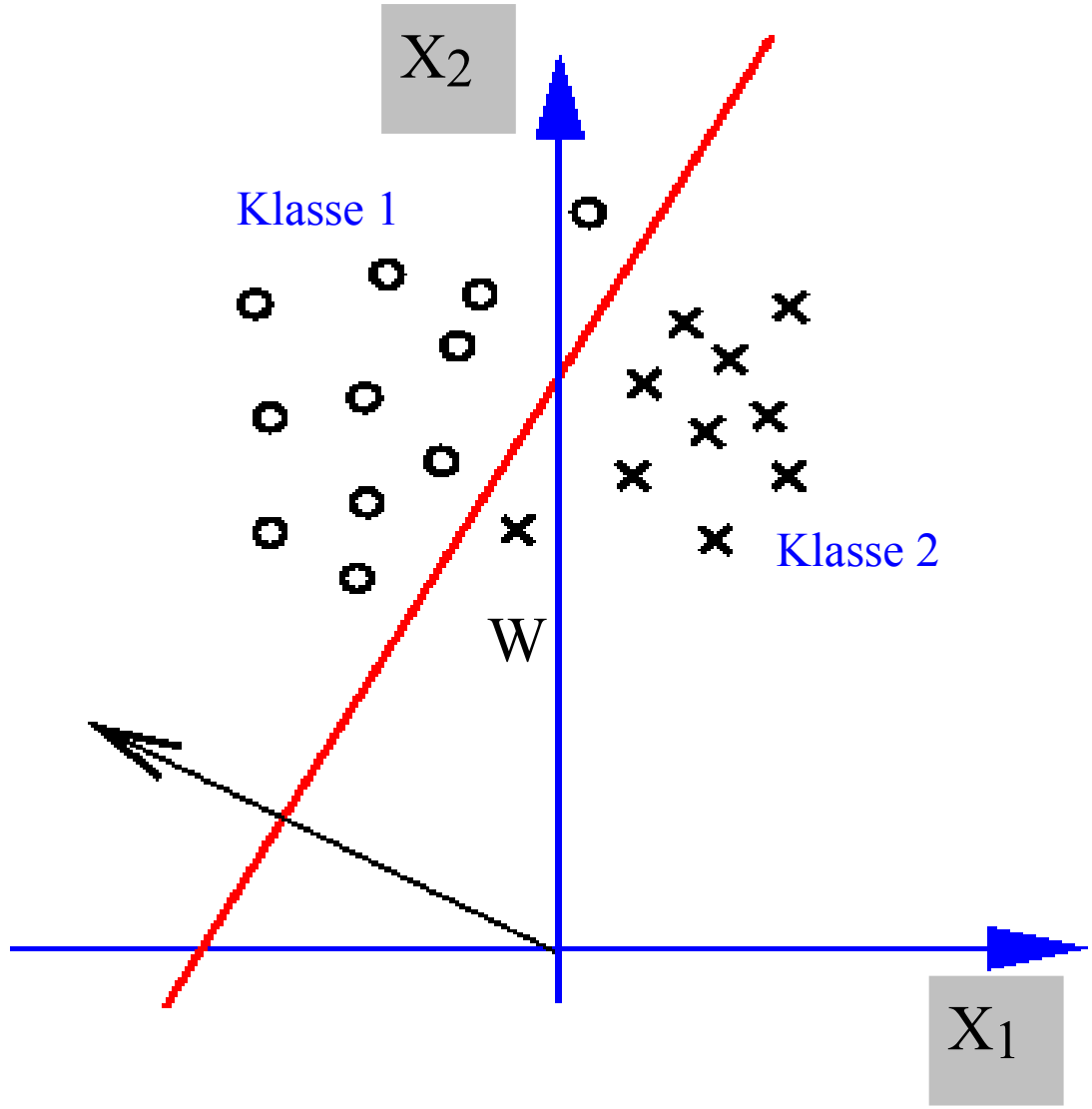
Klassifikation und Neuronale Netze

Hypothesentest/Klassifikation

Klassifizierung

Problemstellung:

gehört ein Punkt (X_1,X_2)
zur Klasse 1 oder 2 ?



Hier einfach: Trennung entlang einer Geraden,
„linear trennbar“

Klassifizierung

Statt einer Trennung im hochdimensionalen Werteraum der Messungen versucht man, eine reelwertige Funktion aller Messungen zu bestimmen, die sogenannte

„Teststatistik“

Beispiele:

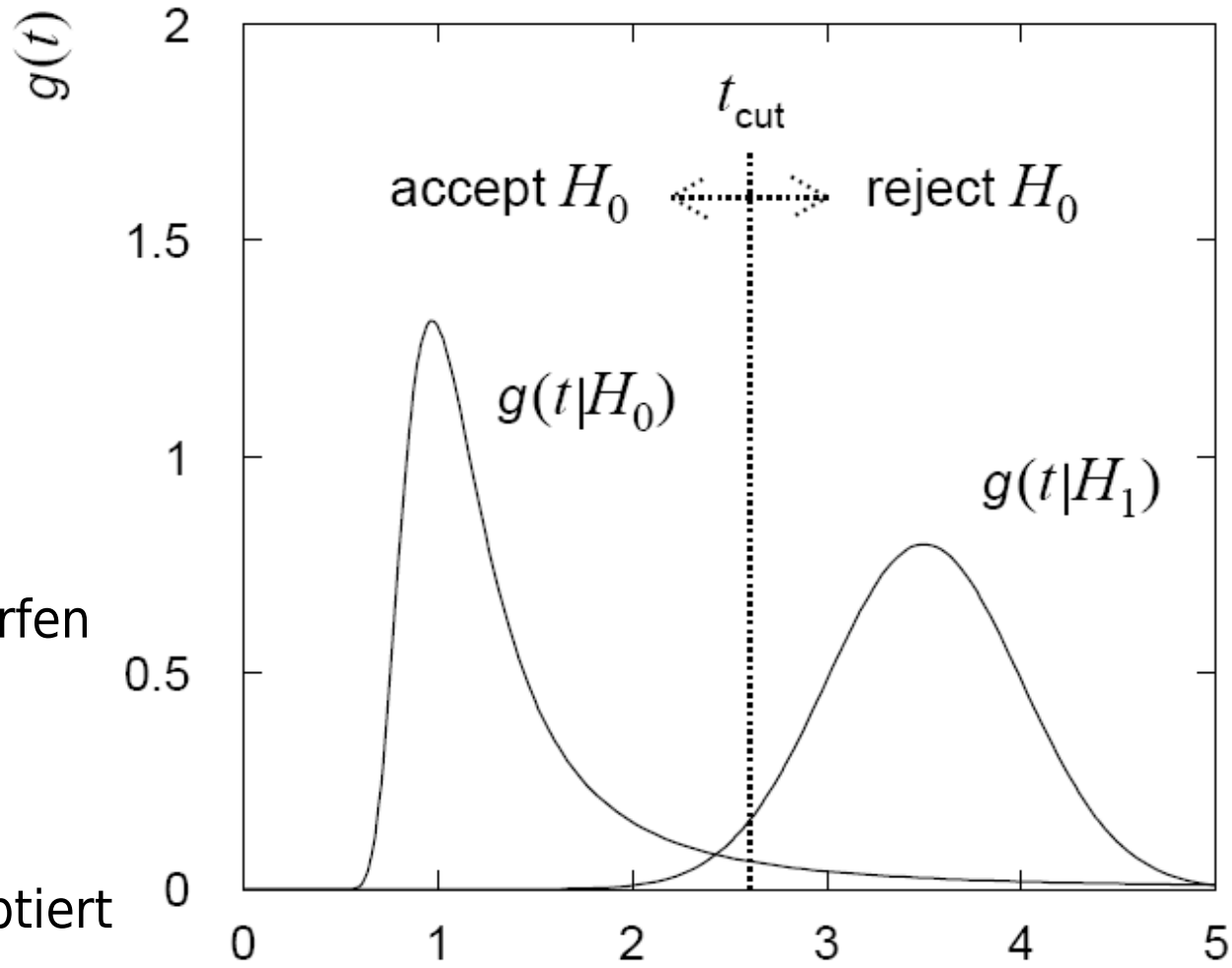
- **Likelihood-Verältnis der Messungen bzgl. der Klassen**
  $Q = L(x_1, \ldots, x_n \mid \text{Klasse 1}) / \ln((x_1, \ldots, x_n \mid \text{Klasse 2})$
- **Verhältnis aus Dekorrelierten Likelihoods**
- **Fisher-Diskriminante: Bestimmung einer Trenn-Hyperebene im Werteraum; für Gaussverteilte x_i analytisch bestimmbar**
- **künstliche neurnale Netzwerke (s. später)**
- **Boosted Decision trees**
- **Support-Vektor-Maschinen**
- **...**

