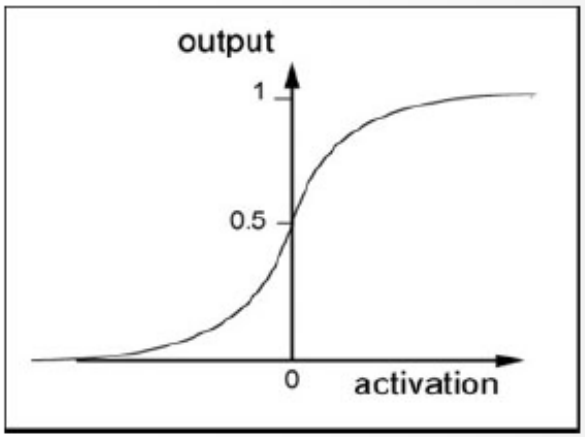
We take the $h_i(x)$ to be such that:

$$t(\vec{x}) = \sum_{i}^{M} w_{0i}\, A\left(w_{i0} + \sum_{j=1}^{n} w_{ij}\, x_j\right)$$

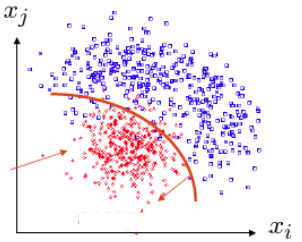
$t(\vec{x})$ is

- A linear combination of
  - non-linear functions of
    - linear combination of
      - the input data

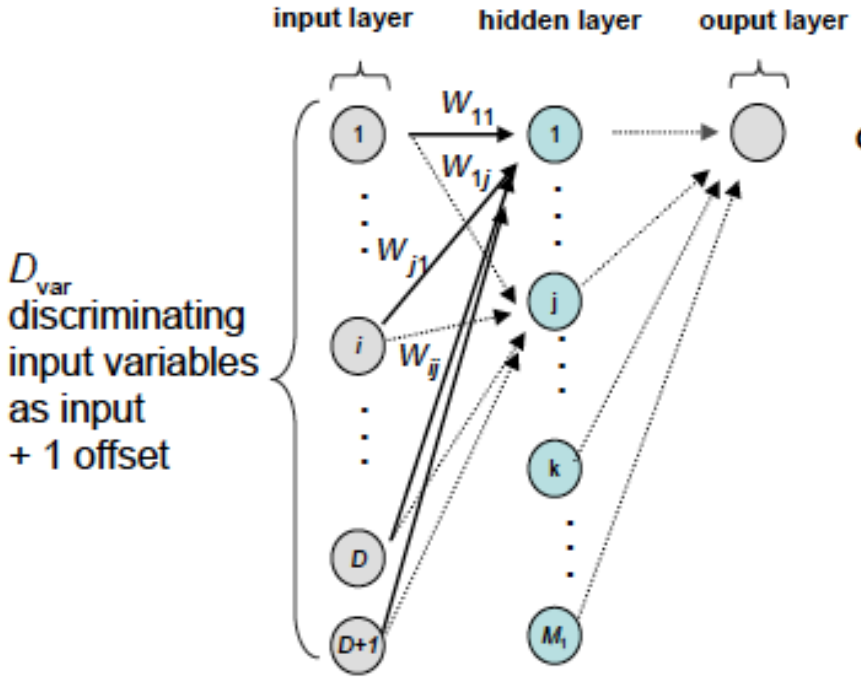
$$A(x)=\frac{1}{1-e^{x}}\text{: the sigmoid function}$$



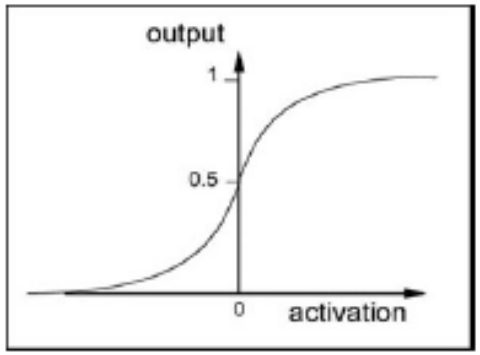
Multilayer perceptron (MLP)



