Statistics for HEP (3/3)

Diego Tonelli (INFN Trieste)
diego.tonelli@cern.ch

Pre-SUSY2017 School
Dec 8, 2017 - TIFR, Mumbai

“Non-physical” ranges

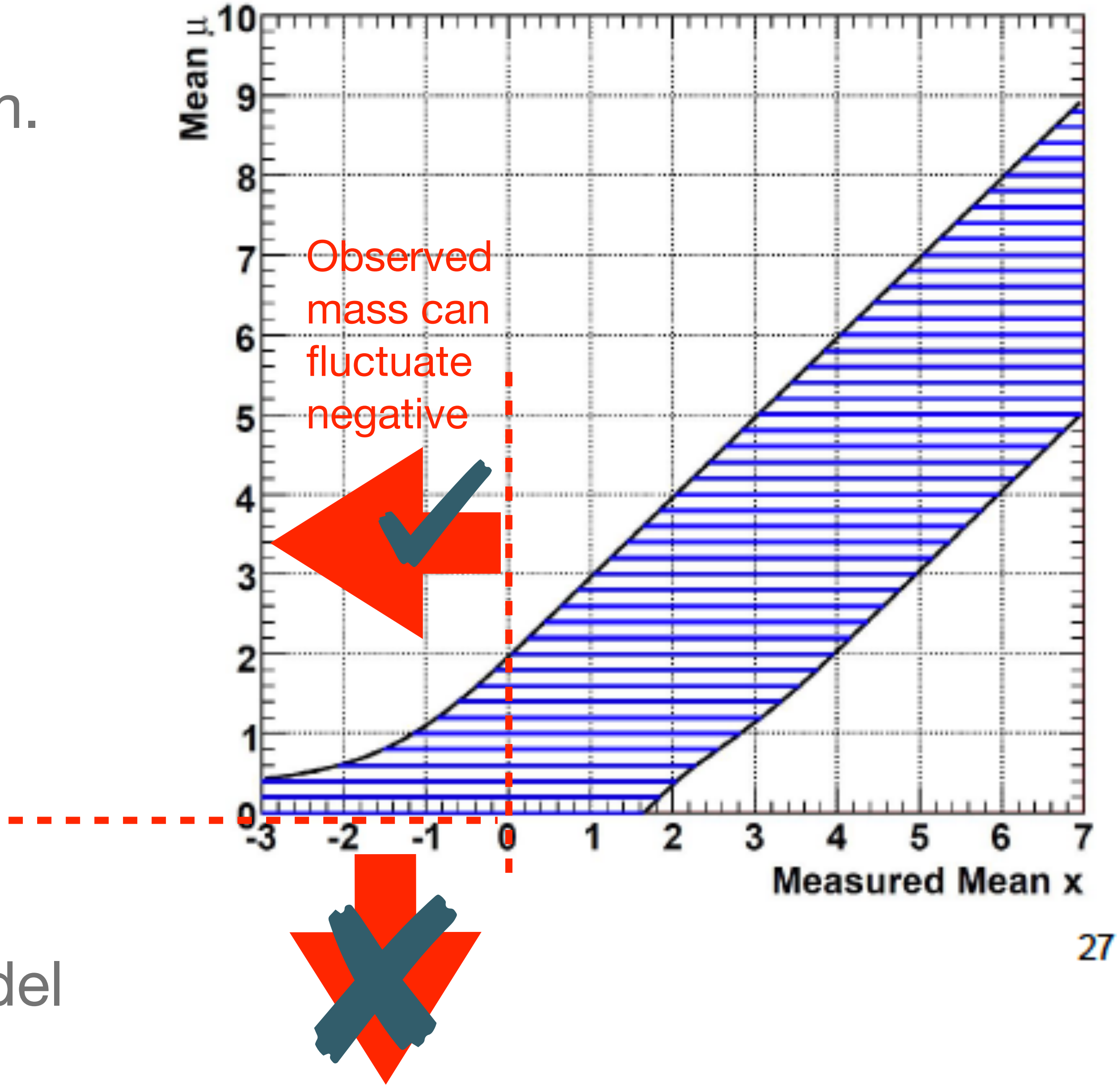
It’s frequent to get confused about the ranges for the confidence band construction.

Example: meaurement of a small mass m. using a Gaussian p(x|m) with x observed mass.

Keep distinct

- data x which, due to resolution, could fluctuate negative

- the mass parameter m, for which negative values do not exist in the model



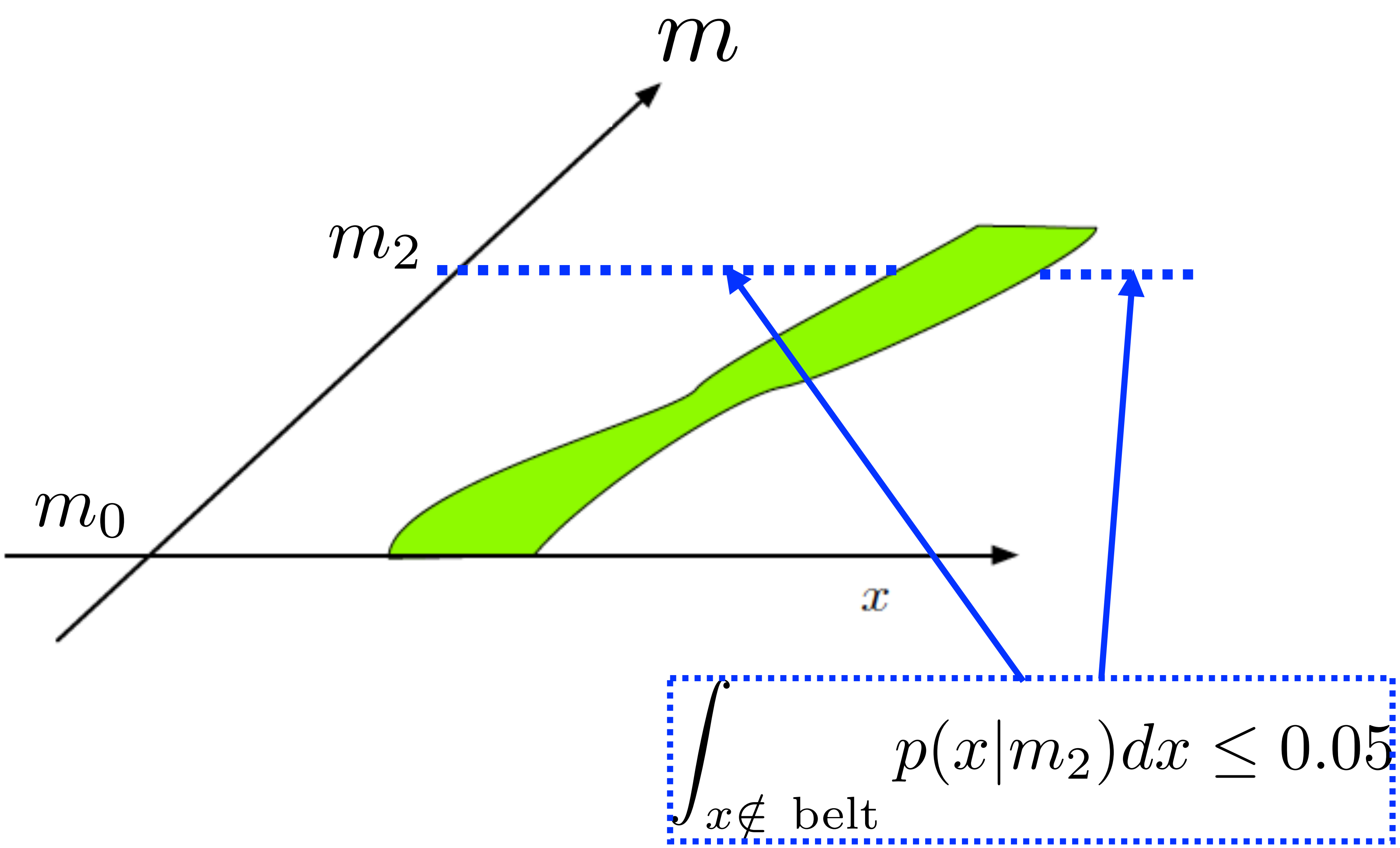
Observed mass can fluctuate negative

Model parameter mass cannot fluctuate negative

[Cousins]

Ordering

The ordering algorithm is arbitrarily chosen, provided that (i) has been defined prior to look at the data (ii) for each value m of the parameter, the integral of the pdf along the x region outside of the belt does not exceed 1-CL.

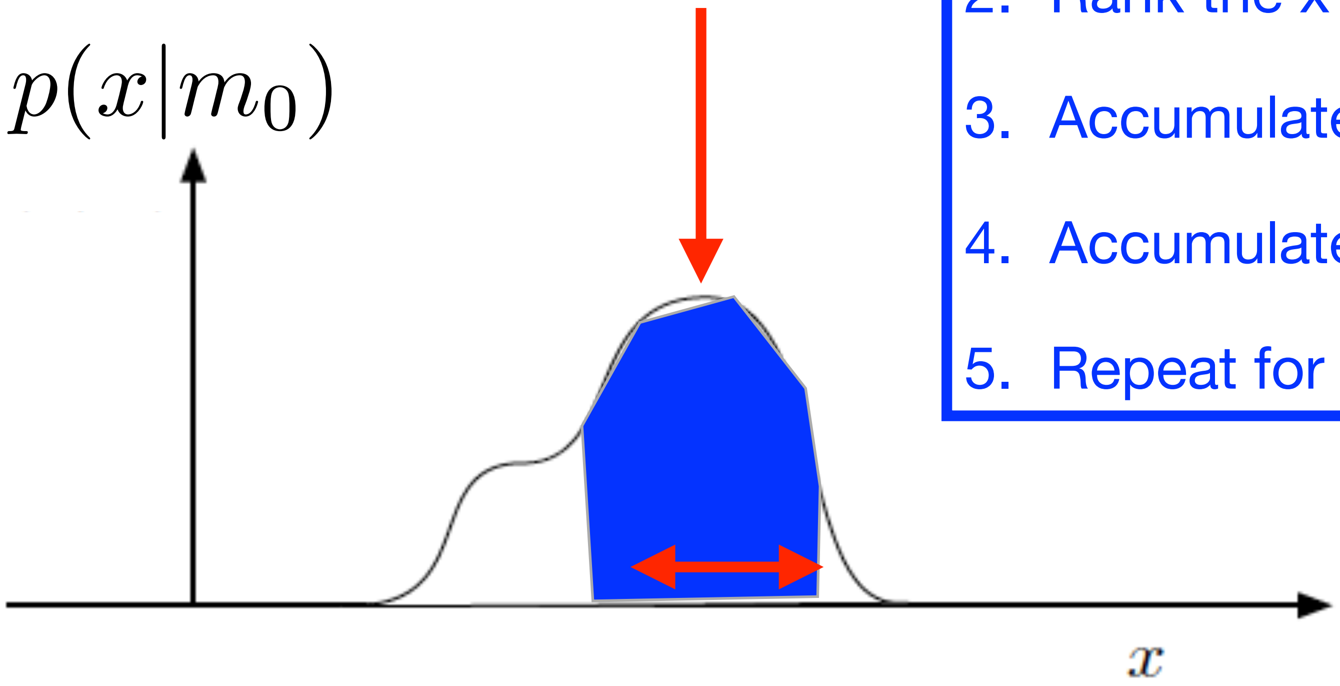


$$\int_{x \notin \text{ belt}} p(x|m_2)dx \leq 0.05$$

[Cranmer]

Probability ordering

In the past, many tried to get the shortest possible interval, so that the resulting confidence intervals were likely narrower yielding more precise measurements. (this is the probability ordering or “Crow-Gardner ordering”)

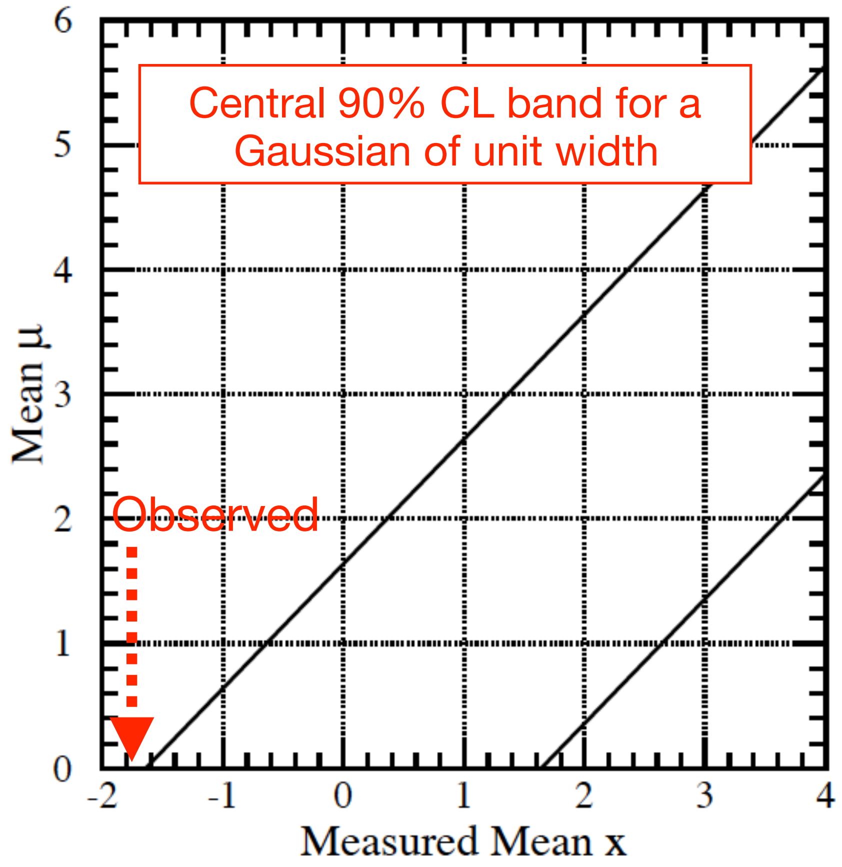


1. Choose one value for m, m_0, and look at $p(x|m_0)$
2. Rank the x values in decreasing order of $p(x|m_0)$
3. Accumulate x starting from the x with highest probability
4. Accumulate all other x until the desired CL is reached.
5. Repeat for all m

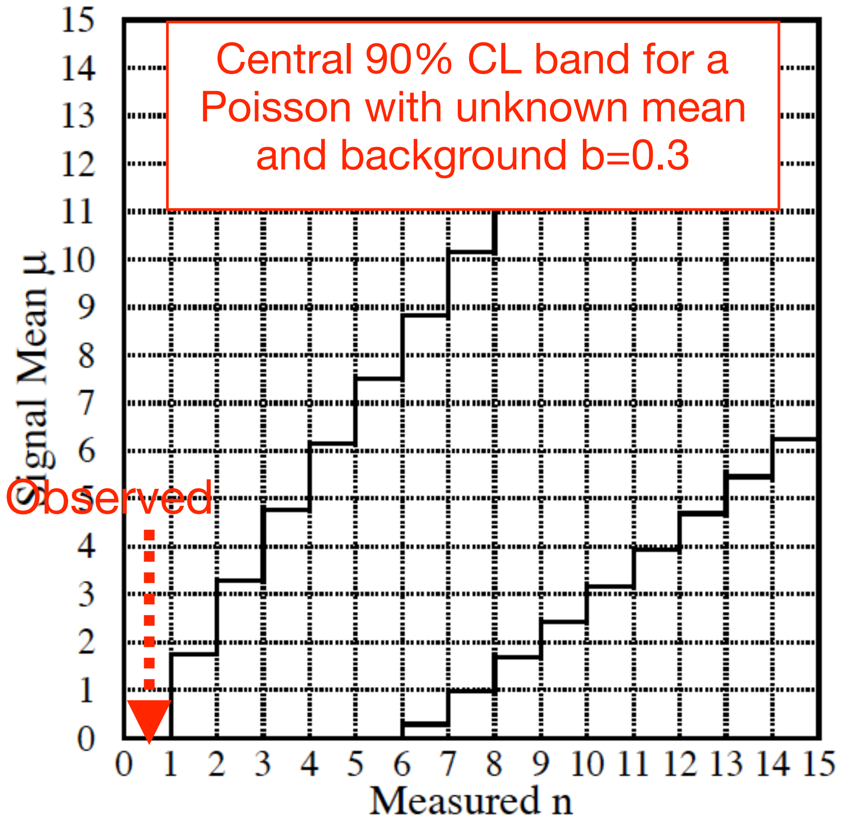
This is ill-defined: as probability depends on the metric for the observable x, the shortest interval in one metric isn’t shortest in others.

Issues

Long-standing inconsistencies found in Neyman constructions based on simplistic ordering criteria (i) Gaussian measurement resolution near a physical boundary (e.g., like a measurement of neutrino mass square close to zero) (ii) measurements of a Poisson signal in the presence of background when observed number of events fluctuates below the expected background count.



What if one observes x = -1.8? or n = 0?



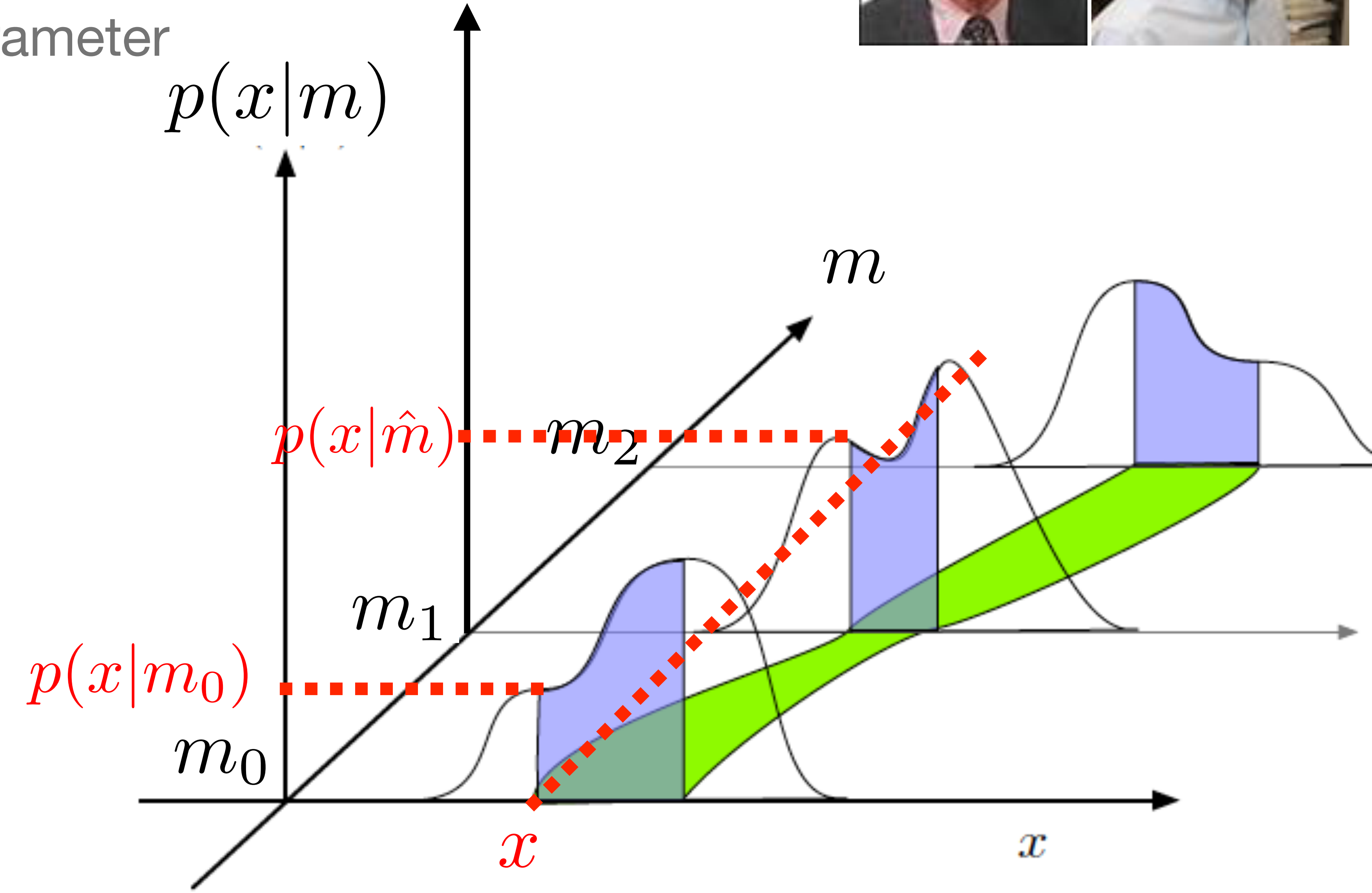
The resulting confidence regions are empty, which is clearly indicative of a problem.