Allgemein: Multivariate Analyse-Verfahren („MVA“)
(Klasse `TMVA` in ROOT)

Hypothesentests

Schnitte in einer **Test-Statistik**:

Akzeptiere Hypothese H0, wenn $t<t_c$



Fehler 1. Art:

wahre Hypothese wird verworfen

Fehler 2. Art:

falsche Hypothese wird akzeptiert

Hypothesentests

Wichtige Größen für alle Klassifikationsverfahren:

Effizienz: $\varepsilon = \frac{P(\text{selektiert und wahr})}{P(\text{wahr})} = 1 - P1$

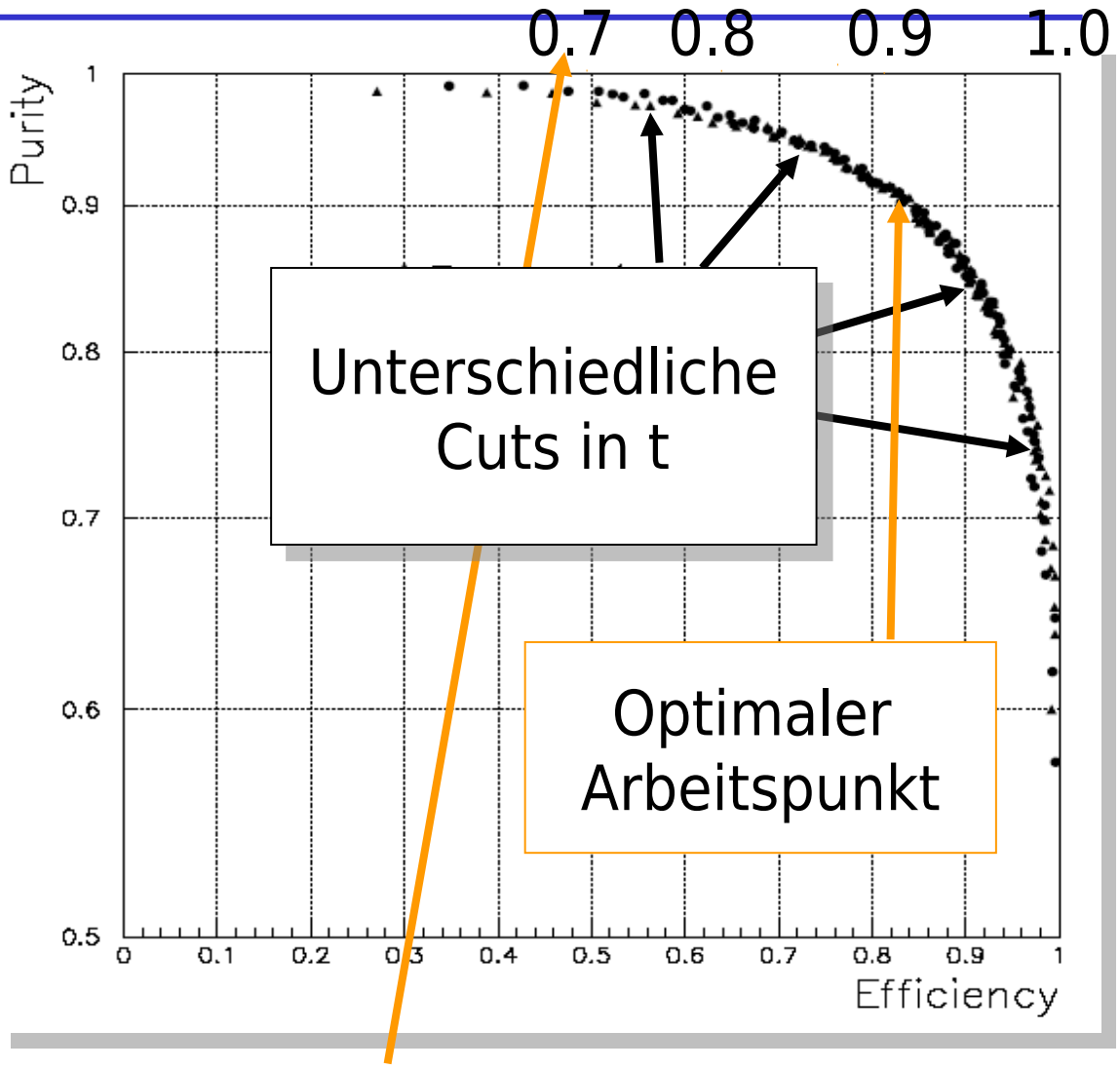
Reinheit (Purity): $\mathcal{P} = \frac{P(\text{selektiert und wahr})}{P(\text{selektiert})} = 1 - P2$

Wahl des Arbeitspunktes (d.h. des Cut-Werts) je nach Anwendung.