We interpret the formula as a neural network (here D=n)



output: $$y(x)=\sum_{i}^{M} w_{0i} A\left(w_{i0}+\sum_{j=1}^{D} w_{ij}\cdot x_j \right)$$



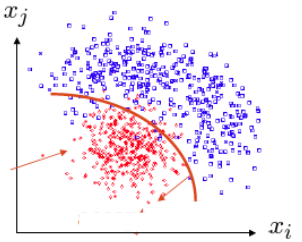
"Activation" function
e.g. sigmoid:

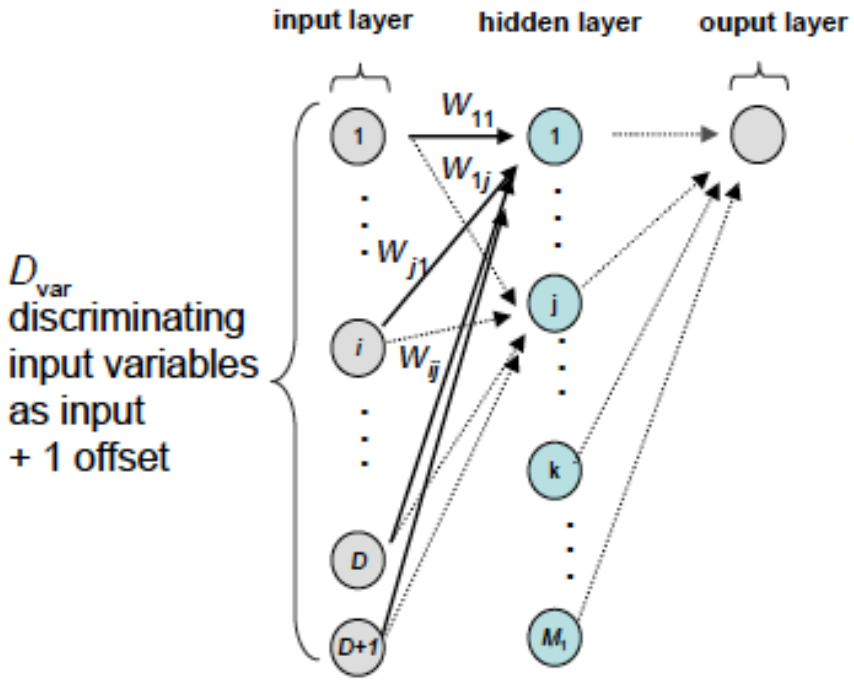
$$A(x)=\left(1+e^{-x}\right)^{-1}$$

or tanh

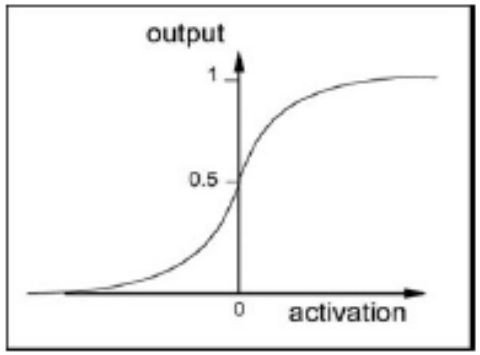
- Nodes in hidden layer represent the "activation functions" whose arguments are linear combinations of input variables → non linear response to the input
- The output is a linear combination of the output of the activation functions at the internal nodes
- Input to the layers from preceding nodes only → feed forward network (no backward loops)
- It is straightforward to extend this to "several" input layers

Multilayer perceptron (MLP)





output: $$y(x)=\sum_{i}^{M} w_{0i} A\left(w_{i0}+\sum_{j=1}^{D} w_{ij} \cdot x_j \right)$$



"Activation" function
e.g. sigmoid:

$$A(x)=\left(1+e^{-x}\right)^{-1}$$

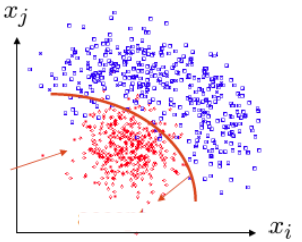
or tanh

Nodes → neurons

Links (weights) → synapses

→ Neural network: try to simulate reactions of a brain to certain stimulus (input data)

Neural network training



Use training events to adjust the weights such that:

- **t(x) → 0 for background events**
- **t(x) → 1 for signal events**

How do we adjust?