Likelihood-ratio ordering (“Feldman and Cousins”)

Those issues were solved by adapting another ordering, based on the likelihood ratio

Choose a value m₀ of the parameter and for each x calculate

$$\mathrm{LR} = \frac{p(x|m_0)}{p(x|\hat{m})}$$



The “accumulation score” of each element in x, no longer depends only on p(x|m₀) but also on p(x|m) at other m values

[Cranmer]

Likelihood-ratio ordering

1. Choose one value for m, m_0 and generate simulated pseudodata accordingly.
2. For each observation x calculate (i) the value of the likelihood at m_0, $p(x|m_0)=L(m_0)$ and (ii) the maximum likelihood $L(\hat{m})$ over the space of m values (for that observation)
3. Rank all x in decreasing order of likelihood ratio $LR=L_x(m_0)/L_x(\hat{m})$.
4. Accumulate starting from the x with higher LR until the desired CL is reached.
5. Repeat for all m

As the likelihood is metric-invariant so is the ratio of likelihoods. Therefore LR-ordering preserves the metric, mostly avoids empty confidence regions and has several other attractive features. By far the most popular ordering in HEP.

Take LR-ordering as default option unless there are strong motivations against it.

Likelihood-ratio ordering practice

It is instructive to trying to reproduce LR bands as per the original paper. http://arxiv.org/pdf/physics/9711021v2.pdf. Further useful and interesting info in http://users.physics.harvard.edu/~feldman/Journeys.pdf

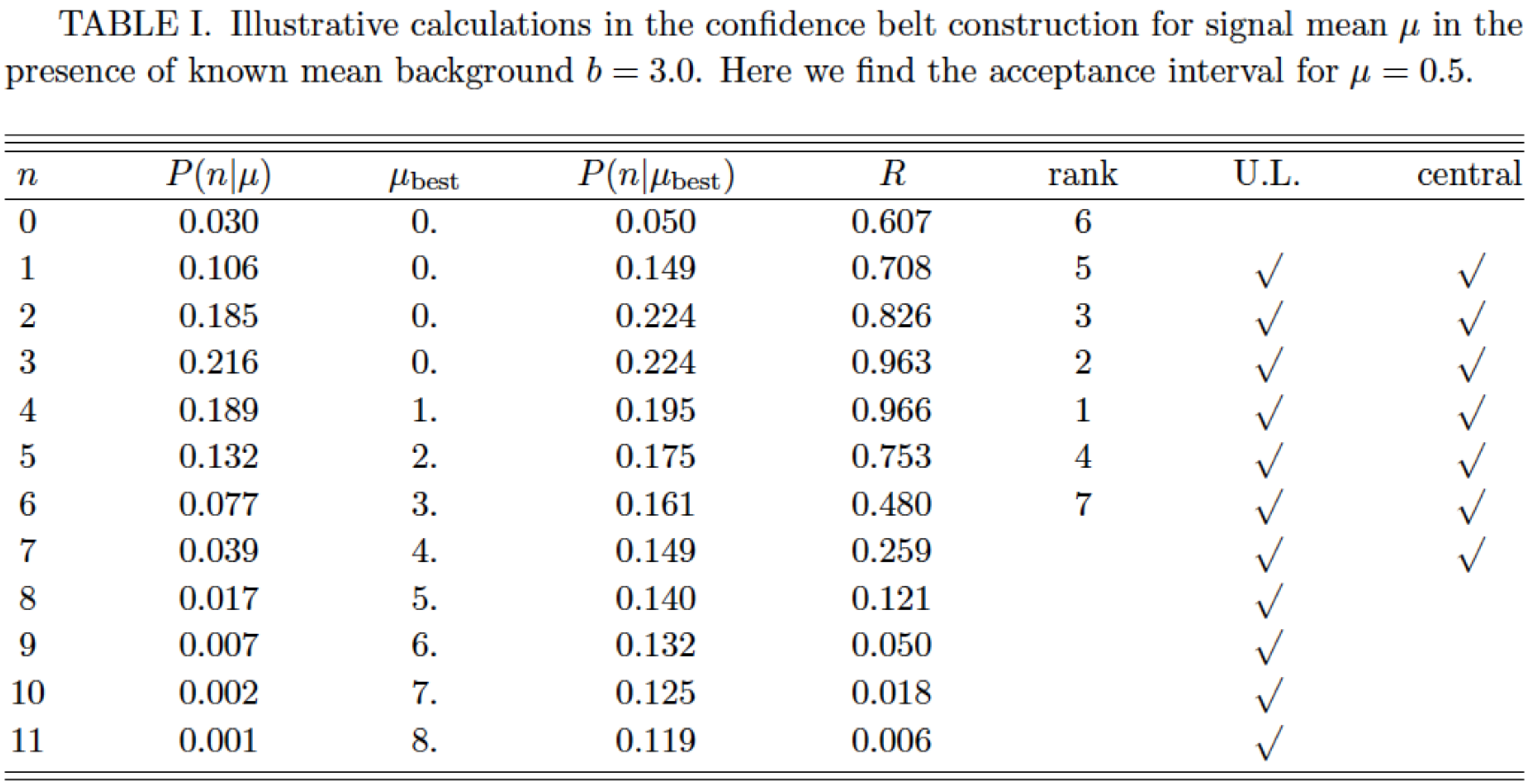


TABLE I. Illustrative calculations in the confidence belt construction for signal mean μ in the presence of known mean background $b = 3.0$. Here we find the acceptance interval for $\mu = 0.5$.

n	$P(n\|\mu)$	μ_{best}	$P(n\|\mu_{\text{best}})$	R	rank	U.L.	central
0	0.030	0.	0.050	0.607	6		
1	0.106	0.	0.149	0.708	5	√	√
2	0.185	0.	0.224	0.826	3	√	√
3	0.216	0.	0.224	0.963	2	√	√
4	0.189	1.	0.195	0.966	1	√	√
5	0.132	2.	0.175	0.753	4	√	√
6	0.077	3.	0.161	0.480	7	√	√
7	0.039	4.	0.149	0.259		√	√
8	0.017	5.	0.140	0.121		√	
9	0.007	6.	0.132	0.050		√	
10	0.002	7.	0.125	0.018		√	
11	0.001	8.	0.119	0.006		√	

Observed count

L(μ =0.5) of observed count

μ̂ that maximizes L of observed count

L(μ̂) of observed count

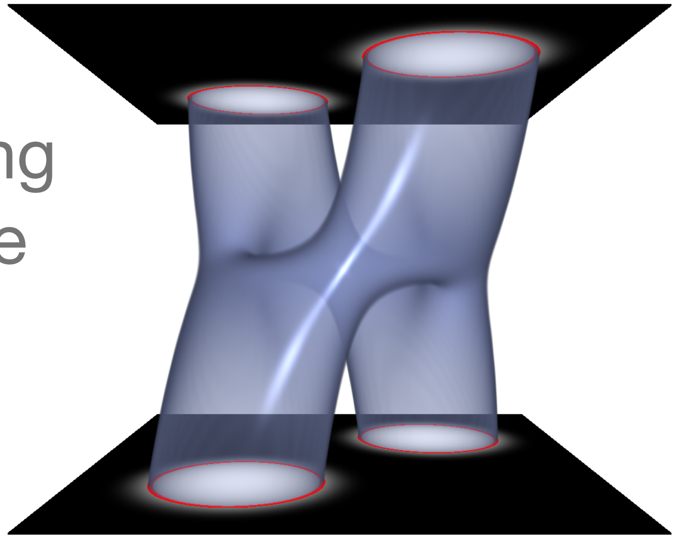
Likelihood ratio L(μ =0.5)/L(μ̂) (ordering score)

Can use this for Poisson http://stats.areppim.com/calc/calc_poisson.php

Real life — high dimensions

In most real problems likelihoods are complicated multidimensional functions that cannot be analytically maximized. Two main issues:

Constructing confidence intervals is a significant computing burden: for each test value of the parameter m, (i) generate many samples of pseudodata, (ii) fit, and (iii) then move to another m value etc.. Diverges quickly with dimensionality

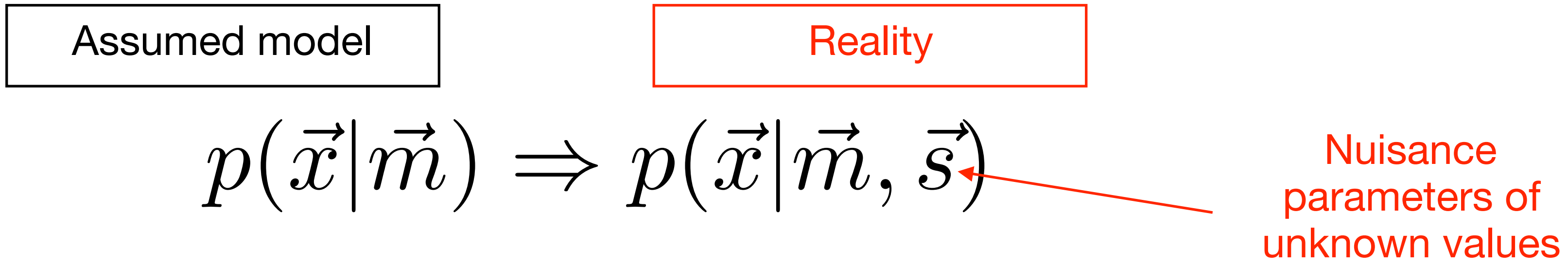


With highly-dimensional likelihoods the “projection” of the full-dimensional confidence band into the lower-dimensional subspace of interest leads to information loss: structure in the full dimensional space is lost when projected. The resulting confidence interval is bigger (less precise results).

Real life — nuisance parameters

In most problems, systematic uncertainties complicate the interval determination Neither Neyman nor Feldman-Cousins have a prescription for that.

Parametrize the uncertainty in the shape of the model by unknown nuisance parameters. Not interesting for the measurement but do influence the result.



Assumed model

Reality

$$p(\vec{x}|\vec{m}) \Rightarrow p(\vec{x}|\vec{m}, \vec{s})$$

Nuisance parameters of unknown values

Cannot define a pdf for s (otherwise s would be part of the model) but just a range. Goal: a procedure that guarantees coverage *whatever* is the value of the nuisance parameters within such ranges

Rigorous frequentist confidence intervals in the presence of nuisance parameters is a complicated problem for which no universal prescription yet exists.

Profile-likelihood ratio ordering

A promising approach: profile-likelihood-ratio (PLR) ordering.

FC ratio-ordering applied to likelihoods profiled (i.e., maximized) with respect to the uninteresting parameters. The profile-likelihood *is not a likelihood.* It is a lower-dimensional derivation of it obtained by maximizing the likelihood wrt to the nuisance parameters. However, it preserves some of the nice features of the likelihood ratio: its asymptotic distribution is known and independent of m

$$\mathrm{PLR} = \frac{L(x|m{=}m_0, \hat{s}^*)}{L(x|\hat{m}, \hat{s})}$$

Variable	Meaning
m	Parameters of interest (”physics parameters”)
s	Nuisance parameters
$\hat{m}, \hat{s}$	Parameters that maximize $L(x\|m, s)$
$\hat{s}^*$	Parameter that maximizes $L(x\|m = m_0, s)$

In practice

Generate pseudodata that sample the full multidimensional space of the parameters. fitt each sample twice, one with all parameters (physics and nuisance) floating, and another one with physics parameters fixed to their test value m_0.

1. Choose one value m_0 for m and one value s_0 for s, and generate pseudodata x accordingly
2. For each sample x (i) maximize $p(x|m=m_0,s)=L(m=m_0,s)$ with respect to s to get $L(m=m_0,\hat{s}^*)$ and (ii) maximize the likelihood $L(m,s)$ over the space of m and s to obtain $L(\hat{m},\hat{s})$
3. Rank all x in decreasing order of profile likelihood ratio $PLR=L(m=m_0,\hat{s}^*)/L(\hat{m},\hat{s})$
4. Start from the x with higher PLR and accumulate the others until the desired CL is reached.
5. Repeat for all values of m
6. [Repeat for values of s sampled in their whole range of existence]

Step 6 is essential to ensure the procedure has coverage for all values of the nuisance parameters.

How to treat nuisance parameters in generation?

Step 6 is essential to ensure coverage for all values of the nuisance parameters: this is the *"supremum p-value method"* (used by the CKMfitter group on combination of the measurements for CKM angle gamma up tyo 2010.) Can be expensive.

Sometimes circumvented using the *"plugin method"*: only generate pseudodata at the s values estimated on data. Equivalent to assume that the true values of the nuisance parameters are exactly those measured in data. Likely to be an optimistic assumption that spoils coverage.

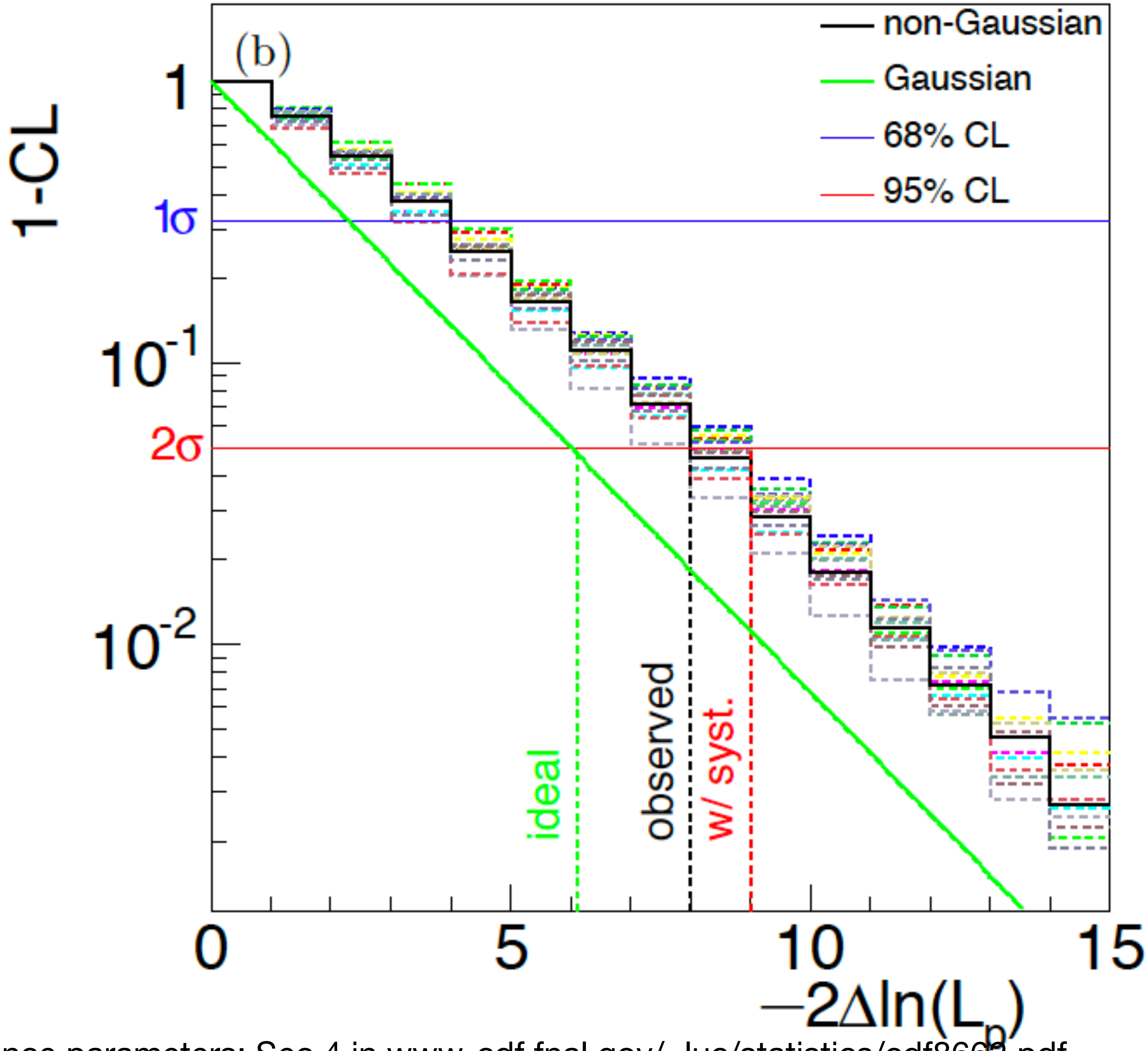
Midway between plugin and supremum: generate pseudata at $\vec{s}$ values sampled in a plausible subvolume centered on their estimates in data. Berger and Boos: sample along each dimension s_i a range around the estimated value $\hat{s}$ with CL much larger than the target CL of the profile-likelihood interval. (e.g, when constructing a 68% CL band in m, sample a 99.7% CL range in each dimension in s space) JASA, 89, 427 (1994)

Application of the Berger-Boos method

27-dimensional case

https://arxiv.org/pdf/0810.3229.pdf

Phys. Rev. Lett 100 161802,
Phys Rev D 85, 072002
Phys. Rev. Lett. 109, 171802,



Comprehensive review of treatment of nuisance parameters: Sec 4 in www-cdf.fnal.gov/~luc/statistics/cdf8662.pdf

The main burden

By this point you probably have realized that in a confidence interval construction, most of the time and effort is spent in generating a fitting simulated data sampled from p(x|m). Effort and the time needed when likelihoods are highly multidimensional can be disconcerting.