Hypothesentests

Ein statistisches Verfahren ist um so besser, desto näher es an den Punkt (1,1) im „Purity-Efficiency-Plot“ kommt

Optimale Wahl des Arbeits-punktes je nach Aufgabe:
Wie skaliert der Gesamtfehler der Analyse mit ε und P?



Konstruktion einer Test-Statistik

Lineare Trennung von korrelierten Input-Variablen durch Hyperebene im n-dim. Raum

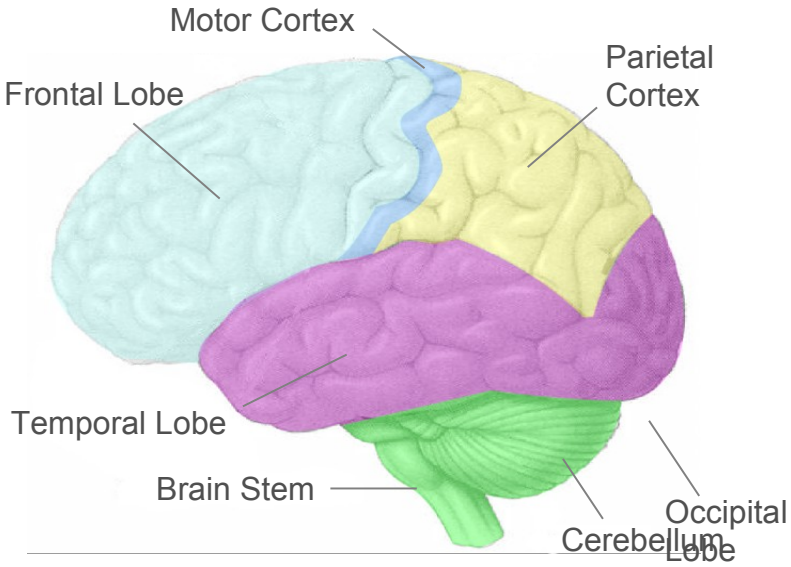
Fischer-Diskriminante:
maximiere Abstand der Mittelwerte der beiden Klassen in Einheiten der Summe ihrer Varianzen

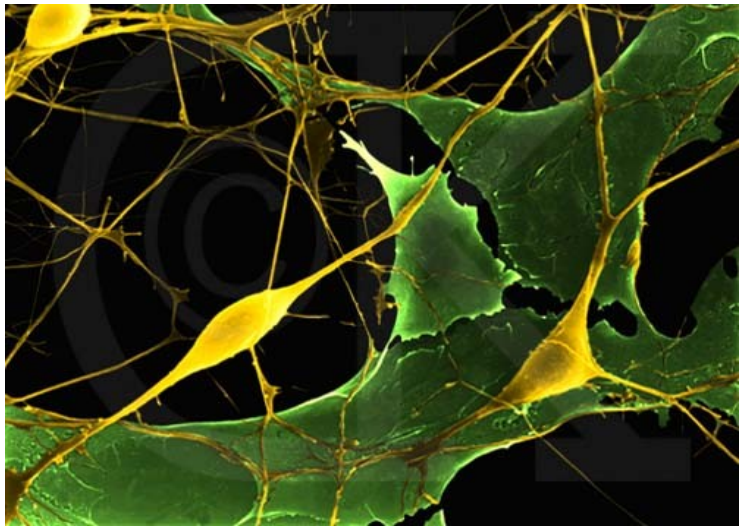
Neyman-Pearson-Lemma: n unkorrelierte Variablen werden optimal durch das Likelihood ratio getrennt:

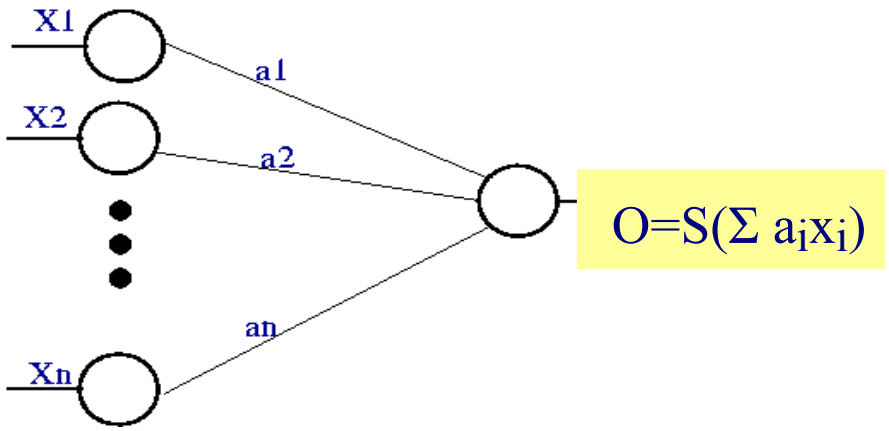
$$t = \sum_{i}^{n} \frac{L_{H_0}(x_i)}{L_{H_1}(x_i)}$$