Minimize loss function:

$$L(w) = \sum_{i}^{\text{events}} (t(x_i) - t(C))^2$$

Predicted event type

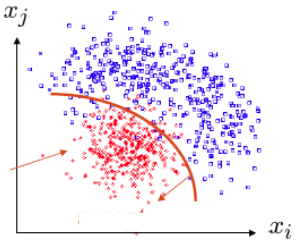
True event type

where:

$$t(C) = \begin{cases} 1 & \text{for } C = \text{signal} \\ 0 & \text{for } C = \text{bkgr} \end{cases}$$

t(x) is a very “wiggly” function with many local minima. A global overall fit in the many parameters is possible but not the most efficient method to train neutral networks ...

Neural network training



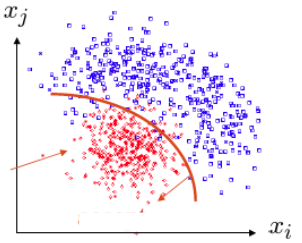
Use smarter methods instead of a global overall fit in the many parameters:

- Back propagation: learn from experience, gradually adjust your perception to match reality
- Online learning: learn event by event and not only at the end of your life from the entire experience

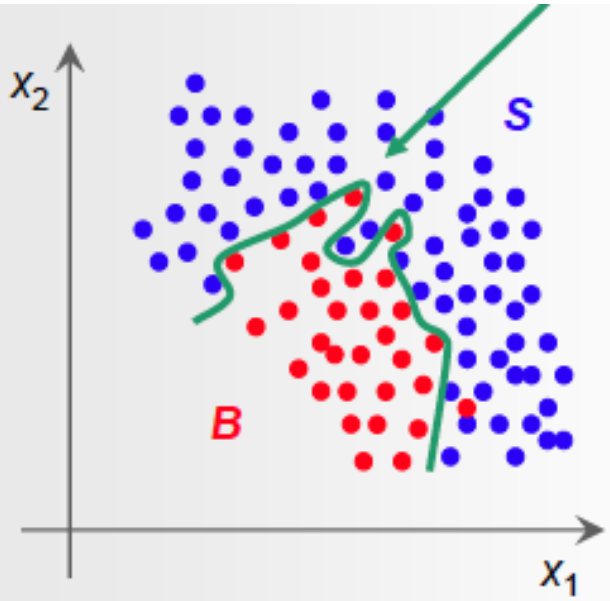
- Start with random weights
- Adjust weights in each step a bit, in the direction of the steepest descent of the loss function
- Training is repeated n times over the whole data sample: HOW OFTEN??

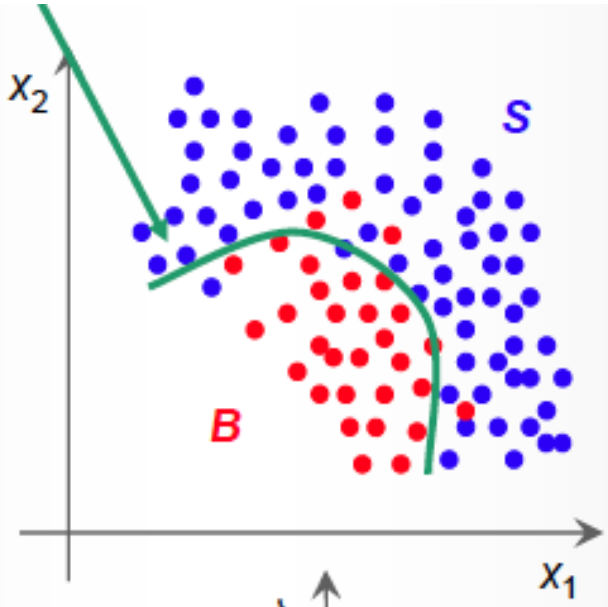
NOTE: for online learning, the training events should be mixed randomly, otherwise you first steer in a wrong direction from which it is afterward hard to get out again !!

Overtraining



Very careful not to OVERDO with the training !!