Any way of avoiding this?

Wilks' theorem

Asymptotically (large N), the distribution of the likelihood ratio

$$-2\ln \mathrm{LR}(m_0) = -2\ln \frac{p(x|m_0)}{p(x|\hat{m})}$$

approaches a χ^2 distribution with # of degrees of freedom equal to # of additional free parameters in the denominator wrt the numerator

Samuel S. Wilks (1906-1964)

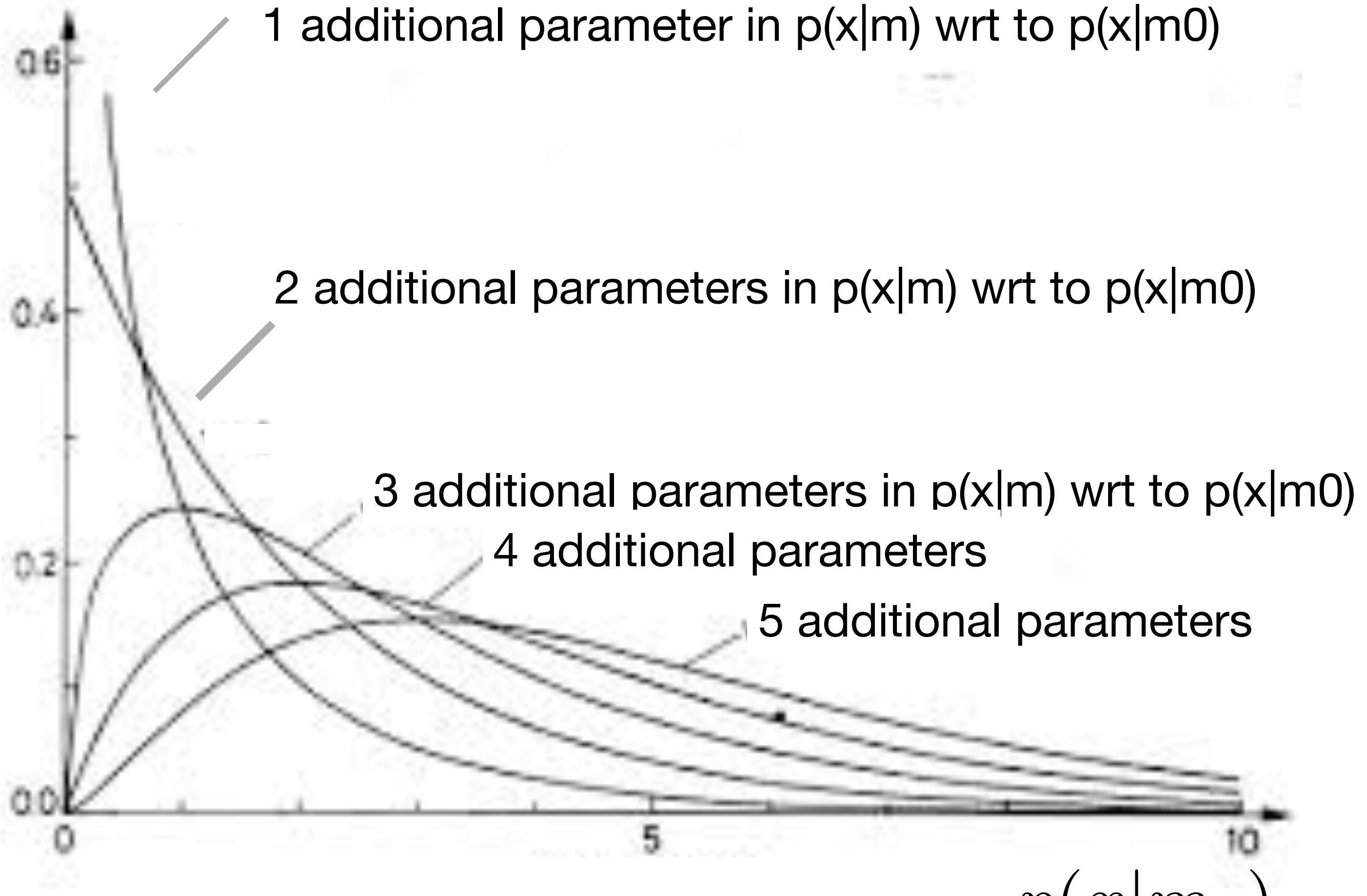
This holds independently of the shape of p(x|m) and on the value of m.

Great helps in usage of likelihood- and profile-likelihood-ratio as ordering quantities in the construction of intervals. If the likelihood is regular enough to be in asymptotic regime, one can avoid massive production of simulated experiments.

Wilks' theorem

No need to generate the sampling distributions of the ordering statistic (that is, no need to generate toys)

Just look at where the (profile)-likelihood ratio observed in my data falls along the appropriate curve (determined by the number of degrees of freedom)



1 additional parameter in p(x|m) wrt to p(x|m0)

2 additional parameters in p(x|m) wrt to p(x|m0)

3 additional parameters in p(x|m) wrt to p(x|m0)

4 additional parameters

5 additional parameters

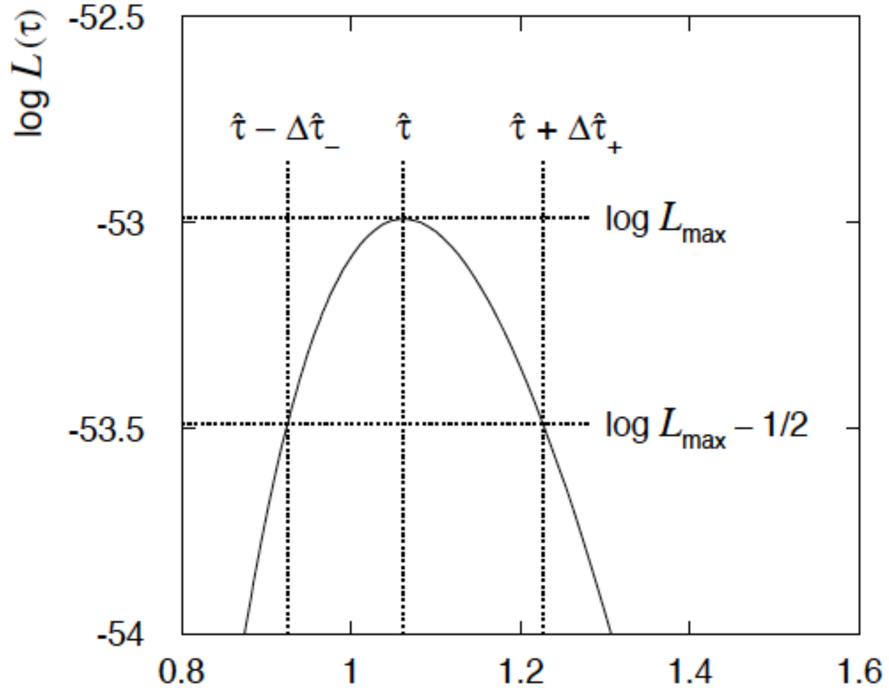
$$-2\ln\frac{p(x|m_0)}{p(x|\hat{m})}$$

How do I know if L is asymptotic? Look at a few samples of pseudodata, and compare with curves above.

Wilks' theorem at work — MINOS

Moves down from the maximum $L(\hat{m},\hat{s})$ evaluating $L(m_0,\hat{s}_m)$ at each point m_0 by maximizing wrt parameters $\vec{s}$ (i.e., likelihood of m profiled wrt $\vec{s}$).

When $L(m_0,\hat{s}_m)/L(\hat{m},\hat{s})$ equals the threshold values tabulated from the χ^2 distribution the corresponding projection of the profile-likelihood onto the m space approximates (large N) a Feldman-Cousins central confidence interval



$$-2\ln \mathrm{LR}(m_0) = -2\ln \frac{p(x|m_0)}{p(x|\hat{m})} = \Delta$$

Δ	CL				
	$n=1$	$n=2$	$n=3$	$n=4$	$n=5$
1.0	0.683	0.393	0.199	0.090	0.037
2.0	0.843	0.632	0.428	0.264	0.151
4.0	0.954	0.865	0.739	0.594	0.451
9.0	0.997	0.989	0.971	0.939	0.891

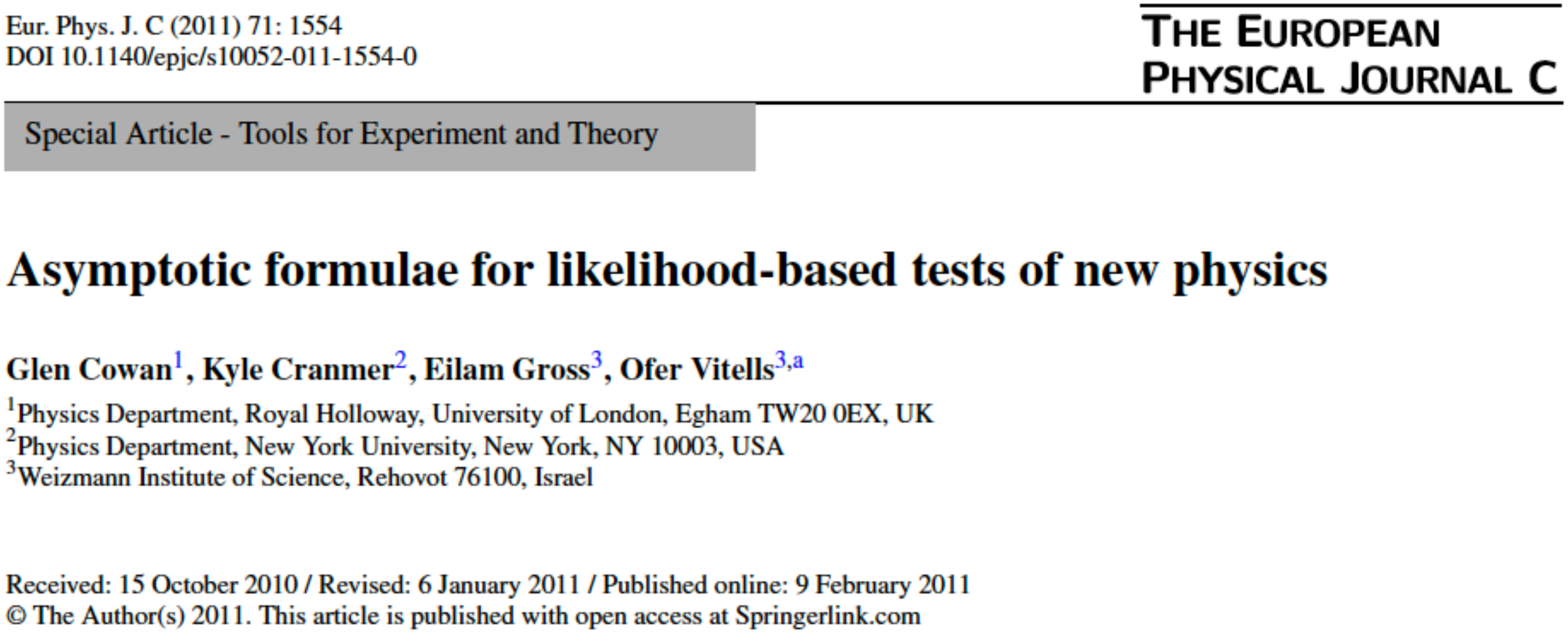
CL	Δ				
	$n=1$	$n=2$	$n=3$	$n=4$	$n=5$
0.683	1.00	2.30	3.53	4.72	5.89
0.90	2.71	4.61	6.25	7.78	9.24
0.95	3.84	5.99	7.82	9.49	11.1
0.99	6.63	9.21	11.3	13.3	15.1

"projection" onto the space of parameters of a 1(2)-dimensional likelihood at the point where -2lnLR varies by 1.0 units identifies a 1(2)-dimensional 68(39)% CL central interval

"projection" onto the space of parameters of a 3-dimensional likelihood at the point where -2lnLR varies by 6.25 units identifies a 3-dimensional 90%CL central interval

The Asimov asymptotic formulas

Significant recent breakthrough allows generalizing the Wilks theorem and provides key asymptotic formulas for the distributions of profile likelihood ratios used in confidence intervals and hypothesis tests.



Eur. Phys. J. C (2011) 71: 1554
DOI 10.1140/epjc/s10052-011-1554-0

THE EUROPEAN PHYSICAL JOURNAL C

Special Article - Tools for Experiment and Theory

Asymptotic formulae for likelihood-based tests of new physics

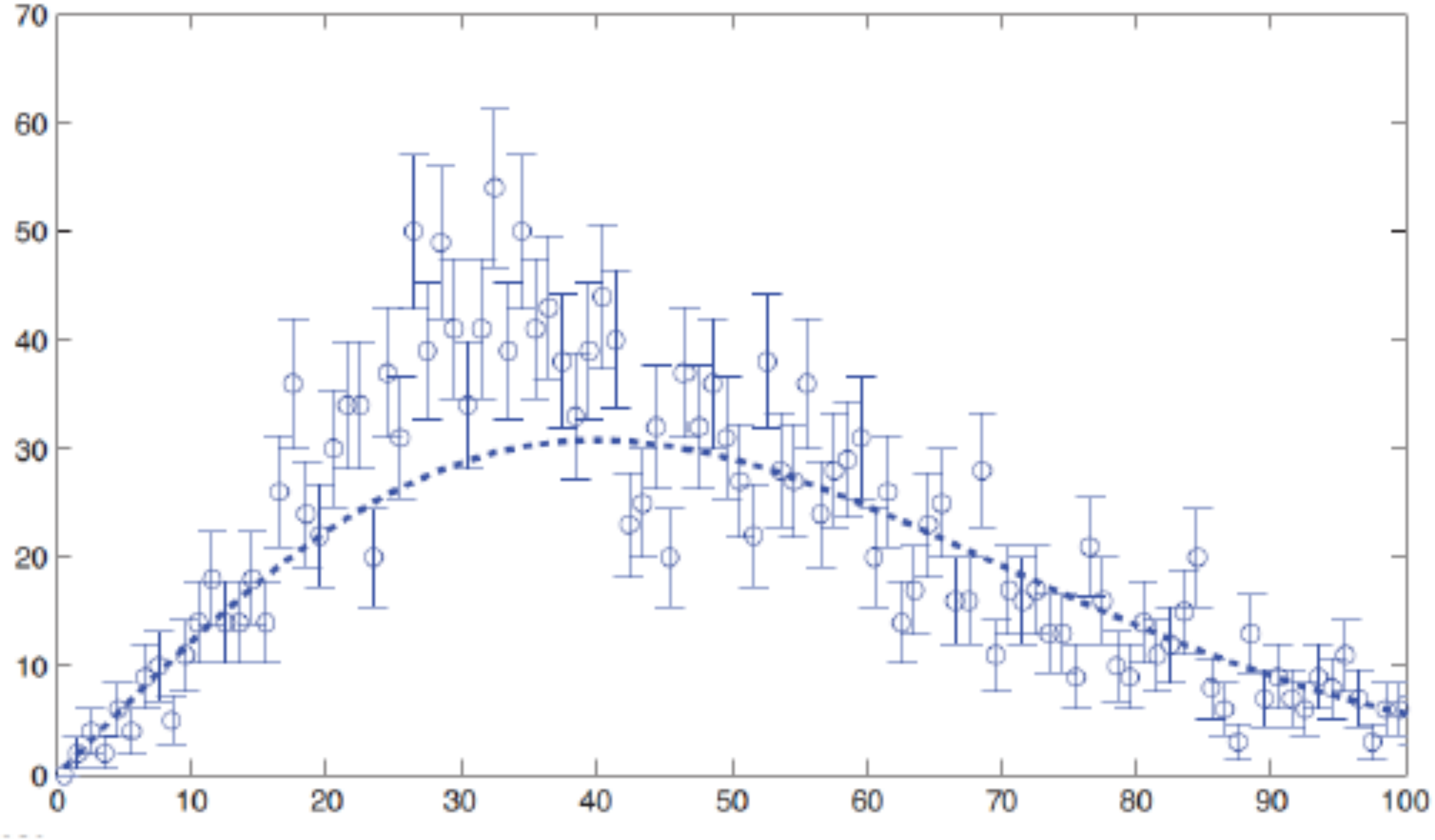
Glen Cowan[1], Kyle Cranmer[2], Eilam Gross[3], Ofer Vitells[3,a]

[1]Physics Department, Royal Holloway, University of London, Egham TW20 0EX, UK
[2]Physics Department, New York University, New York, NY 10003, USA
[3]Weizmann Institute of Science, Rehovot 76100, Israel

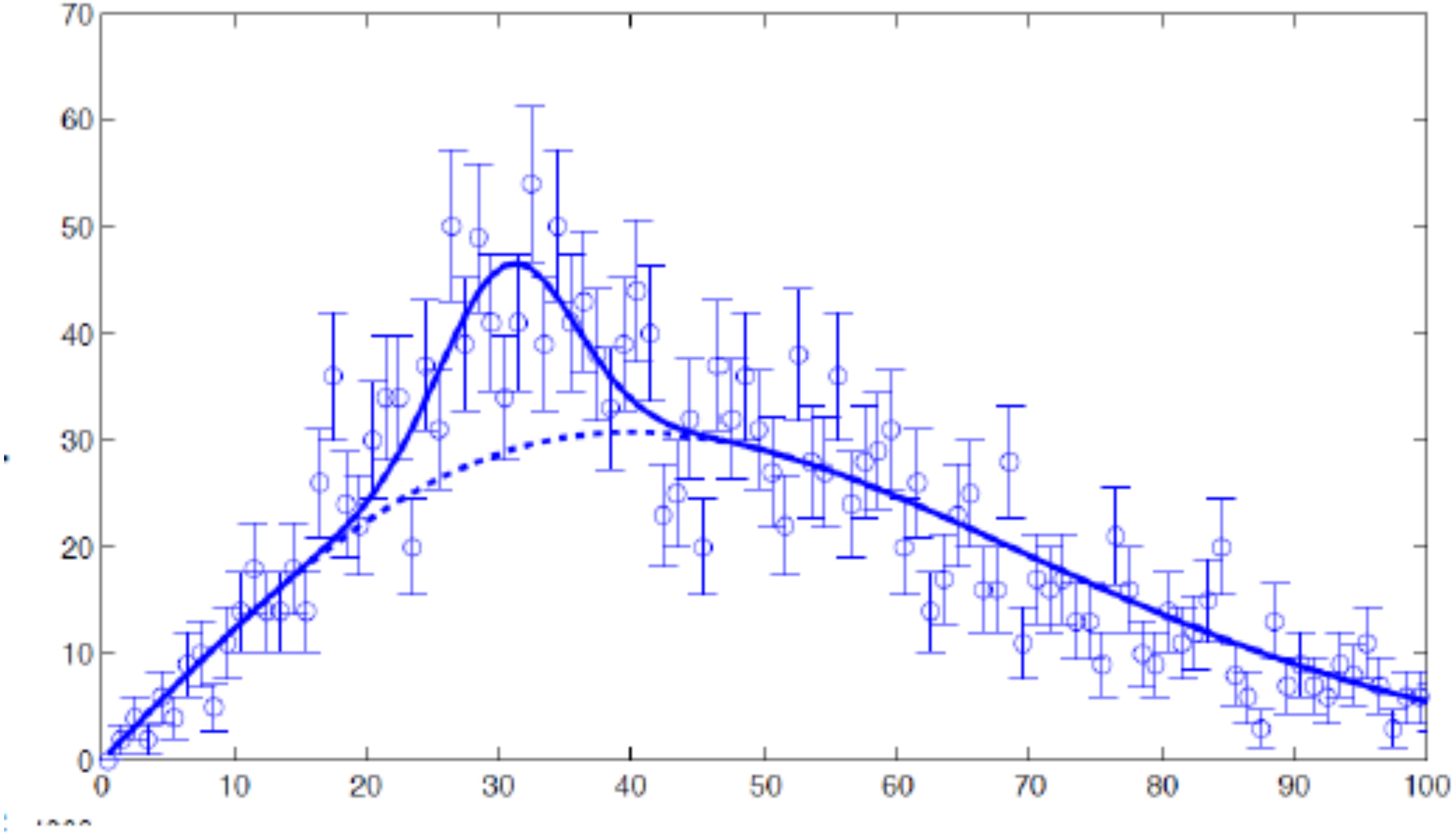
Received: 15 October 2010 / Revised: 6 January 2011 / Published online: 9 February 2011

Hypothesis testing

Are my data compatible with background?