(Künstliche) Neuronale Netzwerke

Selbstlernende Verfahren, der Natur nachempfunden

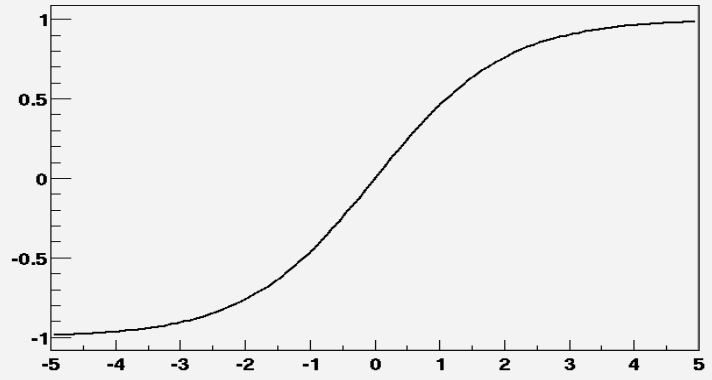




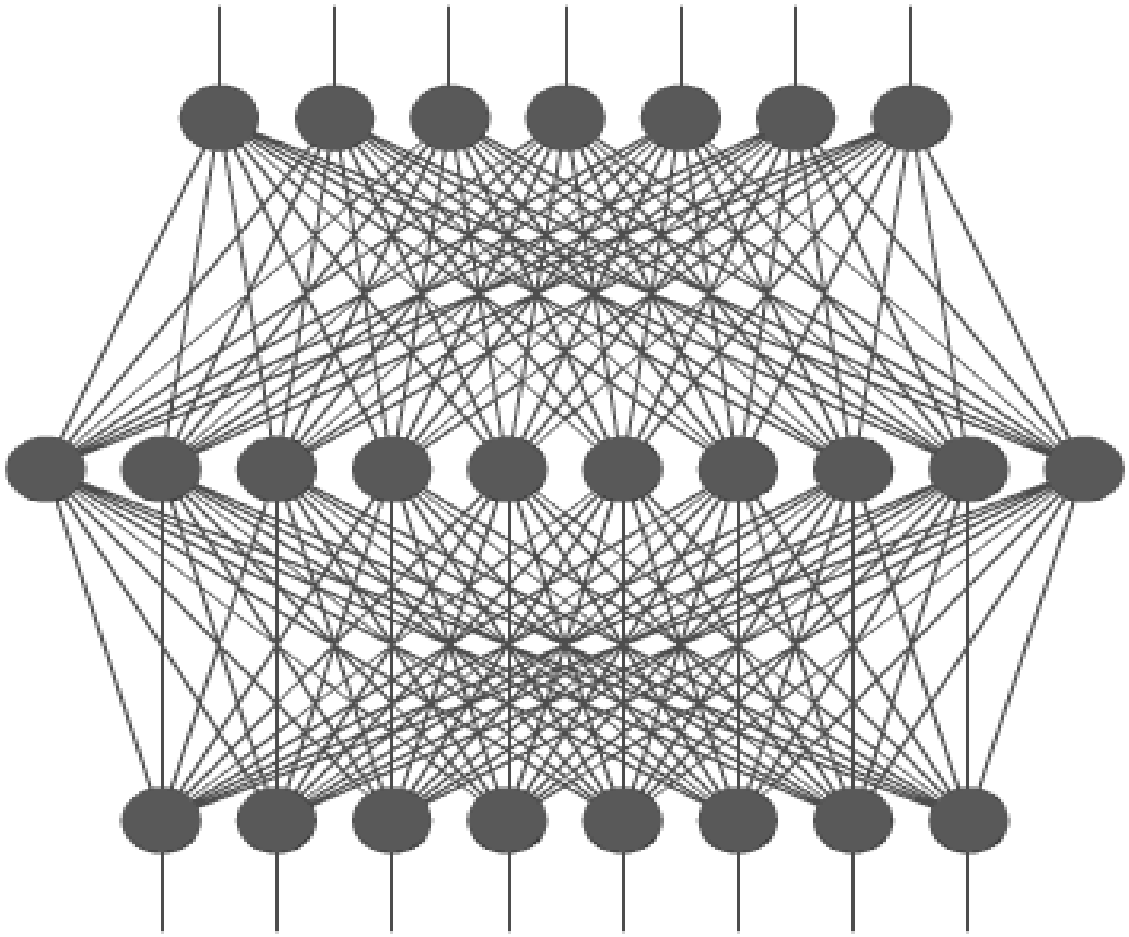


$$O=S(\Sigma\, a_i x_i)$$

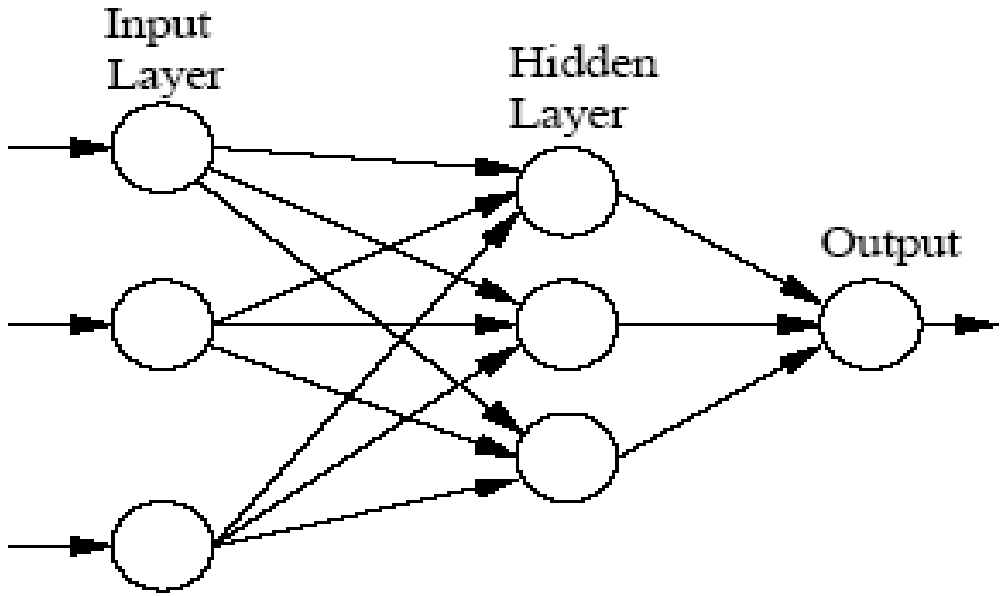
S(x): Aktivierungsfunktion



Die Information
(das Wissen, die Expertise)
steckt in den Verbindungen
zwischen den Nervenzellen



Basisfunktion



Output des Knotens j in Lage n wird aus gewichteter Summe der Outputs aller Knoten der Lage $n-1$ berechnet:

$$x_j^{(n)} = f(\sum_i w_{i,j}^{(n)} x_i^{(n-1)} + w_{0,j}^{(n)})$$

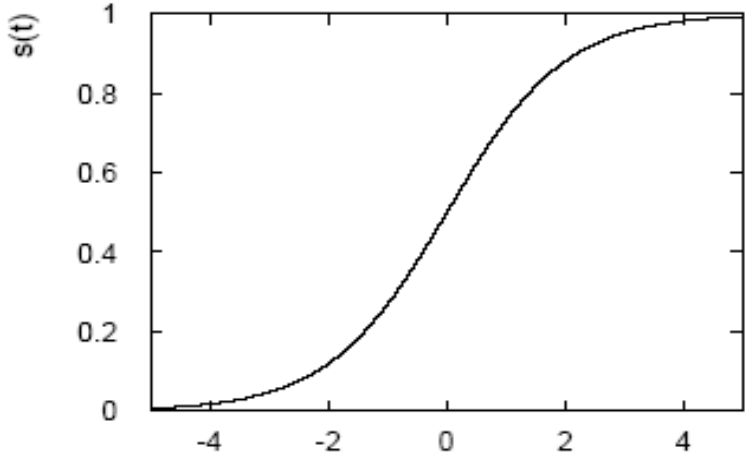
Jeder Verbindung wird ein Gewicht $w_{i,j}^{(n)}$, jedem Knoten ein Bias $w_{0,j}^{(n)}$ zugeordnet.