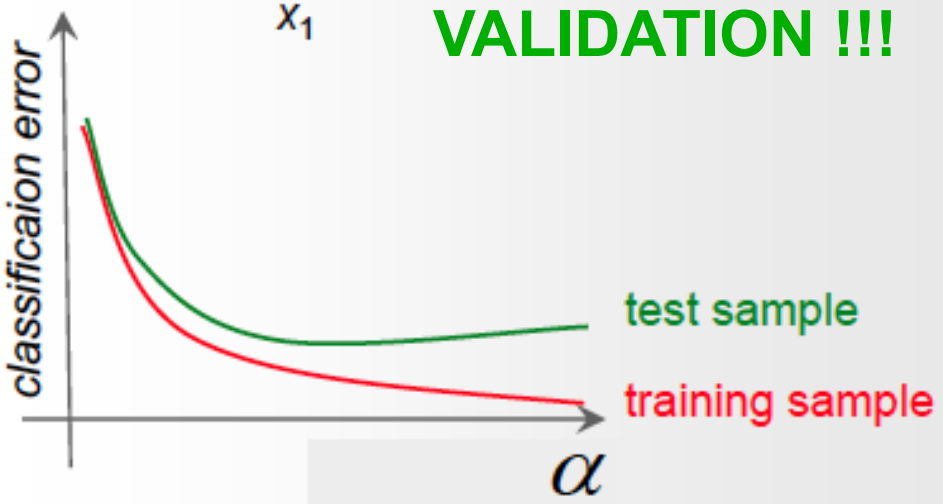


CROSS VALIDATION !!!

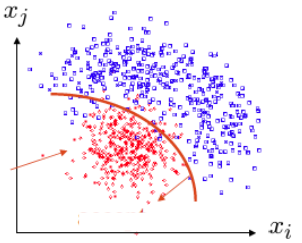
→stop training before you learn statistical fluctuations in the data

→verify on independent "test" sample

→possible overtraining is concern for every "tunable parameter" α of classifiers: Smoothing parameter, n-nodes…



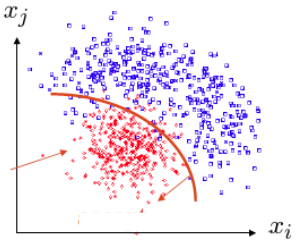
NN: cross validation



- Many (all) classifiers have tuning parameters that need to be controlled against overtraining:
  - Number of training cycles, number of nodes (neural net)
  - Smoothing parameters
  - ...
- The more free parameters a classifiers has to adjust internally → more prone to overtraining
- More training data → better training results
- Divide the data set into “training” and “test” samples (reduces the training data)

Train	Train	Train	Train	Test

What is the best network architecture?



- Theoretically a single hidden layer is enough for any problem, provided one allows for sufficient number of nodes.

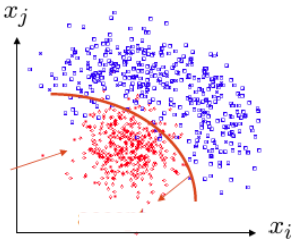
  (K.Weierstrass theorem)

- "Relatively little is known concerning advantages and disadvantages of using a single hidden layer with many nodes over many hidden layers with fewer nodes. The mathematics and approximation theory of the MLP model with more than one hidden layer is not very well understood ……"