Or they suggest the presence of a signal?

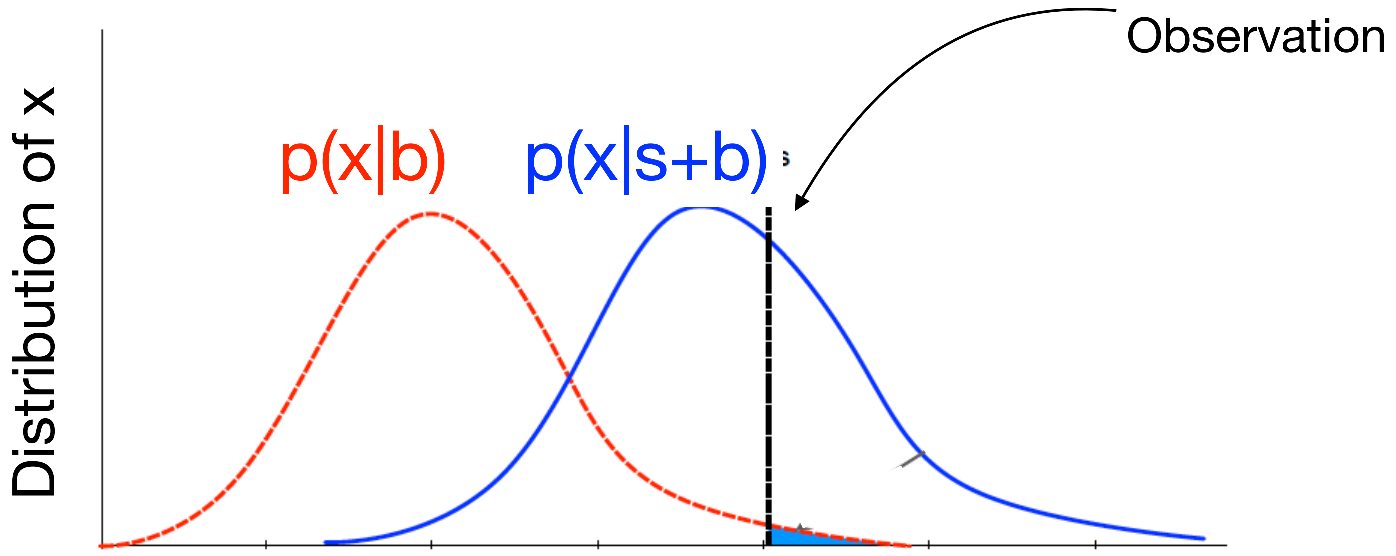


The p-value is a random variable that helps answering this question

http://priceonomics.com/the-guinness-brewer-who-revolutionized-statistics/

Ingredients (prepare prior to any observation)

1. Need two hypotheses. For instance: only known phenomena contribute“null” or “background”) new phenomena contribute too (“alternate” or “signal”)



Arbitrary function x of the data that allows separating between the two hypotheses

2. Need a function x of the data (e.g., signal-event count), whose distribution under the null p(x|b) “departs” from that under the signal hypothesis p(x|s+b).

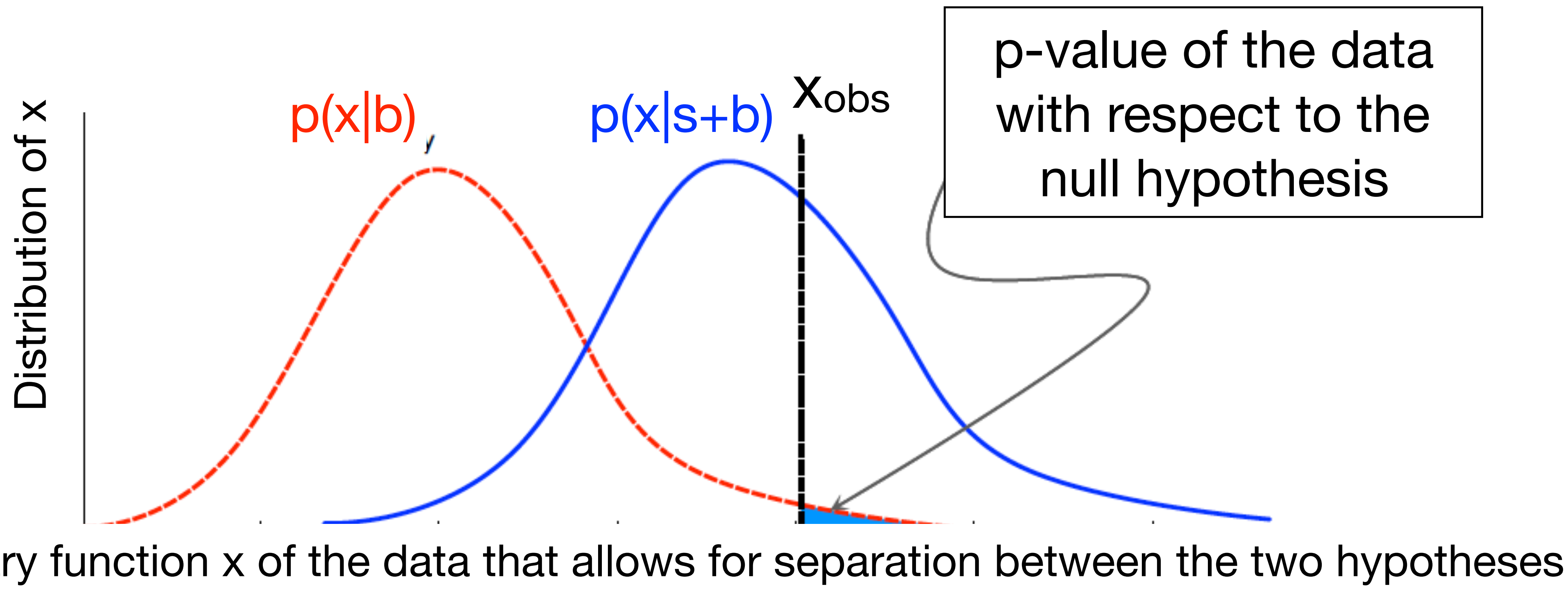
3. Generate these two distributions (typically done using simulation)

3. Set, prior to the observation, the false-positive rate: how much “signal-like” the observed value of x should be to exclude the background only hypothesis.

p-values for discovering a new effect

Observe x_{obs}. The location of x_{obs} relative to the two pdf offers a quantitative measure of data compatibility with either hypotheses.

p-value: relative fraction of the integral of the null model over values of x as signal-like as those observed and more. The smaller the p-value, the stronger the evidence against the null hypothesis. If p-value < false-positive rate, exclude the background-only hypothesis at CL = 1-(p-value).

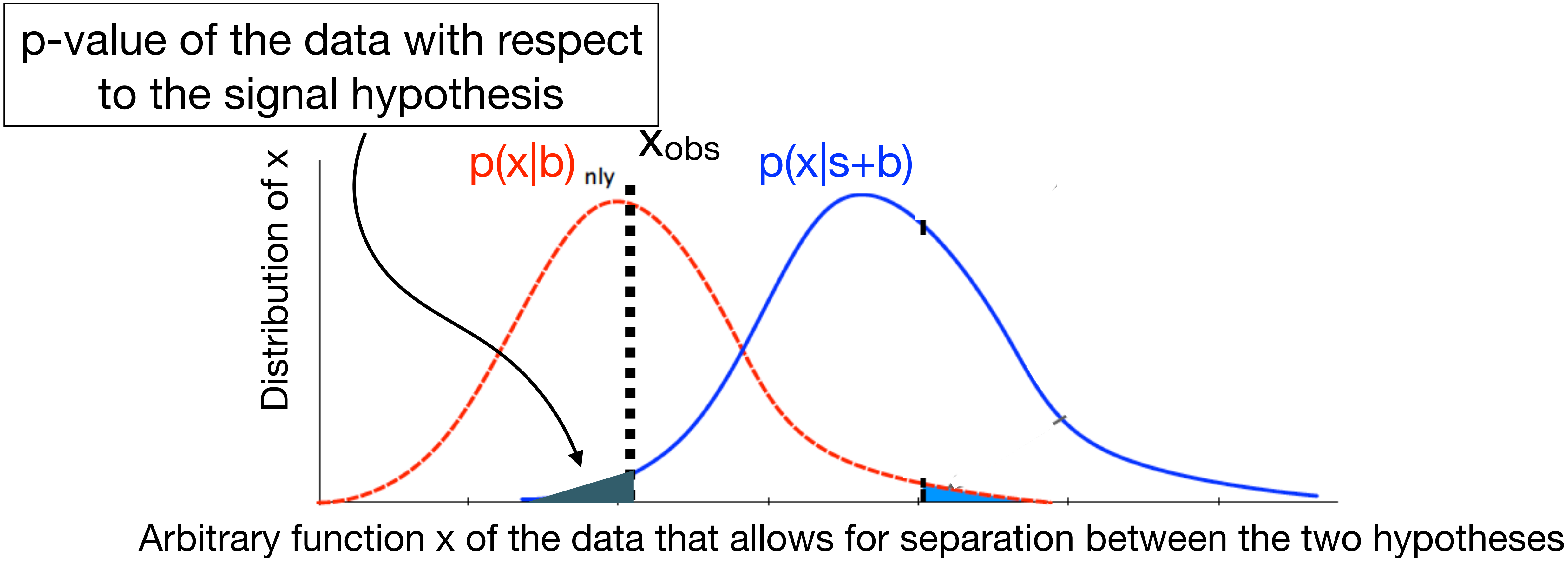


p-value of the data with respect to the null hypothesis

p-values for excluding a new effect

If the purpose is to exclude a new effect, then one tests the signal hypothesis, and quotes the p-value with respect to that.

Is the relative fraction of the integral of the signal model over values of x as background-like as that observed and more. The smaller the p-value, the stronger the evidence against the signal hypothesis.

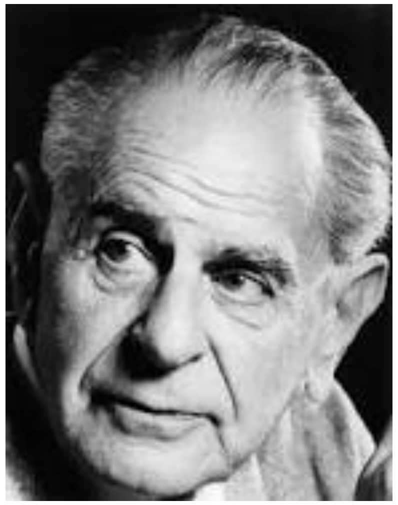


p-value of the data with respect to the signal hypothesis

This is Popperian testing

Cannot prove that an hypothesis is true, only that it's false.

"Discover" a signal by excluding its absence (that is, by excluding that only background contributes). Limit to the existence of a signal by excluding is presence.



Karl Popper (1902-1994)

A p-value is not a probability. It is a random variable (function of the data) that is distributed uniformly if the tested hypothesis is true.

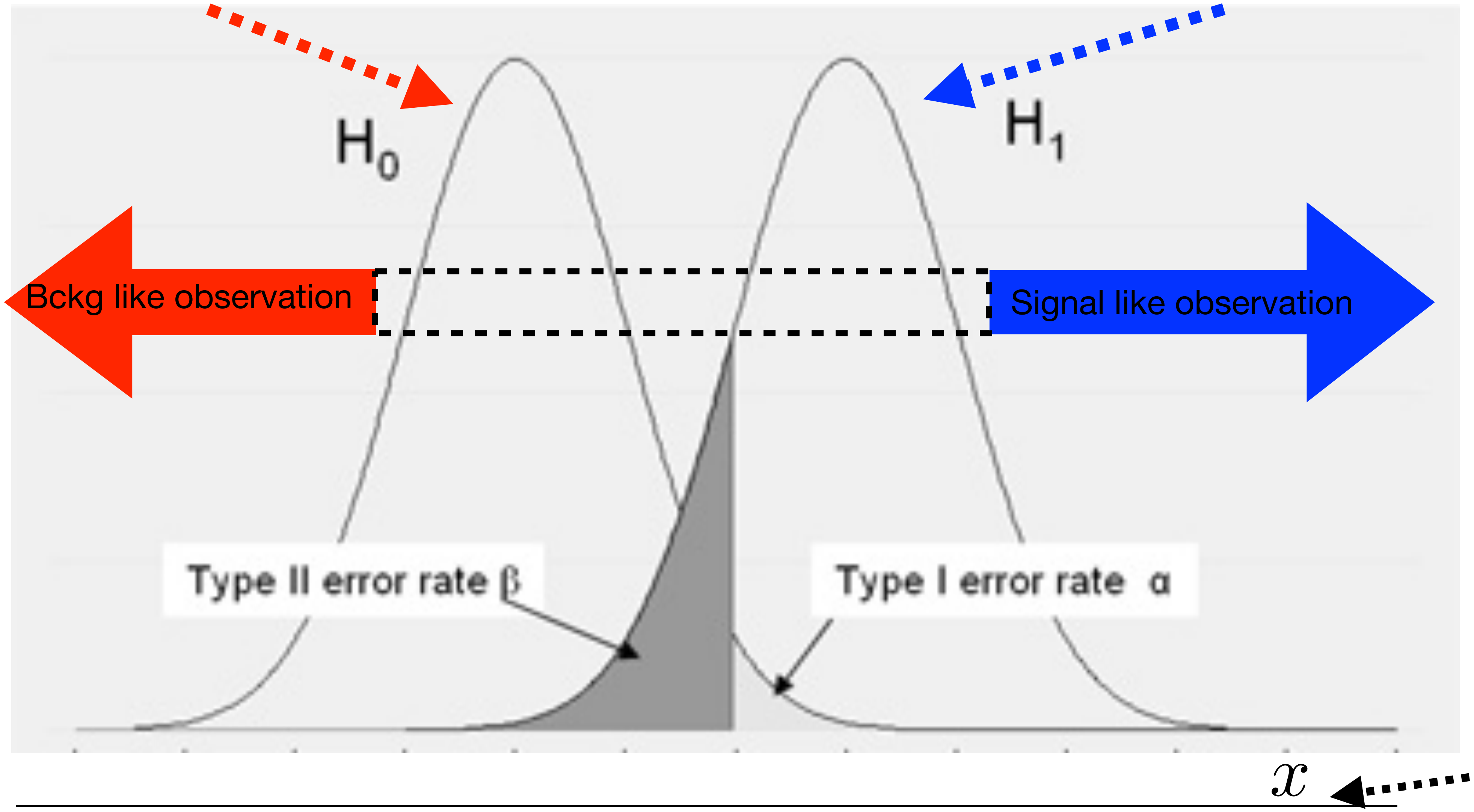
It does not express the probability that an hypothesis is true or false!
Wrong claim "The measurement shows that the probability for hypothesis blah is .."

P-values connect to the probability to observe x_{obs} or a more extreme value *if a specific hypothesis were true*. Proper claim: "Assuming that the hypothesis blah holds, the probability to observe a fluctuation as extreme as that observed in our data or more is…"

Nomenclature recap

This is p(x|b), the distribution of x under the null hypothesis

This is p(x|s+b), the distribution of x under the signal hypothesis



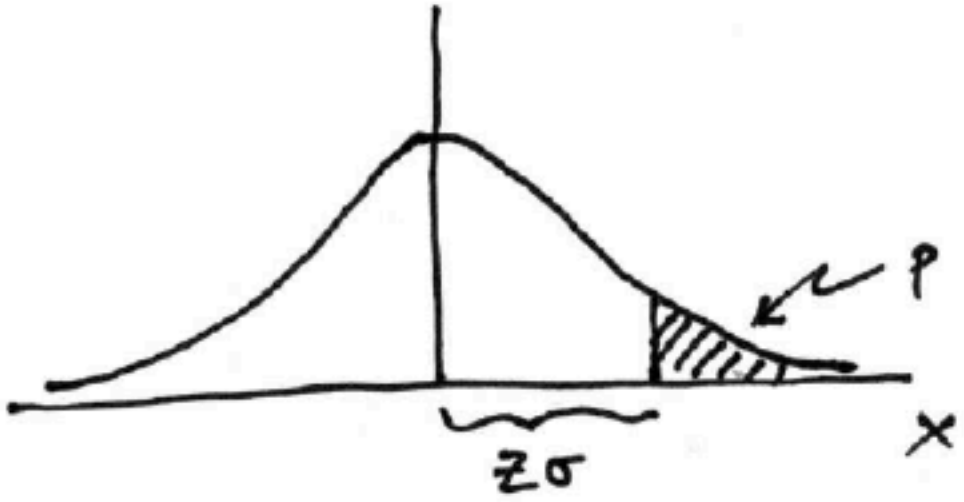
This is x, whatever function of data whose distribution is sensitive to separate H0 from H1

Symbol	Meaning
α	Rate of false positives (Type I error: reject H_0, while it was true)
β	Rate of false negatives (Type II error: reject H_1, while it was true)
$1-\beta$	Power of the test

“Significance”

“At how many sigma such and such result is significant?”