Neuronale Netzwerke - Transferfunktion

Als Transferfunktion $f(x)$ wird eine nichtlineare monotone Funktion mit eingeschränktem Wertebereich benutzt, z.B. die Sigmoidfunktion:

$$f(x) = \frac{1}{1 + exp(-x)}$$

Sie bildet das Intervall $(-\infty, \infty)$ auf $(0, 1)$ ab.



Neuronale Netzwerke - Training

Als Training bezeichnet man den Minimierungsprozess einer Loss-Funktion, bei dem die Gewichte so angepasst werden, dass die Abweichung vom Soll-Output für ein Set von Input-Vektoren minimiert wird.

Mögliche Loss-Funktionen:
Summe der quadratischen Abweichungen
oder Entropie(Maximum Likelihood)

Neuronale Netze: Bestimmung der Gewichte

Problem gelöst 1986 durch Rumelhart, McClelland u. Williams,
„ Rückwärtspropagation („backpropagation“)“

„Training“ mit Input X bei bekanntem Output T, z.B. aus Monte Carlo

Bestimmung der Gewichte durch iterative
Minimierung der „Fehlerfunktion“, gegeben z.B. durch

$$E(w) = 0.5 * \Sigma (t_p - o_p(w))^2$$

Achtung Gefahr des „Übertrainierens“ bei kleinem Trainingssample,
statistische Fluktuationen werden „auswendig gelernt“.
Besonders gefährlich bei großen Netzen!

Neuronale Netze: Bestimmung der Gewichte (2)

Gewichtsbestimmung ist **Minimierungsproblem**
im (hochdimensionalen) Gewichtsraum;

Probleme wie bei jeder Minimierung:

- Konvergenz in einem Lokalen Minimum statt im globalen Minumum
- u.U. Langsame Konvergenz, starke Oszillationen
- Geeignetes Abbruchkriterium finden

In der Praxis: benutze zwei statistisch unabhängige Datensätze:

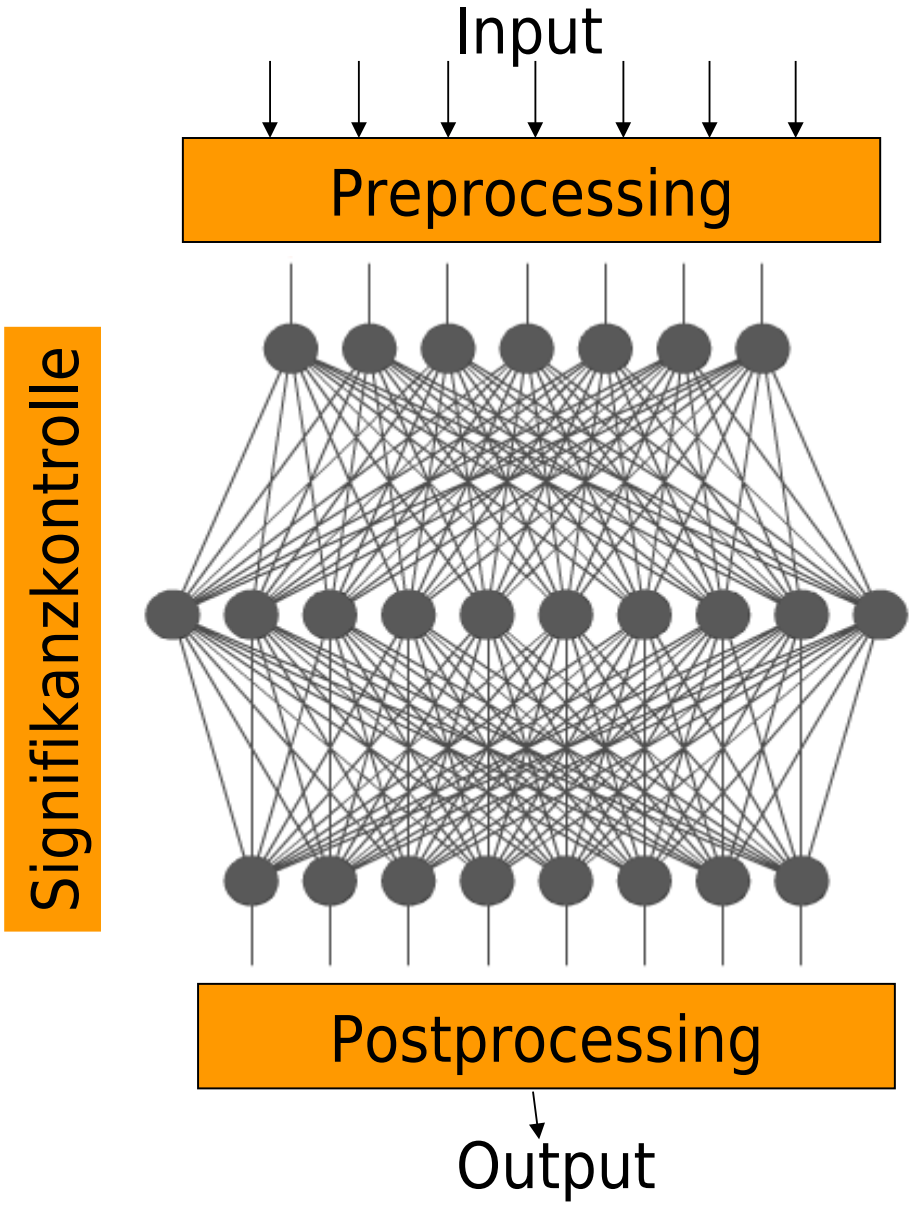
- Trainings-Sample
- Validierungs-Sample

und überwache Lernerfolg am Validierungssample.