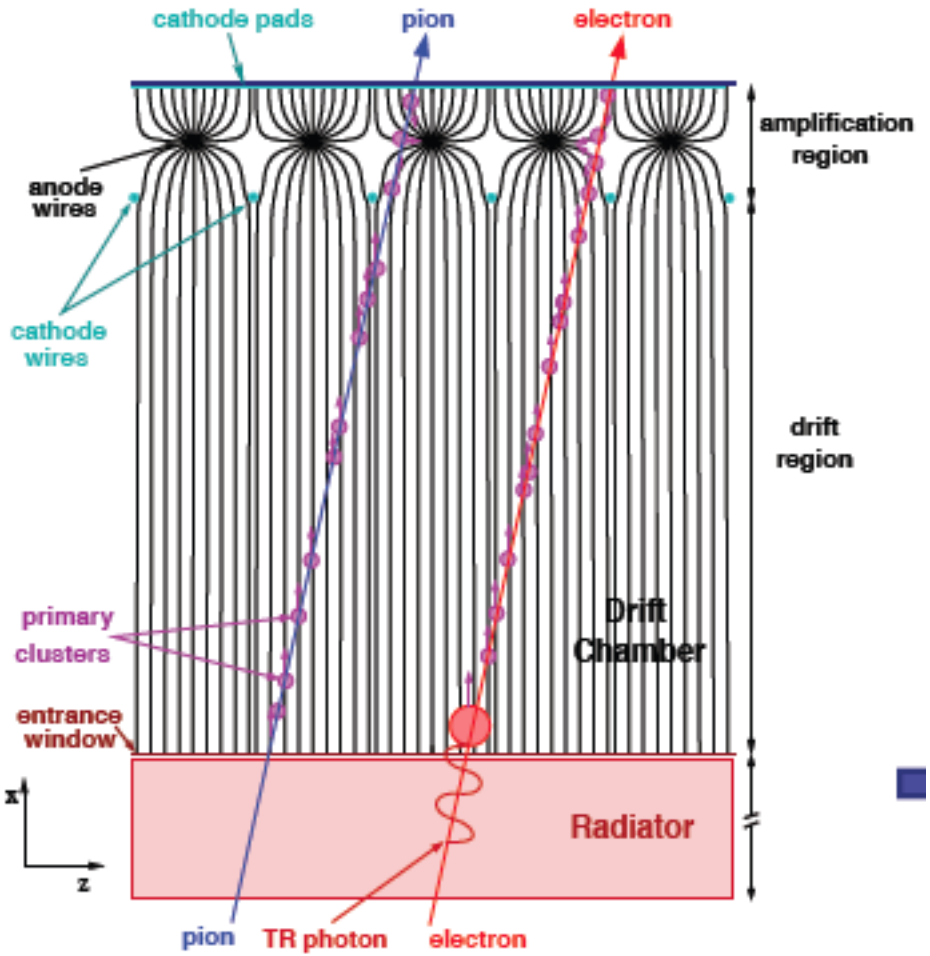
  …."nonetheless there seems to be reason to conjecture that the two hidden layer model may be significantly more promising than the single hidden layer model"

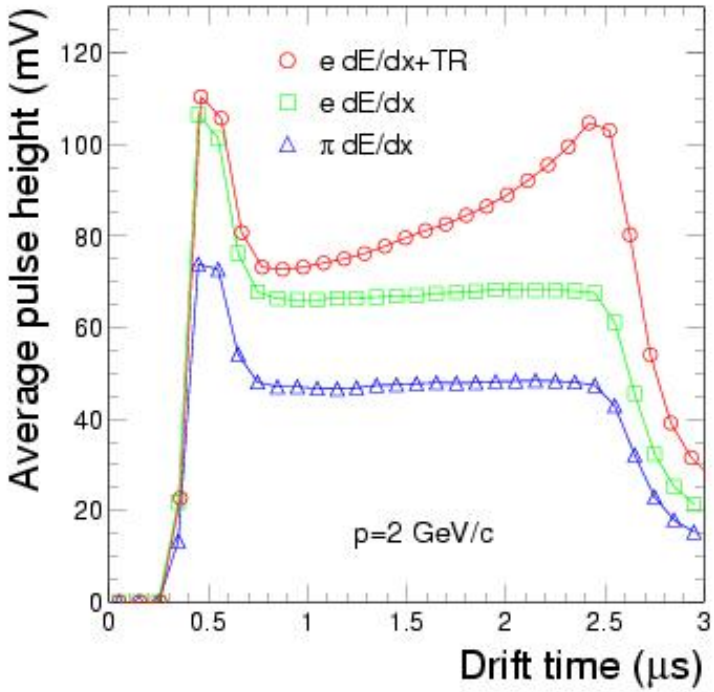
  A.Pinkus, "Approximation theory of the MLP model with neural networks", Acta Numerica (1999), pp. 143-195