The “number of sigma” (or z-value) is just a remapping of p-values into integrals of one tail of a Gaussian. It expresses by how many sigma from the mean my observation would be if the test statistic x would be distributed as Gaussian



$$p = \int_Z^\infty \frac{1}{\sqrt{2\pi}} e^{-x^2/2}\, dx = 1 - \Phi(Z)$$

`1 - TMath::Freq`

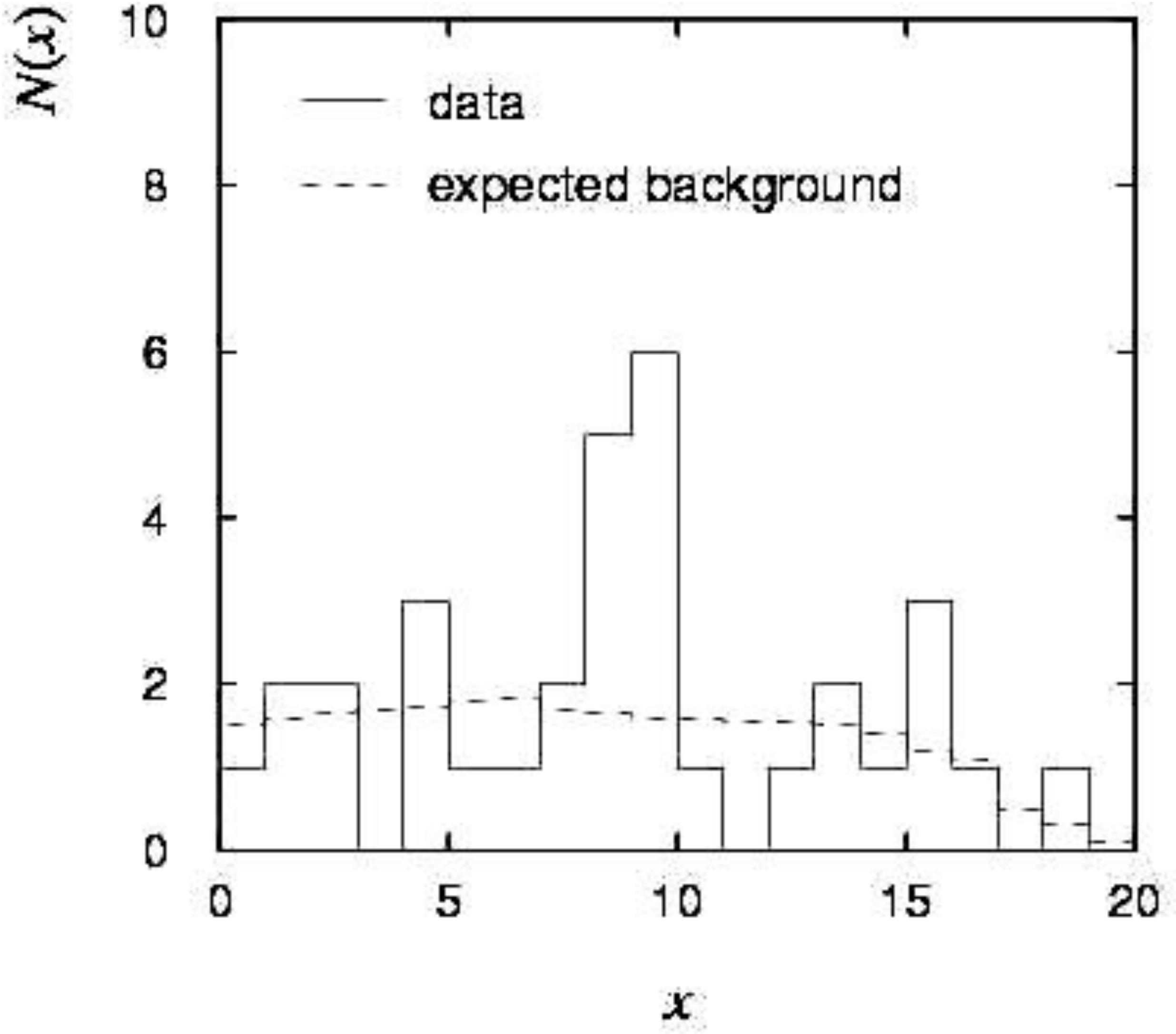
$$Z = \Phi^{-1}(1-p)$$

`TMath::NormQuantile`

[Cowan]

p-values in mass peak

Suppose you measure a value x for each event and bin the resulting distribution.



The count in each bin is a Poisson random variable, whose mean in the H0 hypothesis is given by the dashed line

$$P(n; s, b) = \frac{(s+b)^n}{n!} e^{-(s+b)}$$

Observe a peak of 11 events in the central bins, with expected background 3.2 events.

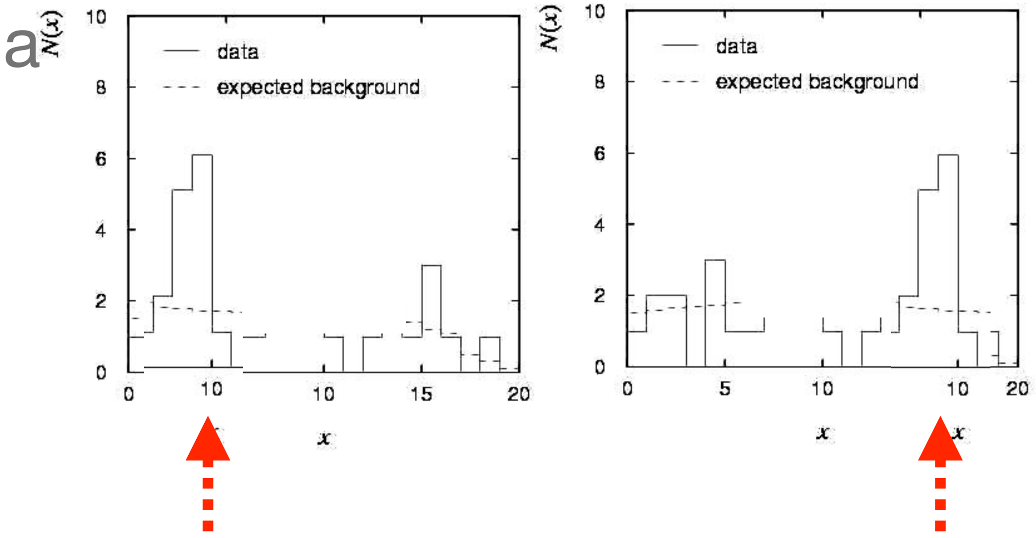
P-value for the background-only hypothesis is P(n>=11, b=3.2, s=0) = $5*10^{-4}$

Is this evaluation fair or biased?

[Cowan]

“Local” p-value and “look-elsewhere effect”

That evaluation only accounts for the chances of a upward fluctuation in that very position at x~9. That’s the “local p-value”.



Peak could have been observed here

..or here

“global p-value” need to account for the chances that an excess could have arisen in any pair of adjacent bins. With 20 bins (10 pairs of adjacent bins) the local p-value gets multiplied by ≈10.

The larger the size of the test space, the higher the probabilities to observe rare fluctuations.

When quoting p-values, need to correct for the effect of multiple testing (i.e., account that we have also been “looking elsewhere” from where the anomaly is).

Use simulation, or approximate correction factors, e.g., in EPJ C70, 525 (2010)

The conventional “5σ rationale”

HEP experimenters conventionally agree to deal with the LEE by setting a rather extreme standard for p-values to justify claims of new effects. (Originated by a survey of experimental results on “far-out hadrons” in 1968 — see backup)

One requires the null to be rejected with significance of 3.5σ (for “evidence”) and 5σ (“observation”), corresponding to very small p-values (fluctuations that occur 3 times every 10 million trials).

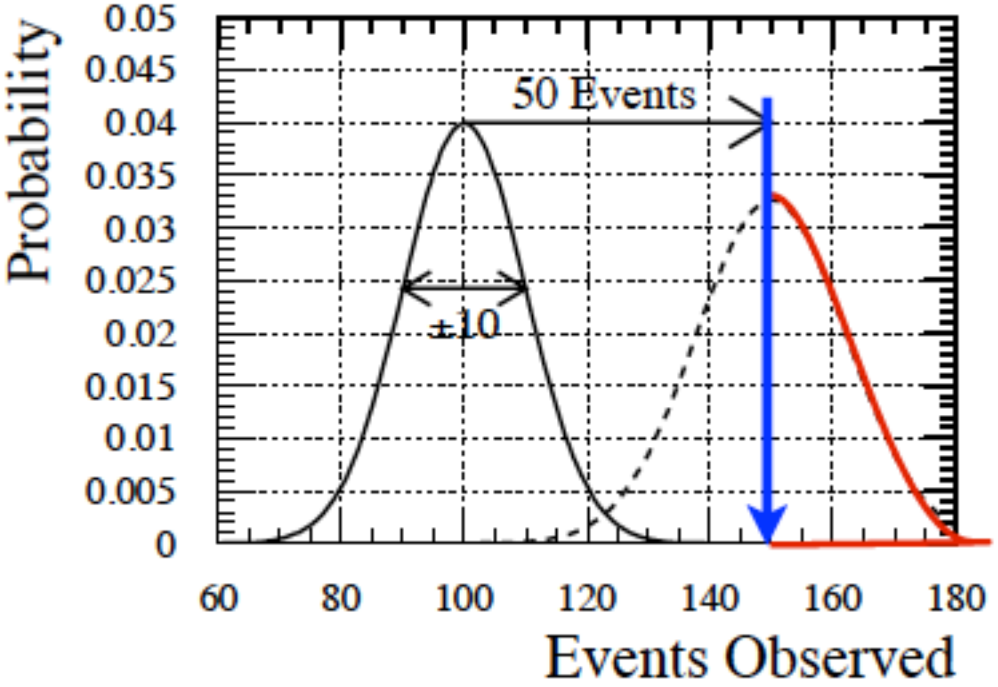
The loose rationale is that such high thresholds should protect from the effects above.

However, one-size-fits-all does not seem appropriate here.

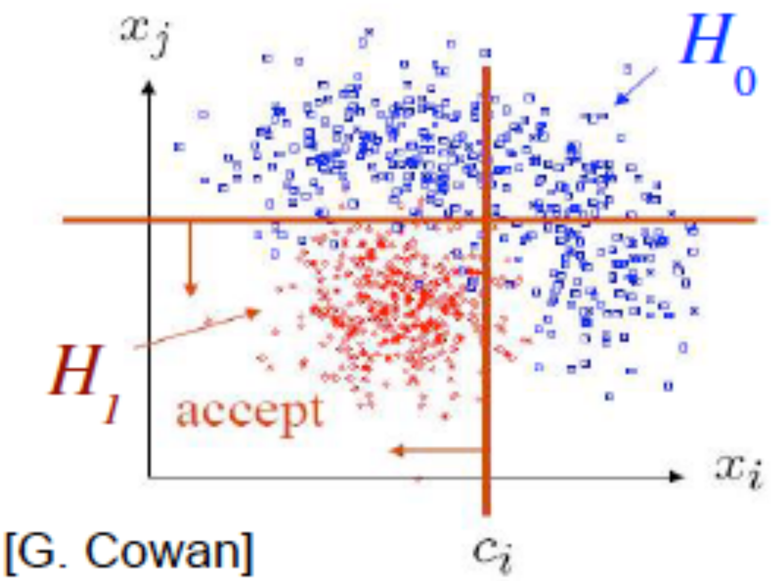
Which function x to choose?

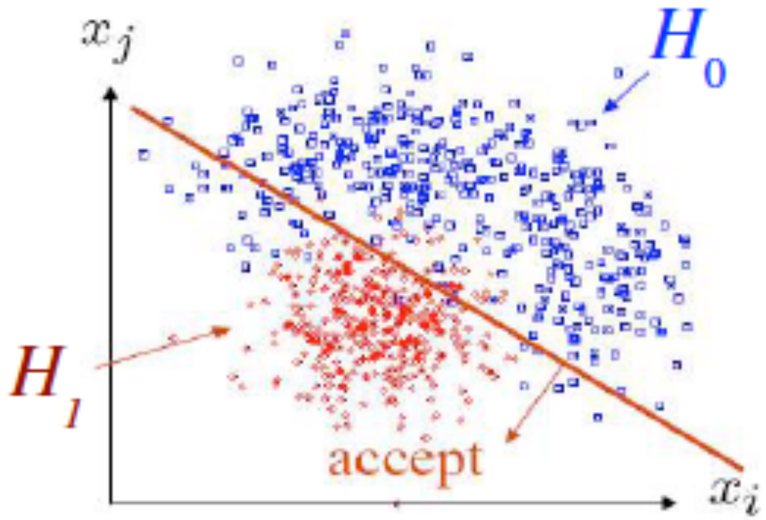
Back to p-values.

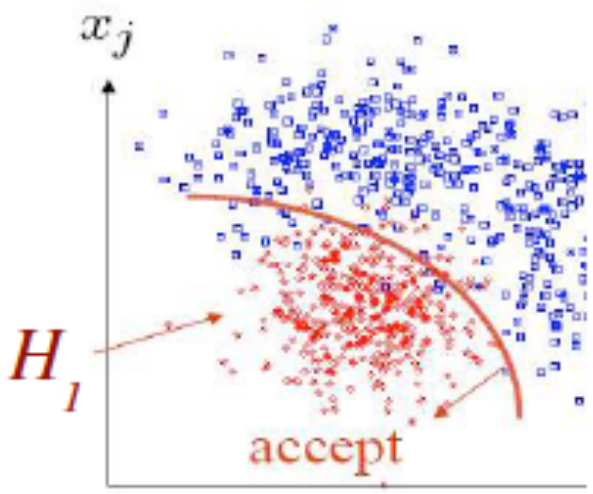
Can we exploit the arbitrariness in choosing the test quantity x? Can we devise a function of the observables x that maximizes the power of my test at fixed false-positive rate.



Pretty obvious in simple counting experiments. Less obvious in multiple-dimensional nonlinear problems





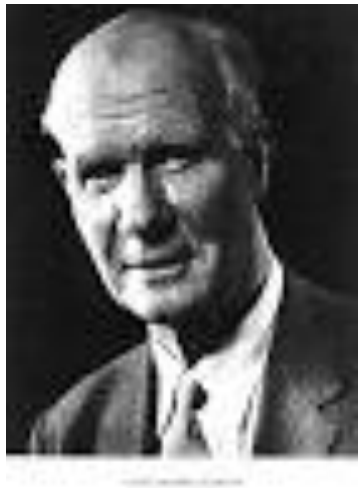


[G. Cowan]

Neyman-Pearson lemma

It does exist an universal statistic for optimal separation between the two hypotheses (for simple completely specified hypotheses)

Jerzy Neyman
(1894-1981)



Egon S. Pearson
(1885-1980)

Ratio between the likelihood for the signal+background hypothesis (H1) and the likelihood for the background-only hypothesis (H0)

The region W of acceptance of the null which minimises the probability to accept the null when the signal hypothesis is true is the contour

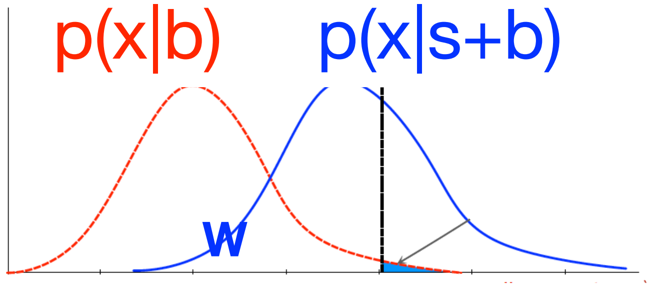
$$\frac{p(x|H_1}{p(x|H_0)} > k_\alpha$$

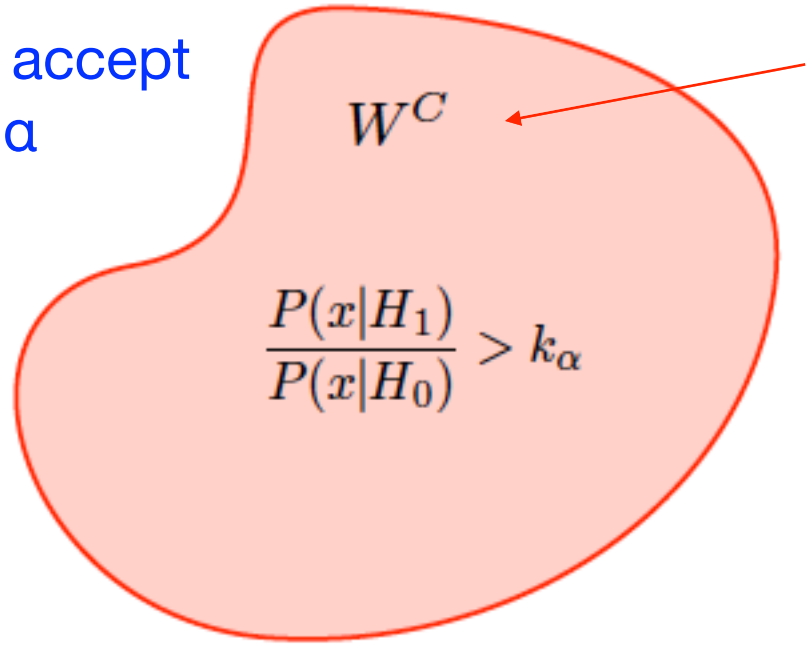
Any region that has the same false-positive rate would have higher rate of false negatives (technically, less power)

NP-lemma illustrated proof

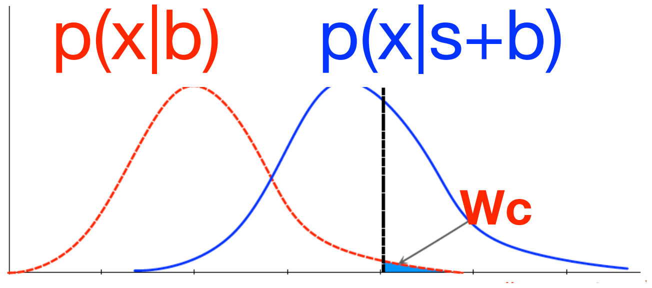
Take a contour of the likelihood ratio that has a given rate α of false positives, that is a given probability under H0

Region W: if data fall here we accept H0; probability under H0 is 1-α





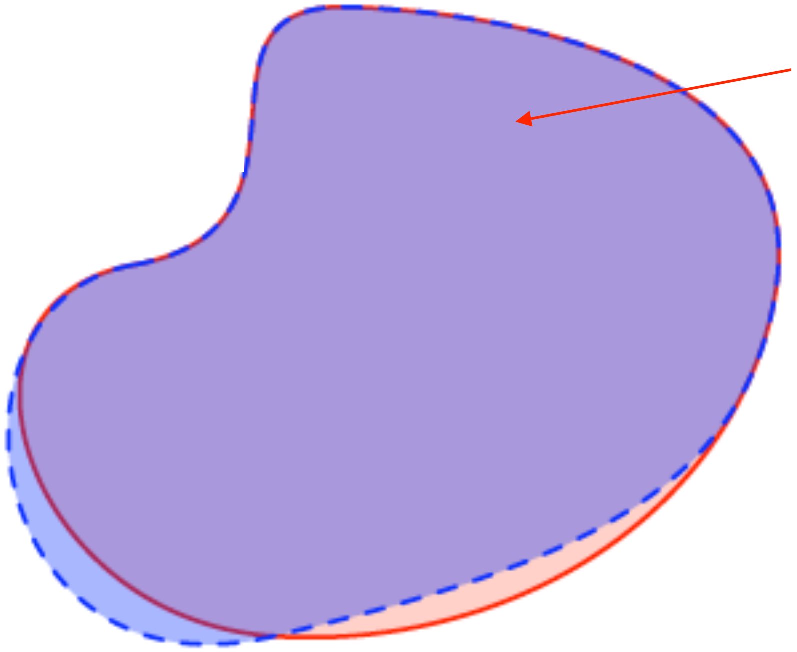
Region W^c: if data fall there we reject H0; probability under H0 is α