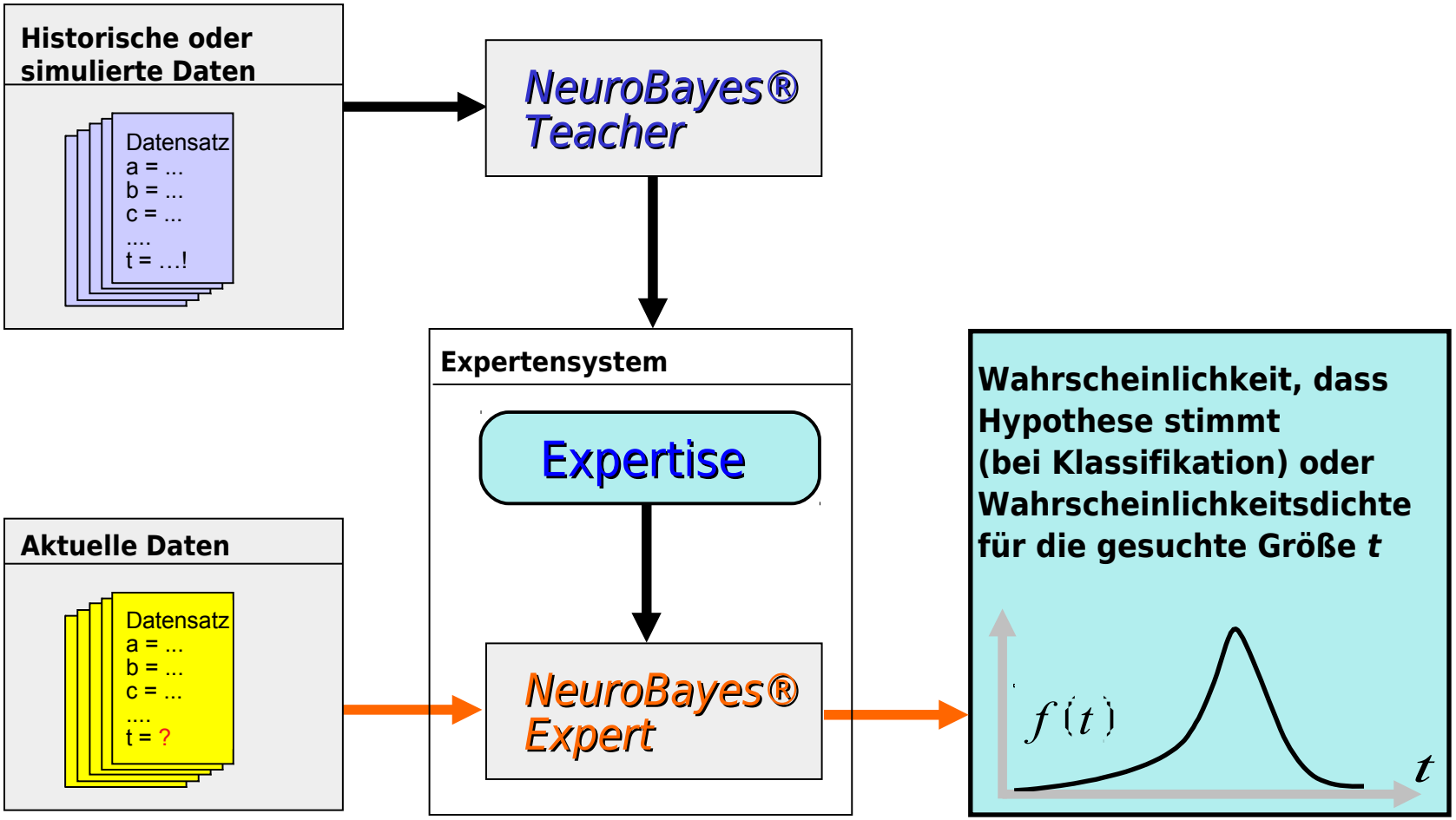
<phi-t>®
Physics Information Technologies
M. Feindt

NeuroBayes® Teacher:
Lernen von komplizierten Zusammenhängen aus bestehenden Datenbanken

NeuroBayes® Expert:
Prognosen für unbekannte Daten

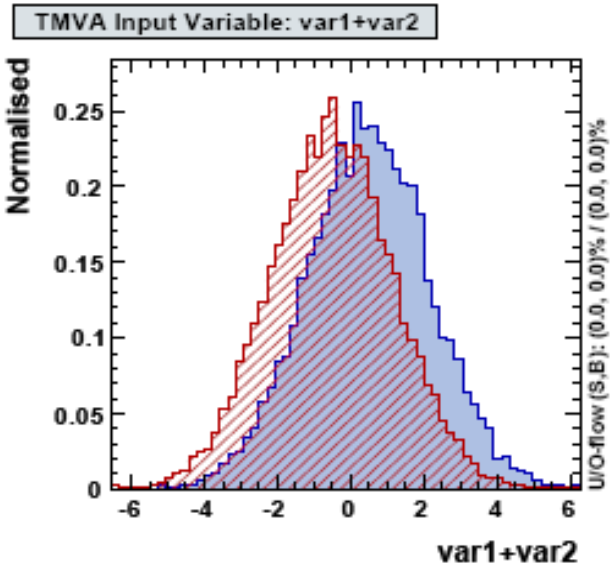


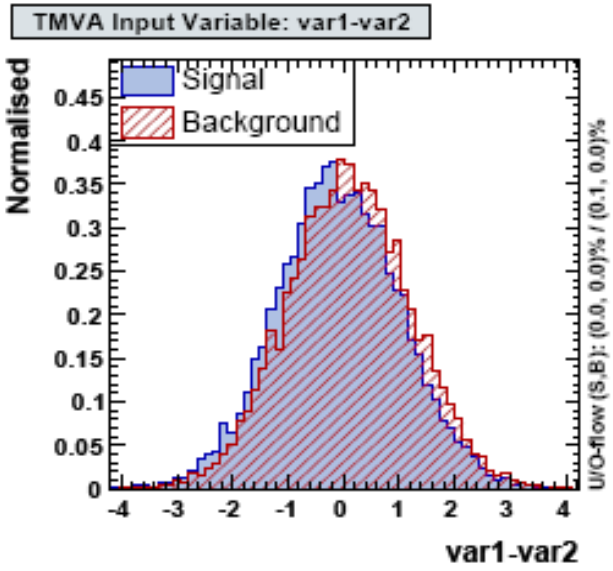
Funktionsweise: Training und Anwendung

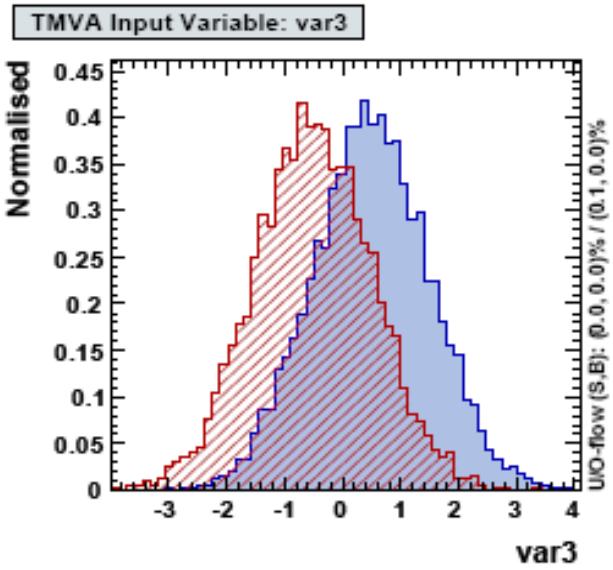


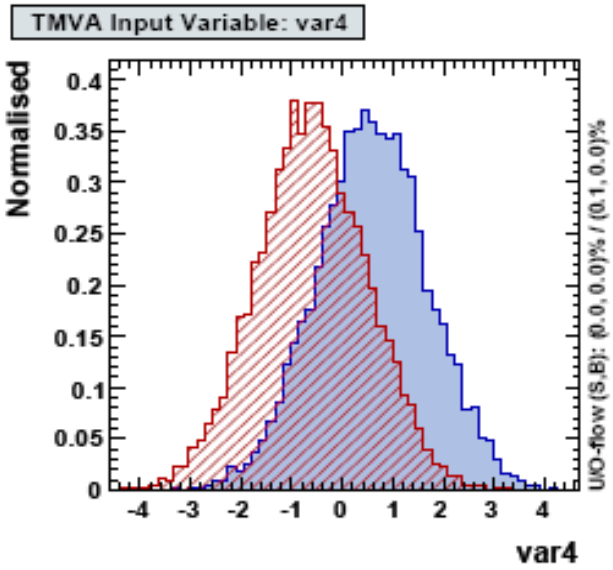
Multivariate Analysemethoden in ROOT:

Multi-layer Perceptron („MLP“) in TMVA



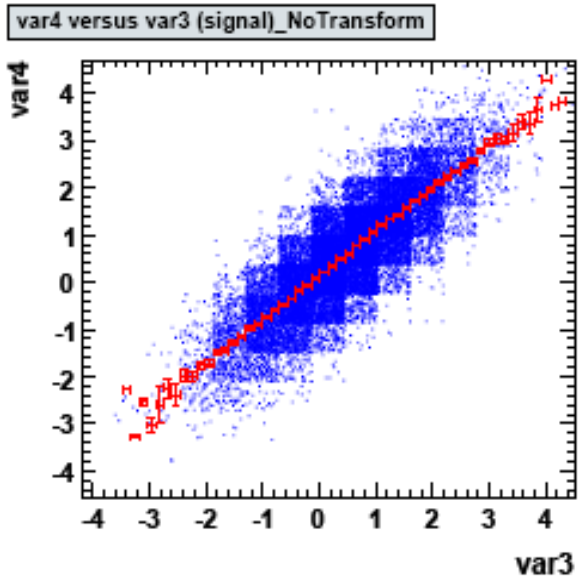






Beispiel

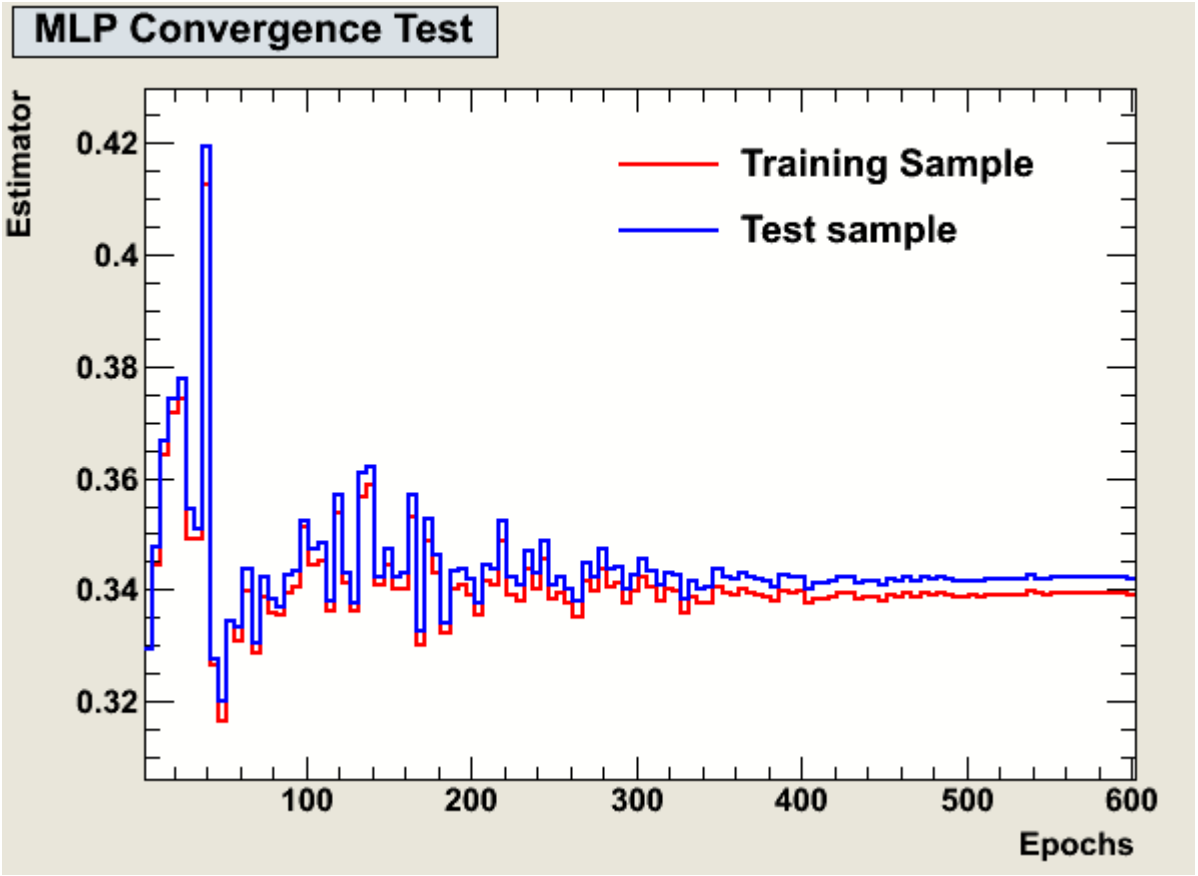
- 4 stark korrelierte Eingangsvariable



- Signal und Untergrund
- Test- und Validierungs-Samples

Multi-layer Perceptron („MLP“) in TMVA (2)