Another example

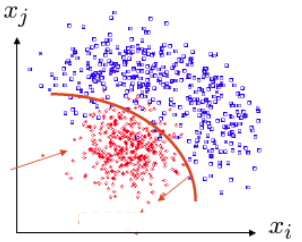


Select electrons from pions using a Transition Radiation Detector (ALICE)

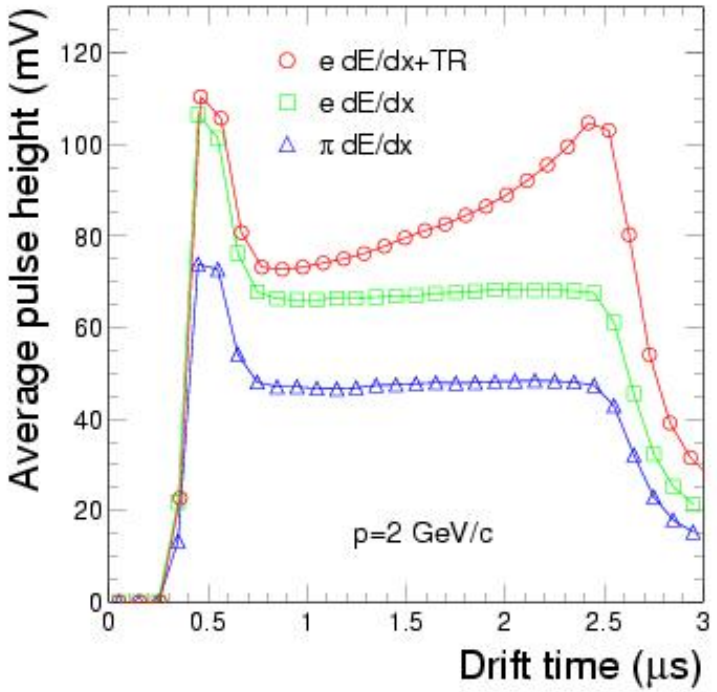




Another example



Select electrons from pions using a Transition Radiation Detector (ALICE)

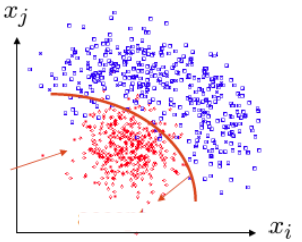


In order to identify individual particles (PID track by track) compare the energy deposit (dE/dx) and signal temporal shape with results from test beam !

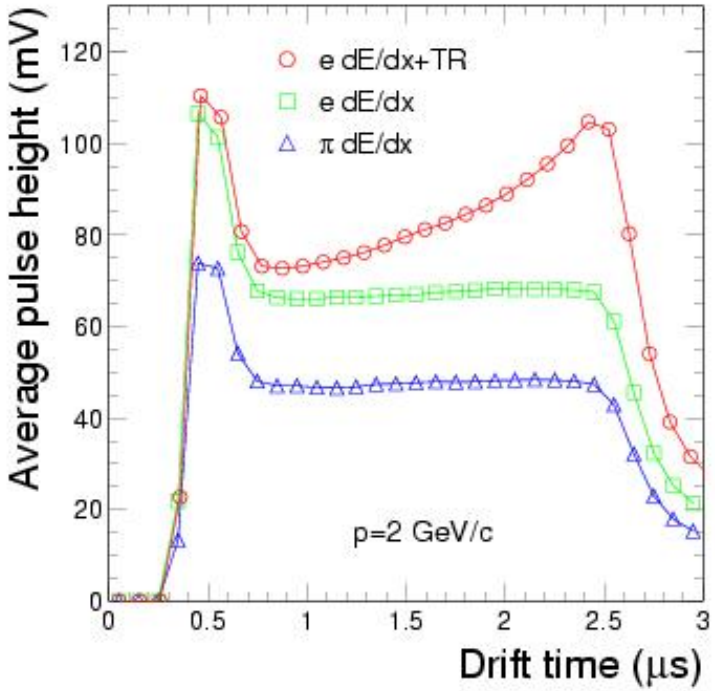
→ in MC the energy deposit in the TRD chambers is not reproduced well enough, cannot use MC for comparison or training