[Cranmer]

NP-lemma illustration

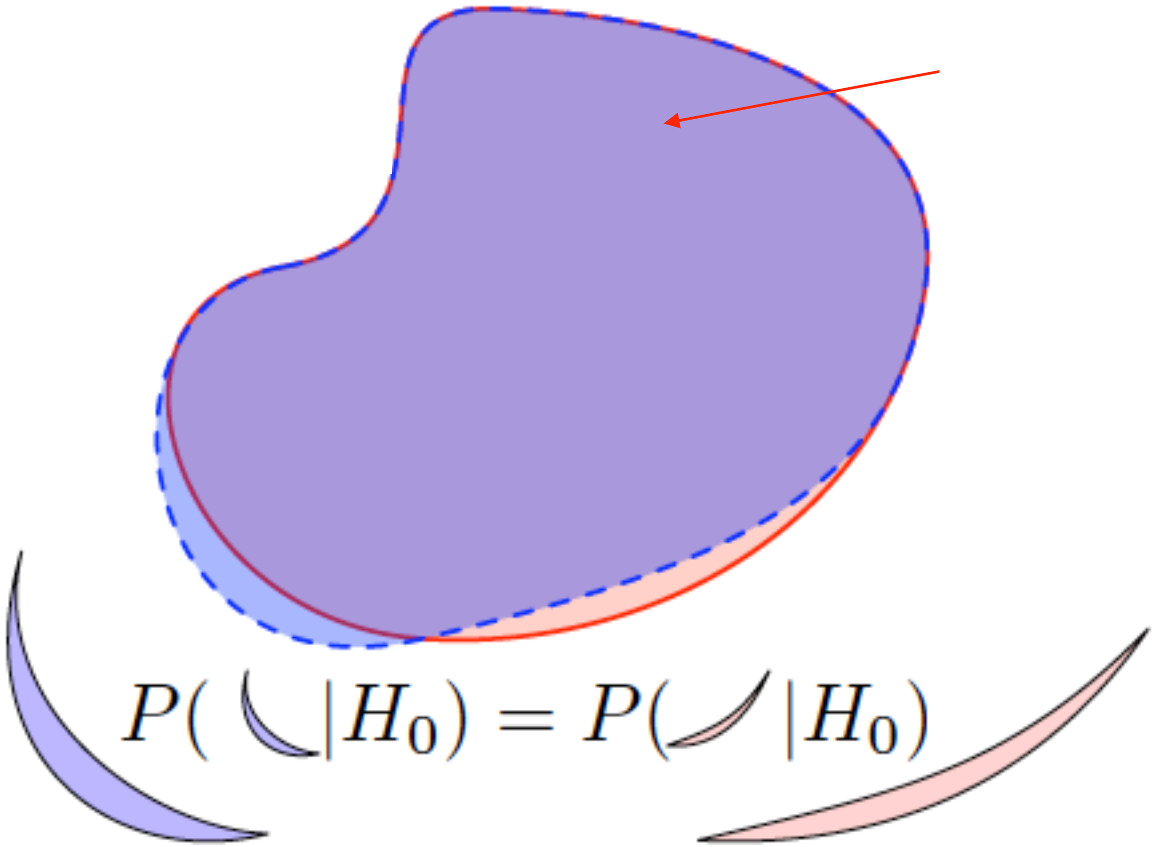
Take a variation that has the same rate α of false positives (same probability under H0)



[Cranmer]

NP-lemma illustration

Take a variation that has the same rate α of false positives (same probability under H0)



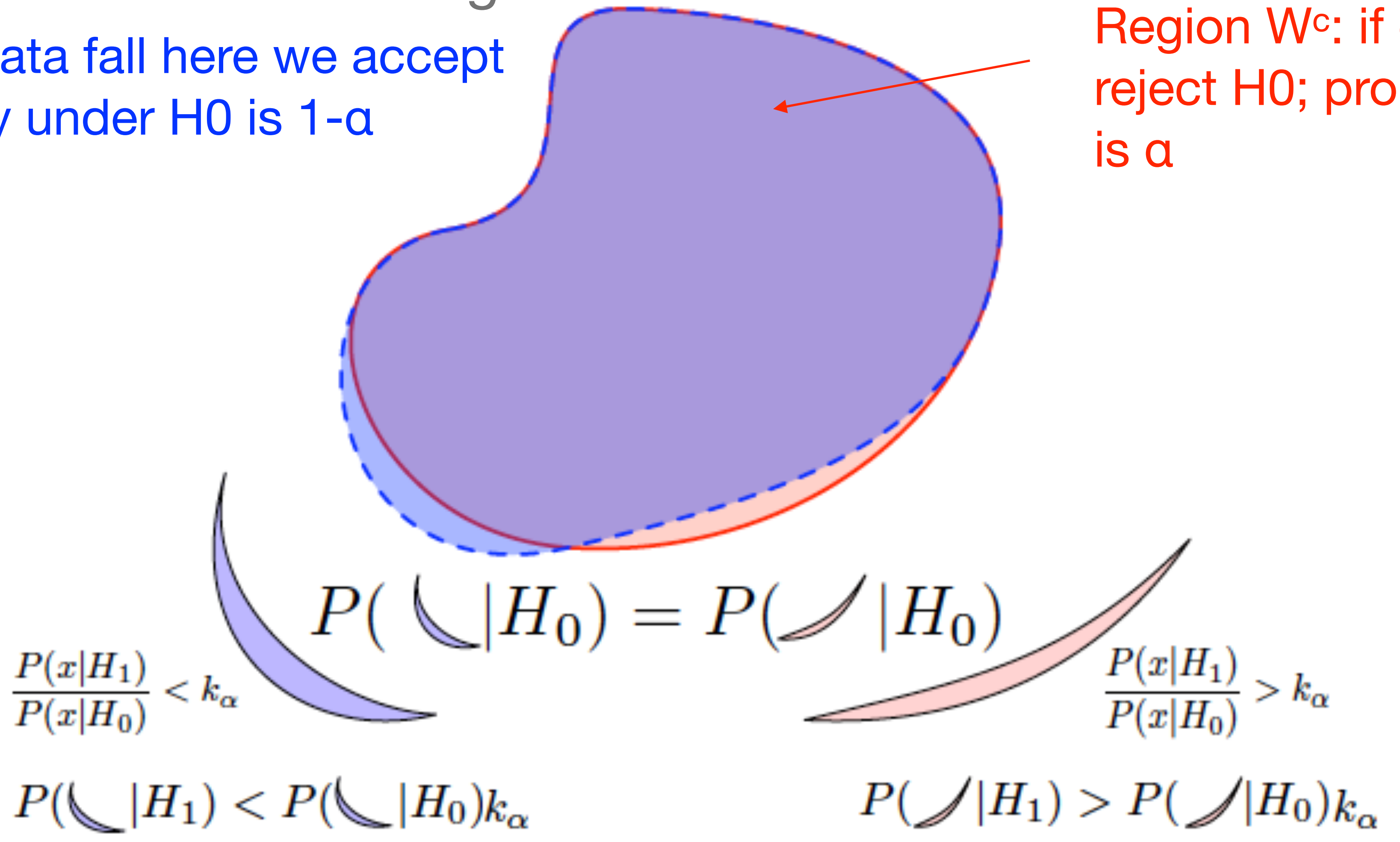
[Cranmer]

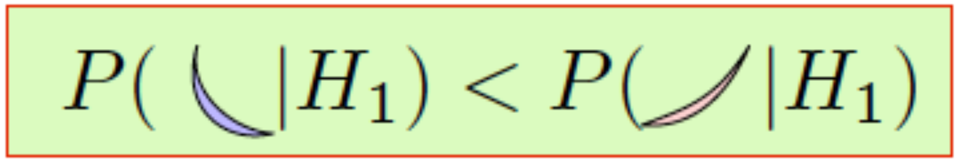
NP-lemma illustration

Because the region gained with the new contour was outside of the likelihood ratio contour and the region lost lost was inside it, the hierarchy between probabilities under H0 and H1 in the two regions is inverted.

Region W: if data fall here we accept H0; probability under H0 is 1-α

Region W^c: if data fall there we reject H0; probability under H0 is α


$P(\;|H_0) = P(\;|H_0)$
$\frac{P(x|H_1)}{P(x|H_0)} < k_\alpha$
$\frac{P(x|H_1)}{P(x|H_0)} > k_\alpha$
$P(\;|H_1) < P(\;|H_0)k_\alpha$
$P(\;|H_1) > P(\;|H_0)k_\alpha$



The new region region has less power.

[Cranmer]

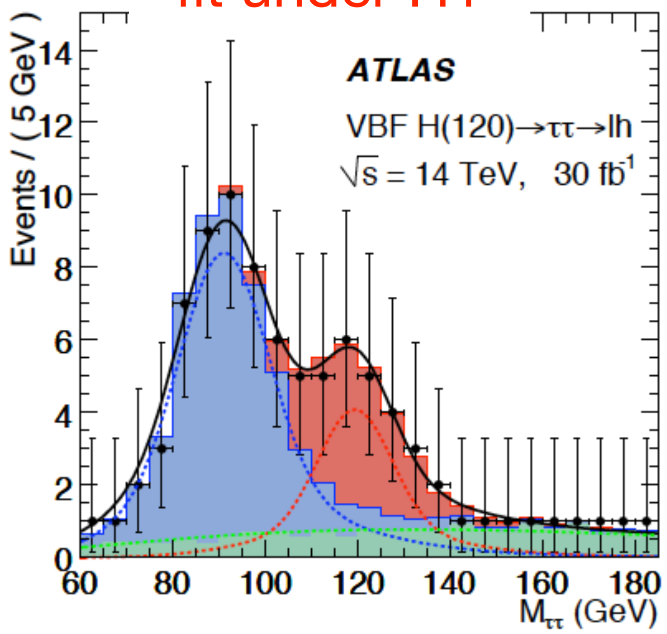
(profile) likelihood-ratio as a test statistic

Convenient because (1) has optimal performance and (2) allows for testing with no need to laboriously construct distributions by generating and fitting pseudodata since its large-sample distribution is known (χ^2)

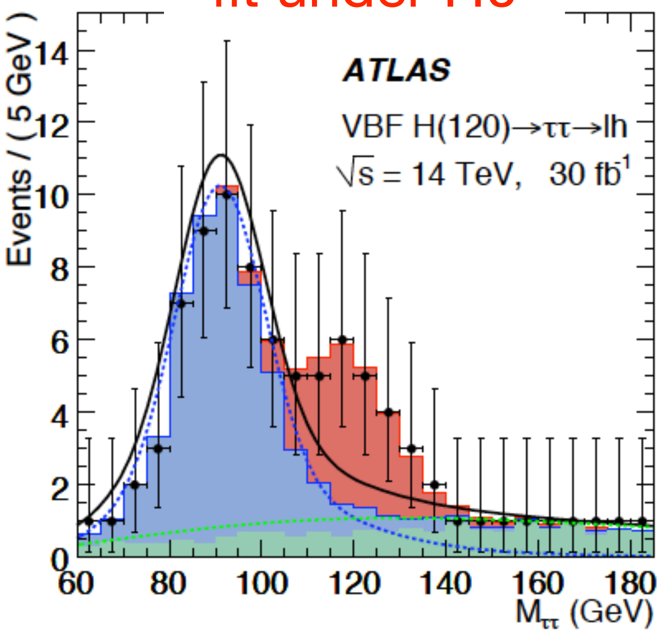
[Cranmer]

1. Fit data under H0: i.e. with a likelihood that only has “background” parameters.
2. Fit data under H1: i.e. with a likelihood that includes n additional “signal” free parameters
3. The ratio between the resulting values of the likelihood functions at their maxima is distributed as a χ^2 with n degrees of freedom.
4. Comparison of the ratio obtained in data with the relevant χ^2 distribution allows for testing H1 vs H0.

fit under H1

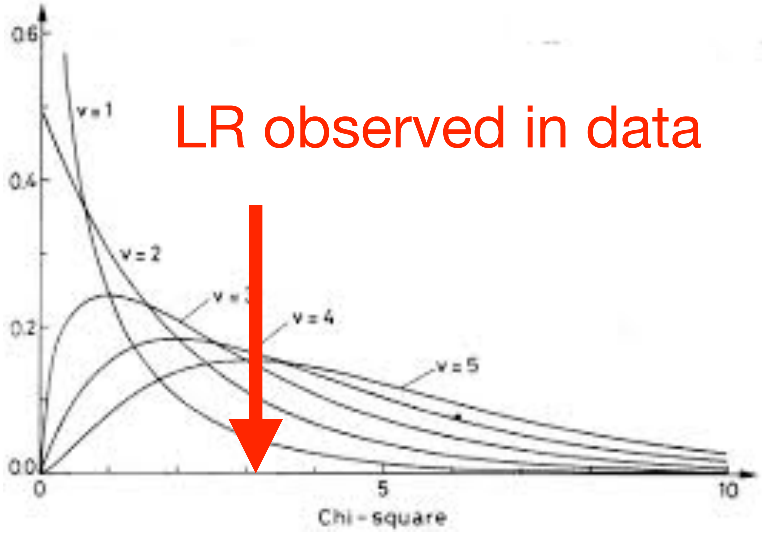


fit under H0

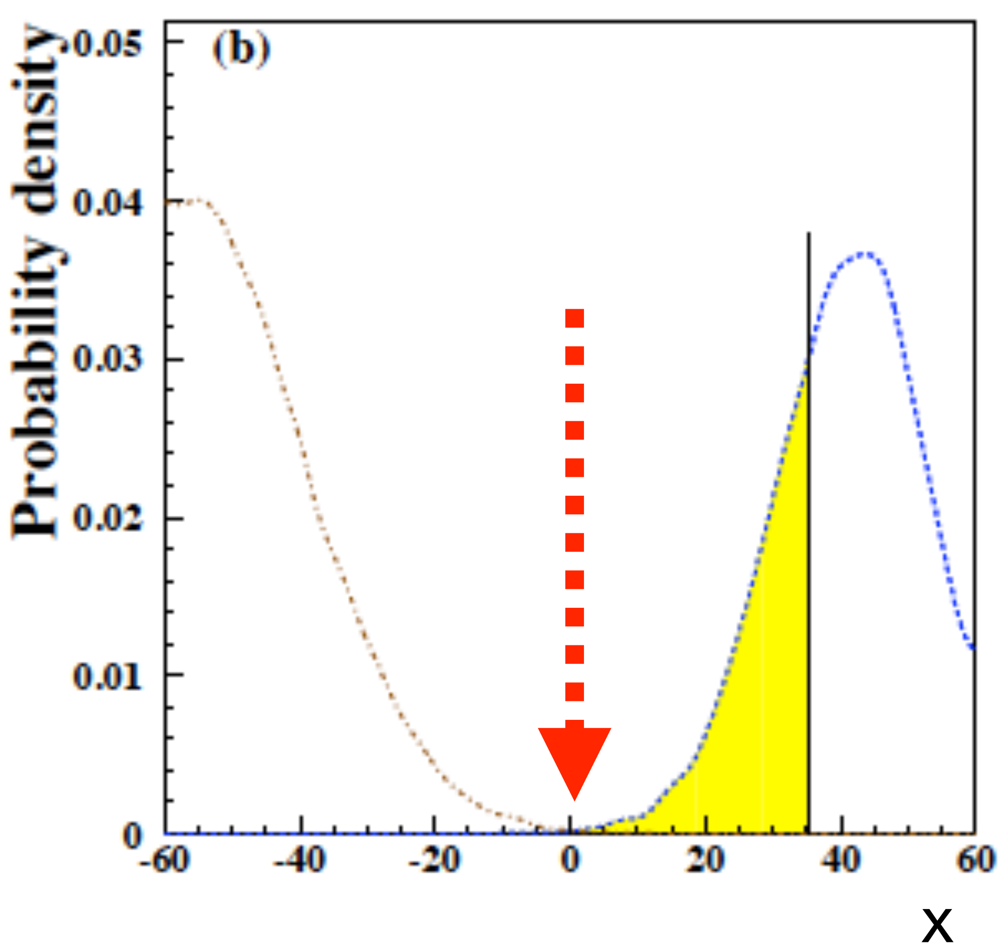


χ^2

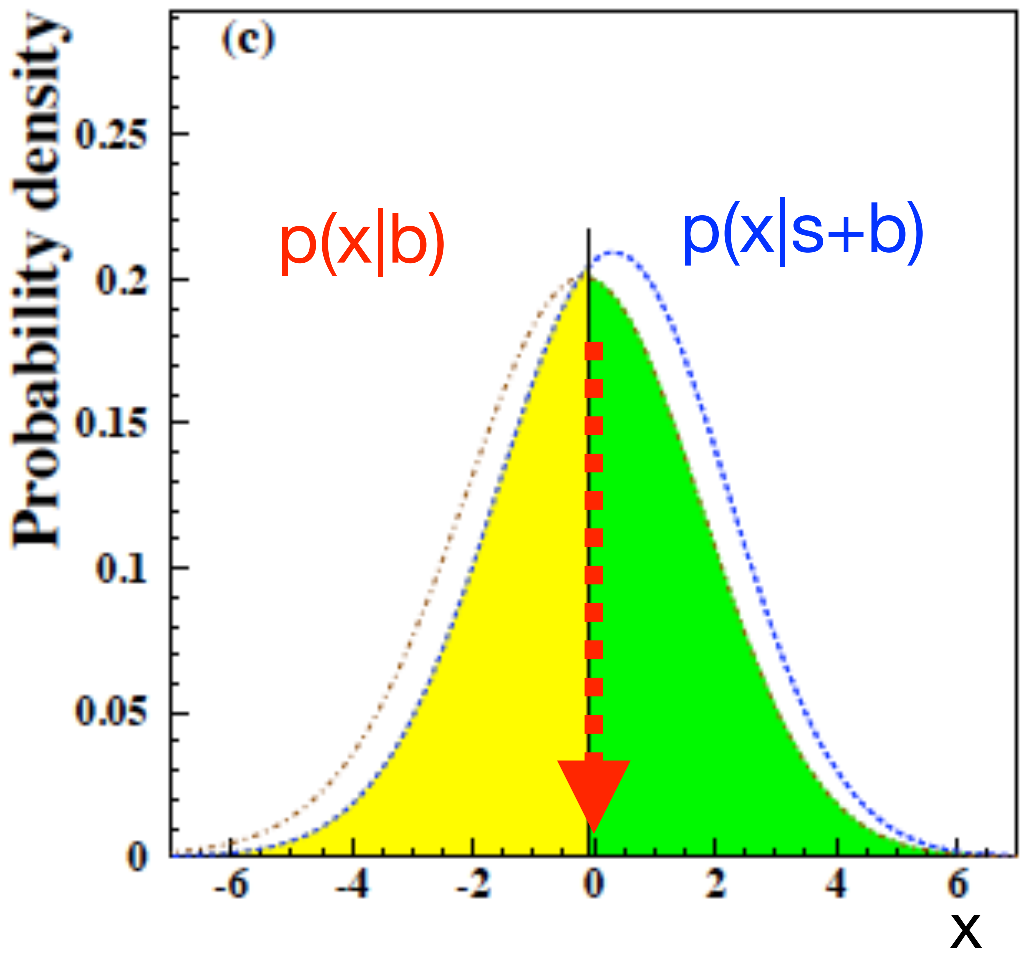
LR observed in data



Issues with p-values



Possible to get an observation that rejects both the null and the signal hypotheses