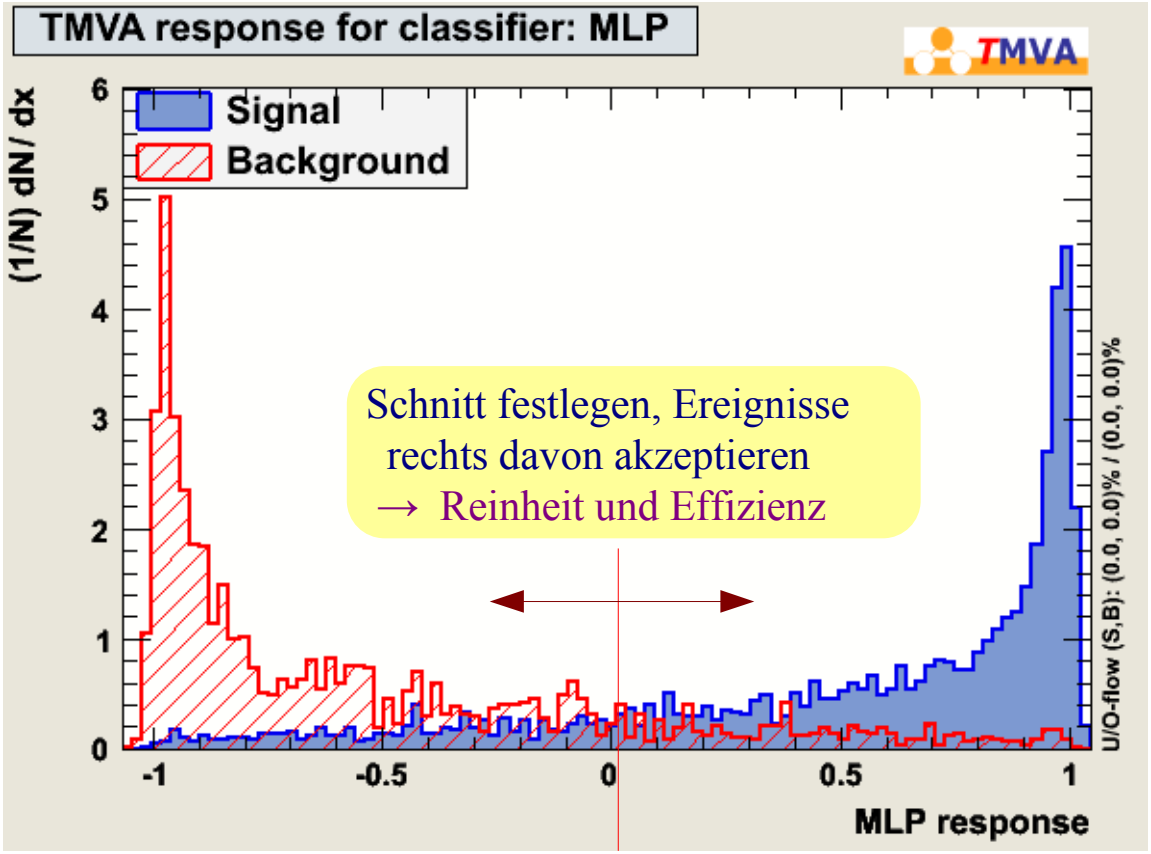
1. Trainingsschritt



Fehler als Funktion des Trainings-Zyklus;
auf Trainings- und Validierungssample

Multi-layer Perceptron („MLP“) in TMVA (3)

2. Analyseschritt



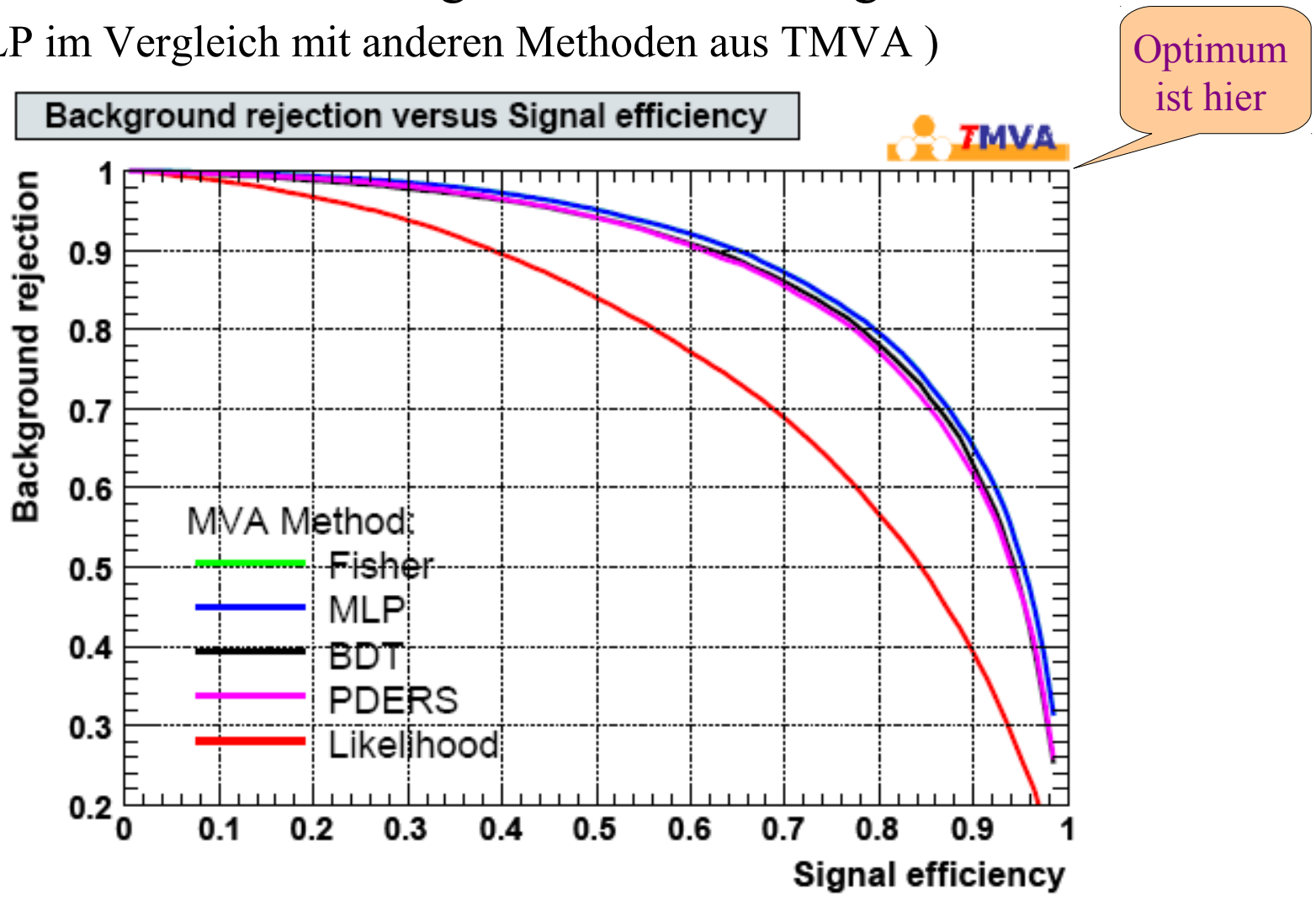
Verteilung des MLP-Outputs für
Signal- und Untergrund auf Validierungssample

Statt vieler Verteilungen für Signal und Untergrund jetzt je eine !

Trennung durch einen Schnitt!

(bzw. aus Netz-Output für jedes Ereignis die Signalwahr-scheinlichkeit berechnen !)

Multi-layer Perceptron („MLP“) in TMVA (4)

Signal-Effizienz vs. Untergrundunterdrückung

(MLP im Vergleich mit anderen Methoden aus TMVA)