→ use test beams where clean beams of electrons or pions of well defined energy hit the chambers

Another example

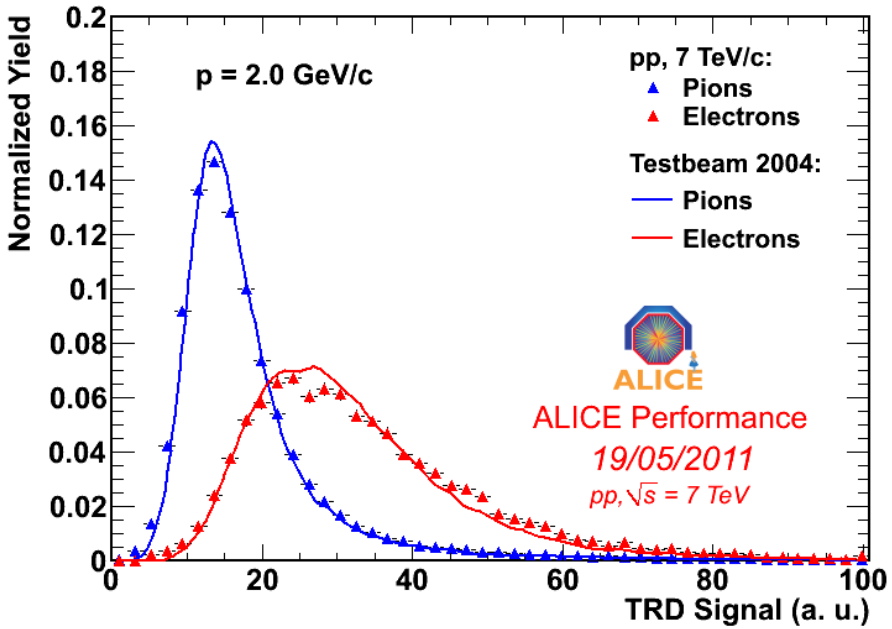


Select electrons from pions using a Transition Radiation Detector (ALICE)

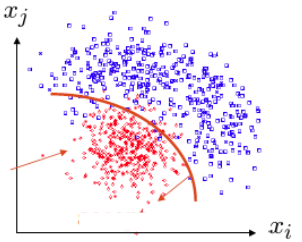


We use different methods to evaluate the PID, with increasing amount of information used:

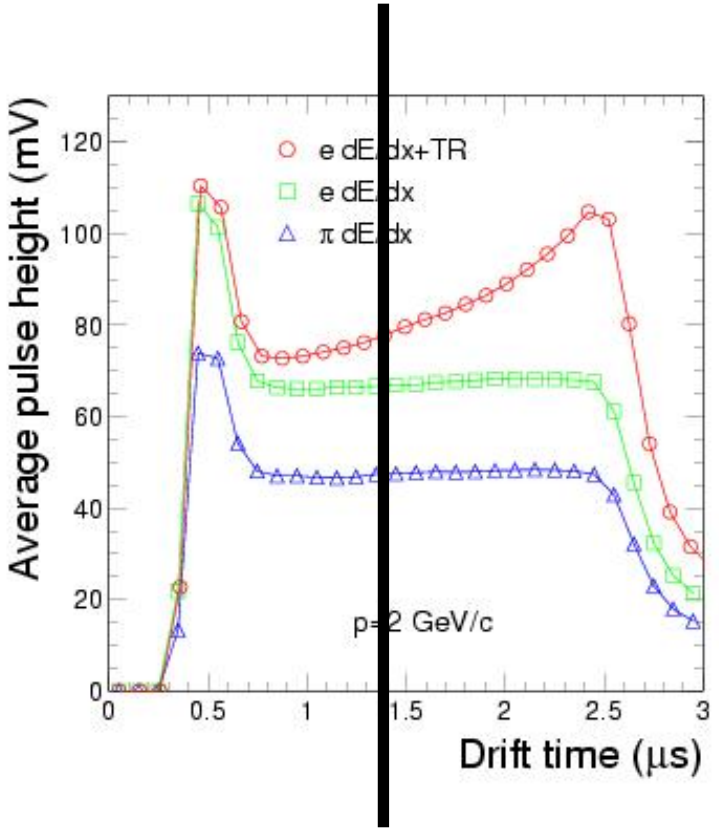
→ 1-dimensional likelihood: only the total charge is compared



Another example



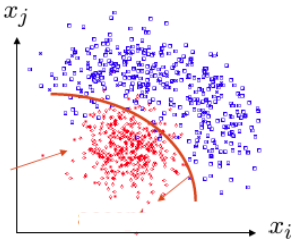
Select electrons from pions using a Transition Radiation Detector (ALICE)



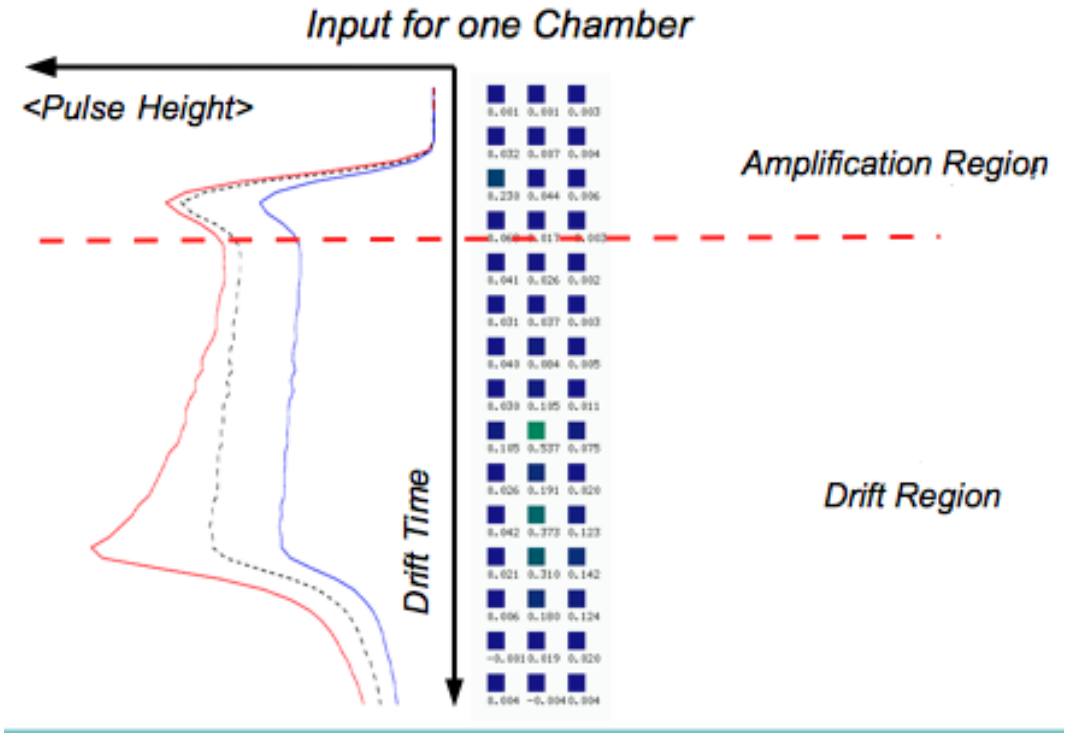
We use different methods to evaluate the PID, with increasing amount of information used:

→ 2-dimensional likelihood:select 2 regions and compare each

Another example



Select electrons from pions using a Transition Radiation Detector (ALICE)

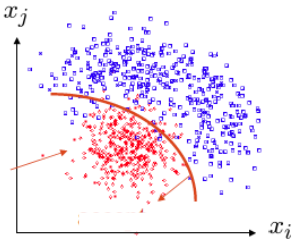


We use different methods to evaluate the PID, with increasing amount of information used:

→ neural network !!!
→ many inputs !!!

More difficult to train (needs more data) but more efficient and more robust!

Next time



Multivariate data analysis methods:

- Fisher discriminant *(linear decision boundary)*
- Neural networks
- Kernel density methods
- Support Vector Machines
- Decision trees:
  - Boosting
  - Bagging
- Toolkit for Multivariate Data Analysis: TMVA
  - Framework for “all” MVA-techniques, available in ROOT