For small signals with poor S-vs-B separation, sensitivity is low, which means that distributions of test statistics are nearly equal. Can make no statement about the signal, regardless of the outcome

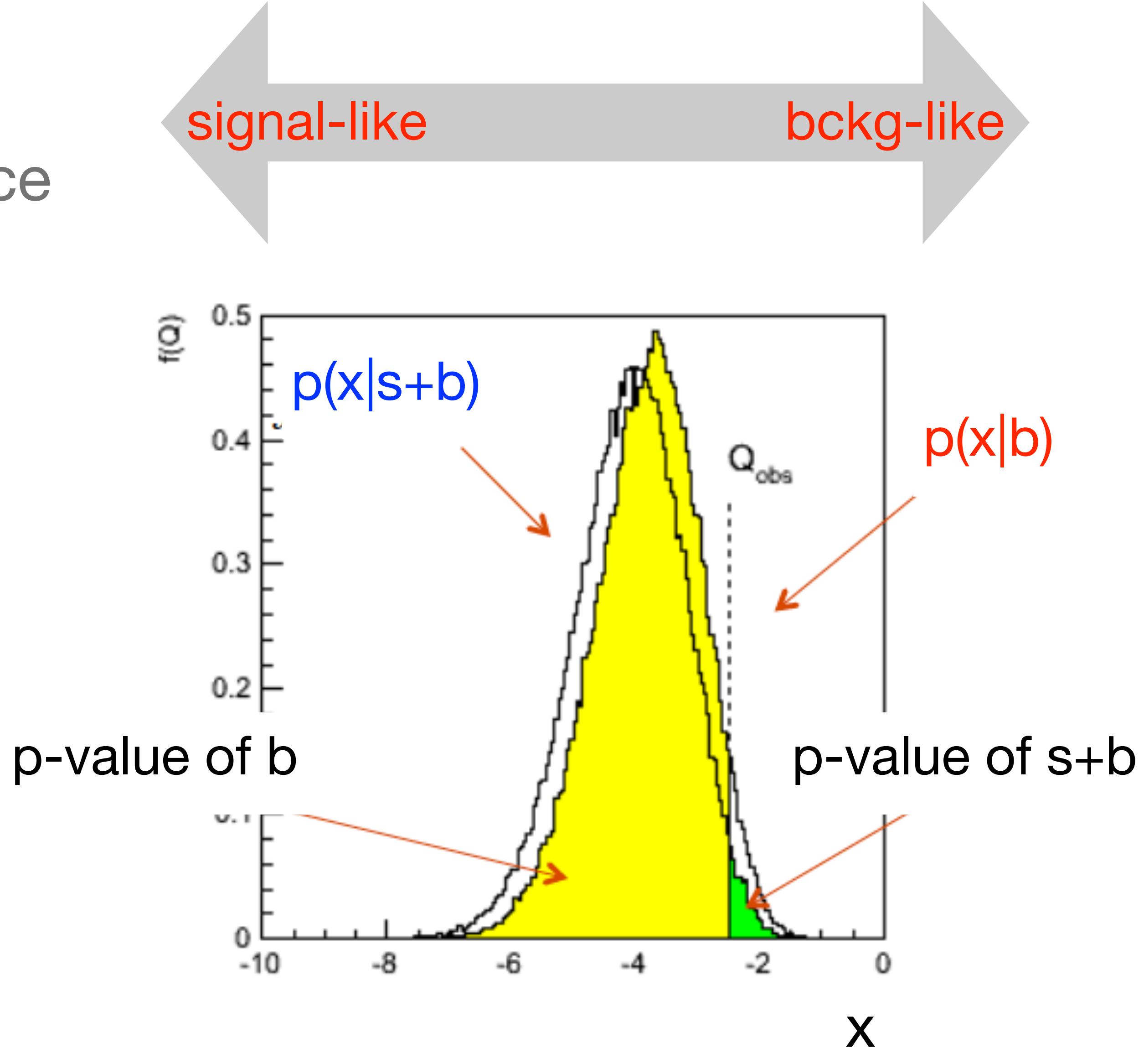
[Junk]

Spurious exclusion

Use the likelihood ratio x to test the presence of a signal p(x|s+b).

Typically, if p-value of the hypothesis s+b is smaller than 5%, signal gets excluded with 95% CL.

However, when the distributions of the test statistic are similar, (1-pvalue) of the background hypothesis is just marginally higher than p-value of s+b.



signal-like

bckg-like

p(x|s+b)

p(x|b)

p-value of b

p-value of s+b

x

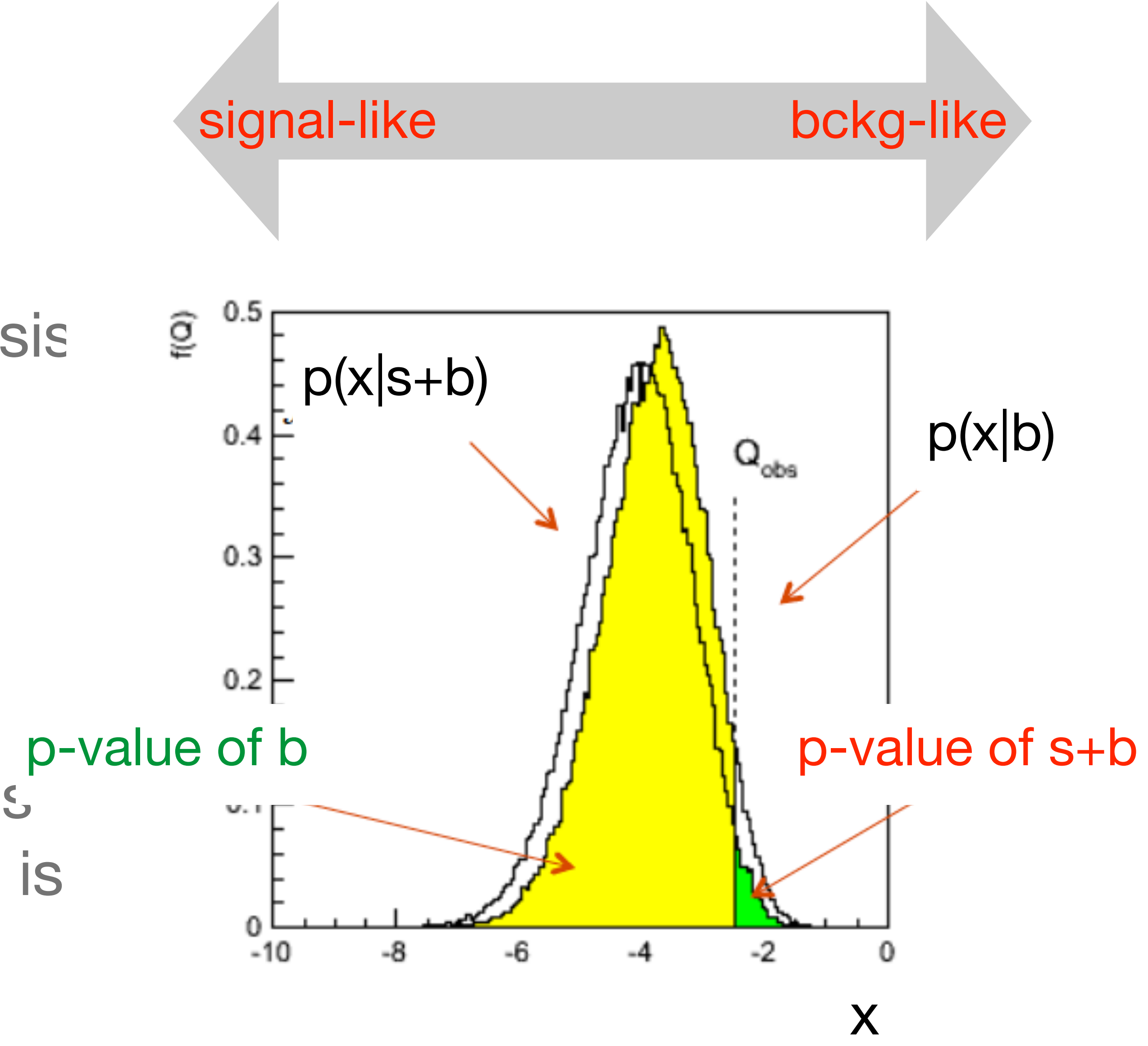
The CLs method

Scaling the p-value prevents from excluding hypotheses to which there is no sensitivity.

Base test on the pvalue for the s+b hypothesis scaled by (1-pvalue of b). Exclude only if

CLs = [pvalue for s+b] / [1 - pvalue of b]

is small. Denominator increases the CLs thus preventing excluding signals for which there is no sensitivity.



When quoting limits, it’s good practice to assess the analysis sensitivity in terms of median expected limits based on ensembles of simulated experiments or asymptotic formulae if applicable to your case

Duality

Given an ordering, there is a one-to-one correspondence between hypothesis testing and construction of confidence intervals.

It is the same problem.

Testying if parameter m equals m_0 or rather any other value, with a chosen false-positive rate = pvalue, corresponds to checking if m_0 is included in the confidence interval for m with CL=1-(pvalue)

Subtends why the likelihood-ratio based ordering of Feldman and Cousins is a generalized and powerful criterion for construction of confidence intervals, thansk to the NP-lemma.

What?

Giving computers the ability to learn without explicitly programming them

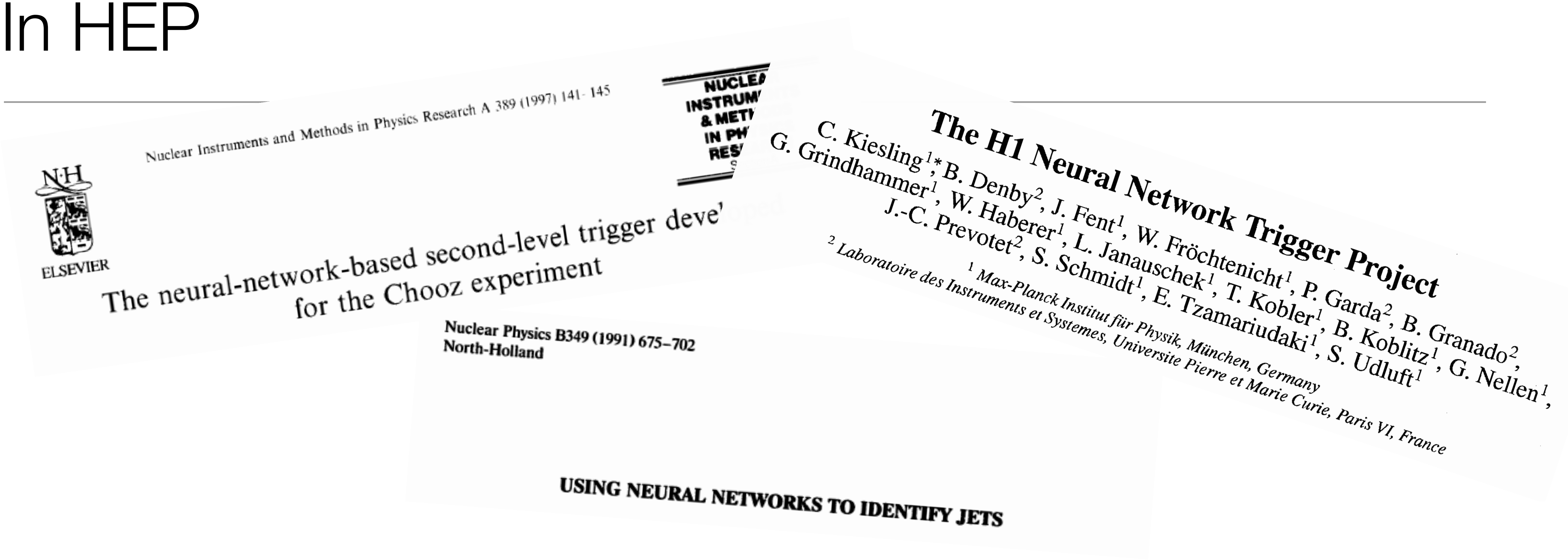
Use statistics, mathematics, and computer science to determine mathematical models, learned from data, that capture the patterns and relationships between the features of the data.

Formulated around the 1950ies.

Currently rapidly evolving, driven by many relevant applications in language processing, speech and handwritring recognition, vision, computer vision, fraud detection, financial markets analysis, search engines, spam/virus detection, medical diagnosis, robotics, automation, advertising, data science.

Statistical learning

In HEP



Nuclear Instruments and Methods in Physics Research A 389 (1997) 141- 145

N·H

ELSEVIER

The neural-network-based second-level trigger deve'

for the Chooz experiment

Nuclear Physics B349 (1991) 675–702
North-Holland

USING NEURAL NETWORKS TO IDENTIFY JETS

The H1 Neural Network Trigger Project

C. Kiesling[1]*, B. Denby[2], J. Fent[1], W. Fröchtenicht[1], P. Garda[2], B. Granado[2], G. Grindhammer[1], W. Haberer[1], L. Janauschek[1], T. Kobler[1], B. Koblitz[1], G. Nellen[1], J.-C. Prevotet[2], S. Schmidt[1], E. Tzamariudaki[1], S. Udluft[1]

[1] *Max-Planck Institut für Physik, München, Germany*

[2] *Laboratoire des Instruments et Systemes, Universite Pierre et Marie Curie, Paris VI, France*

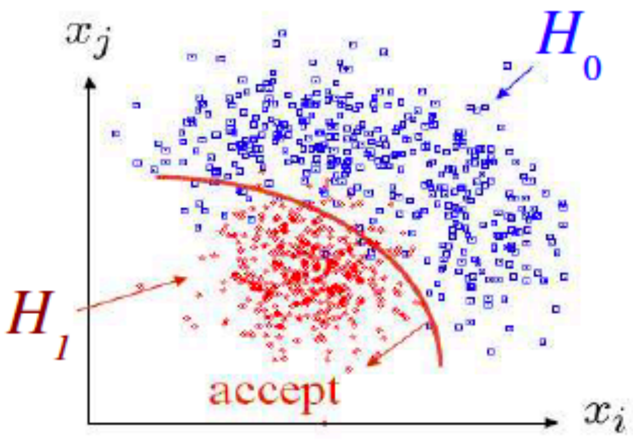
First, pioneering applications appeared in the 90ies

Became more popular in the 2000’s (LEP/Tevatron) until today’s boom: classify “signals” from “backgrounds” both online and offline, improve reconstruction of heavy particles from incomplete decay products, etc..

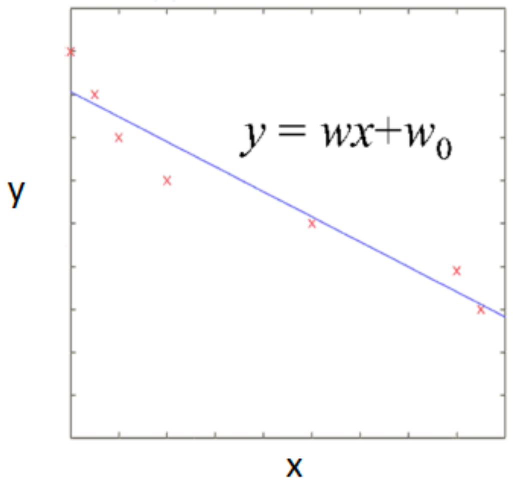
The model

Central assumption: observed data are generated from the probabilistic distribution p(x|m), the “model”, which is a mathematical description of the system of interest. The model depends on the data and on what we want to accomplish, e.g.:

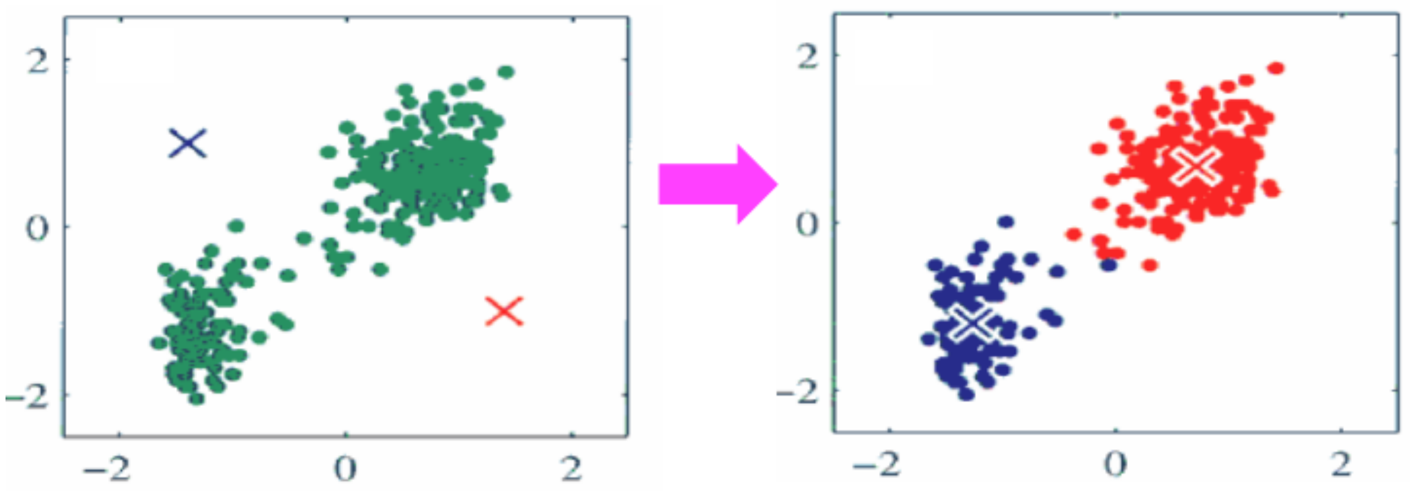
Classification



Regression



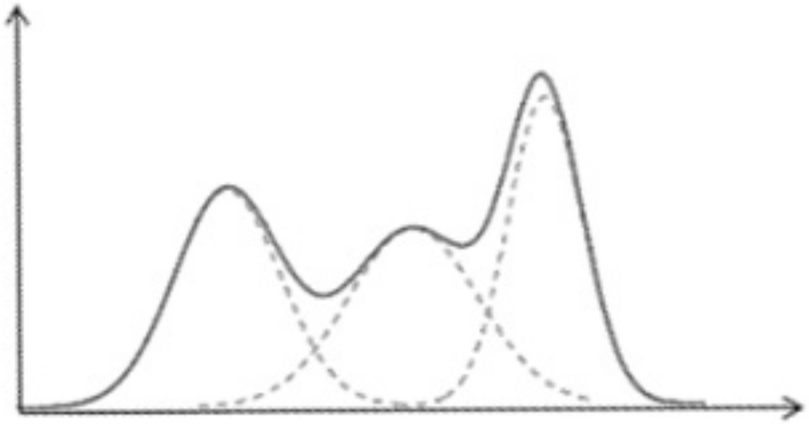
Clustering



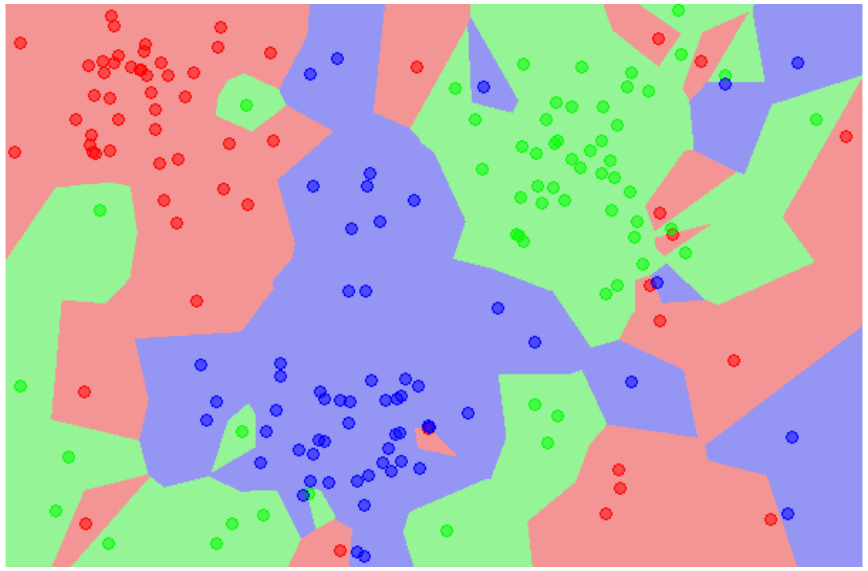
An approximation of the model is learned by using the information associated with input data. It is then used to identify the relevant properties of the system of interest and predict new data points.

Parametric vs non-parametric

Parametric models — fully specified by a number of parameters that does not grow with the size of the data set used to extract them. E.g, Gaussian mixture models



Non-parametric models — may grow in complexity with more data. E.g., a model that predicts the location of a data point in the feature space using the nearest known set of data points



Supervised learning

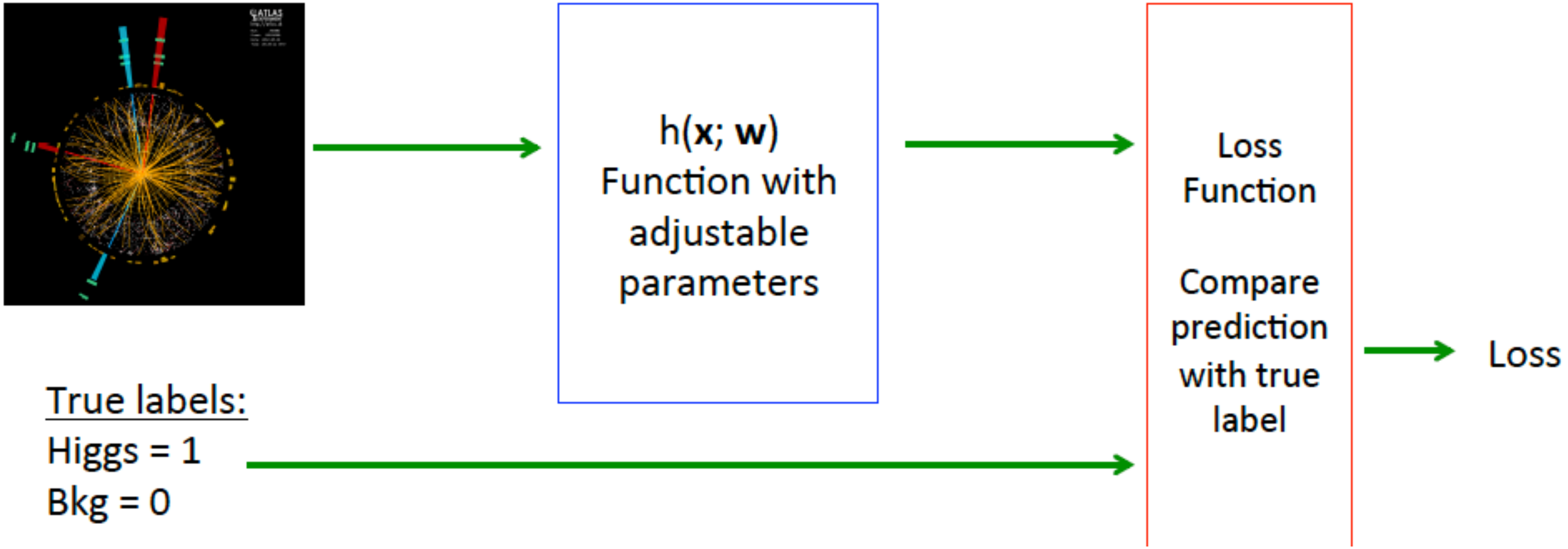
Define a model $h(\vec{x}|\vec{w})$ flexible enough to be able to adapt to the problem at hand (but not more flexible than that)

Feed a set of “training” data $\vec{x}_t$ to the model so that it can “learn” (adjust its parameters) for modeling any new data optimally [for the task]: give it N example events, each associated with feature variables $\vec{x}$ and the label (or target) y. This is the value of the quantity I want the model to predict — can be a class label (signal, background or pion, kaon) or a real number (electron energy..).

During learning iterate over the training data by adjusting the model-parameters w until a “distance” figure of merit that quantifies the difference of the model from the truth reaches a sufficiently low value. Define $h(\vec{x}) = y$.

Test performance on an independent labeled sample

Supervised learning



h(**x**; **w**)
Function with adjustable parameters

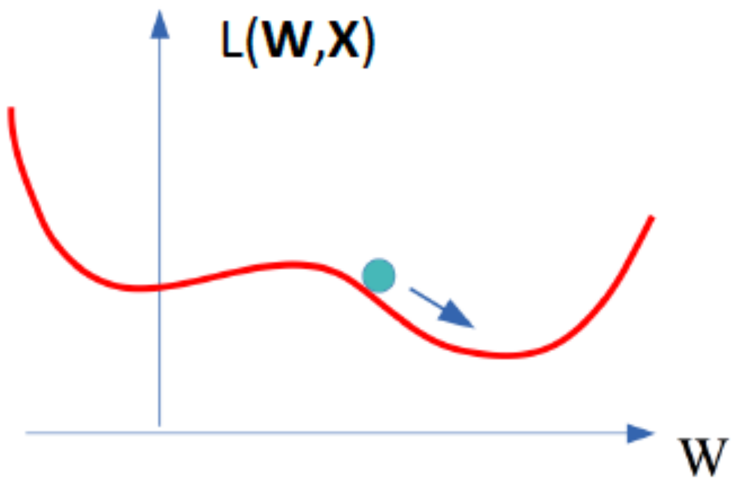
Loss Function

Compare prediction with true label

Loss

True labels:
Higgs = 1
Bkg = 0

- Design function with adjustable parameters
- Design a Loss function
- Find best parameters which minimize loss
  - Use a labeled *training-set* to compute loss
  - Adjust parameters to reduce loss function
  - Repeat until parameters stabilize
- Estimate final performance on *test-set*



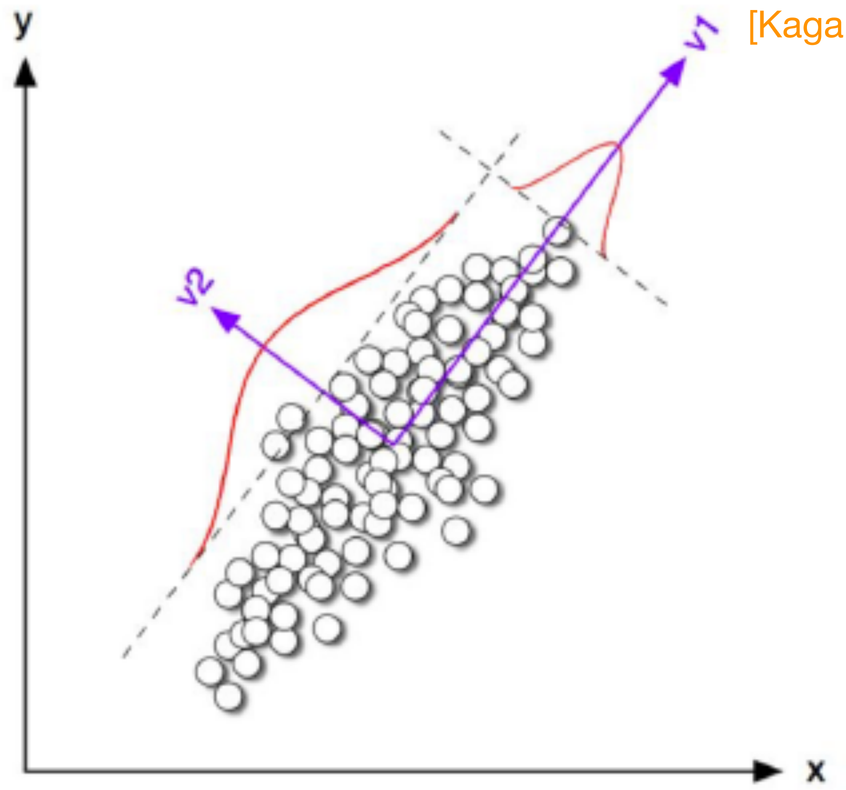
[Kagan, Le Cun]

Unseupervised learning

As before, but labels are not known. The task is to find structure/pattern in the data.

Clustering: partition data into subsets according to similarities in the feature space

Dimensionality reduction: find a lower-dimensional (simpler) representation of the data



[Kagan]

What follows

In HEP, ML approaches have been mostly applied to classification: signal/ background, kaon/pion, photon/electron; quark-jet/gluon-jet, b-jet/ light-jet. Typically supervised due to availability of simulated and control samples

With the LHC, applications to a broader set of tasks are becoming popular (e.g., reweight multidimensional distributions to match to each other)

Our general discussion will be mostly restricted to supervised binary classification

The classification task

In a sample of physics data, observe candidate “signal” events, contaminated by “background” events. Each is associated to a set $\vec{x}$ of variables (or features or predictors) e.g.,

x_1 = transverse momentum

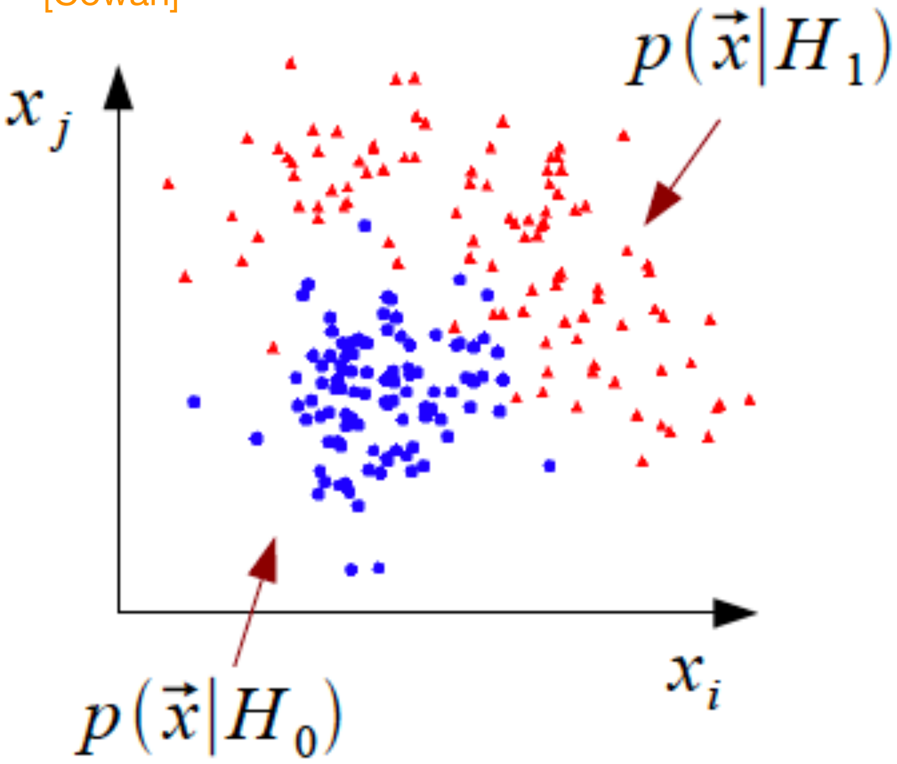
x_2 = displacement from collision point

x_3 = …

x_n = …

$\vec{x}$ is distributed according to an n-dimensional joint probability density $p(\vec{x}|m)$, which differs for signal (H_1) and background candidates (H_0).

[Cowan]

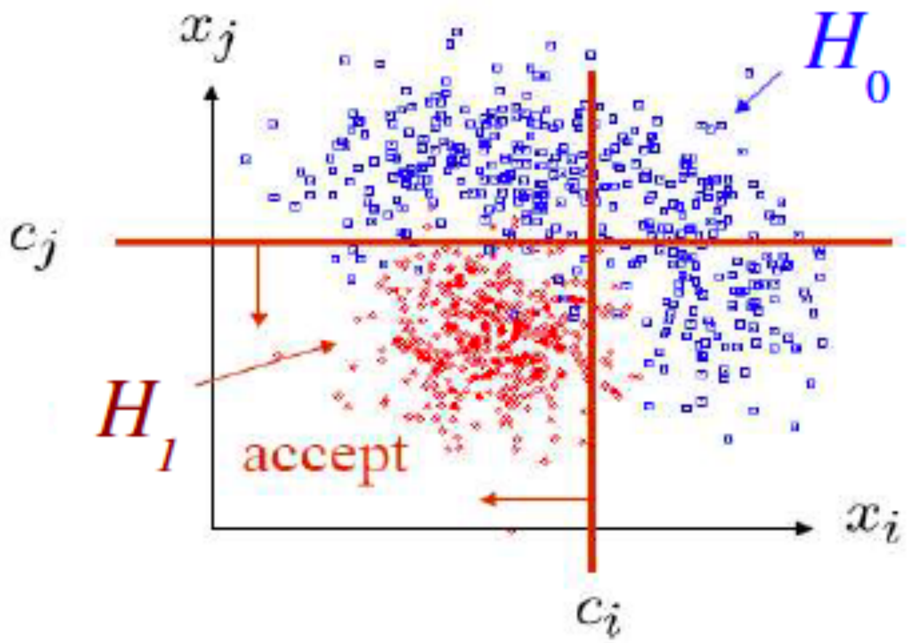


The goal is to classify the events within the signal or background categories.

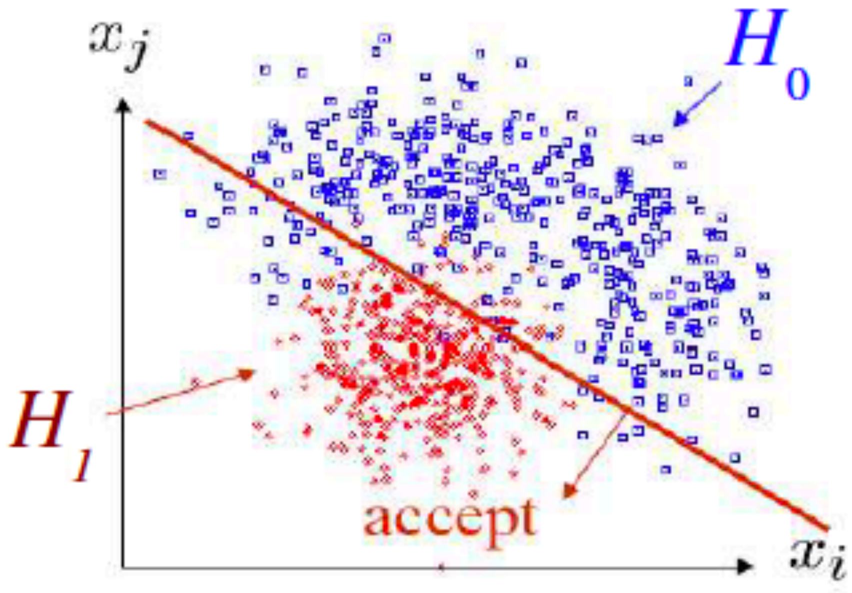
Decision boundaries

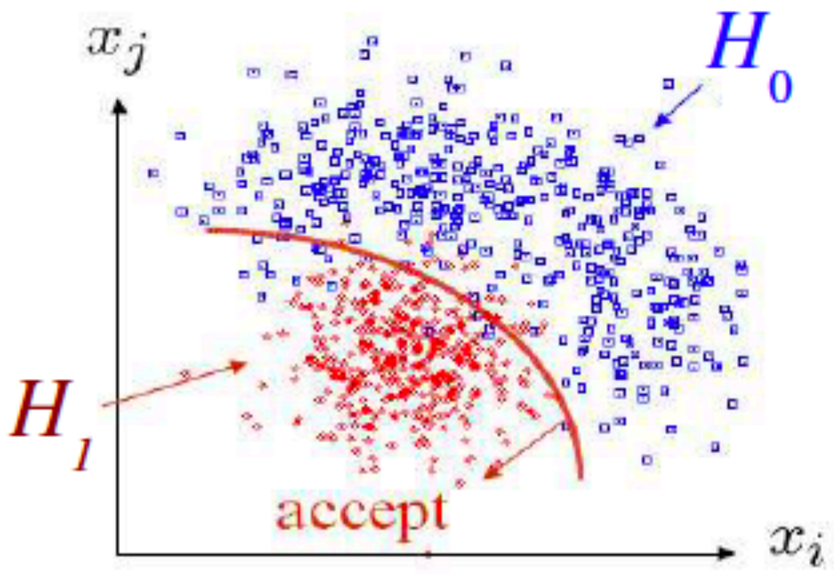
Can do it with cuts

[Cowan]



Or identify some sort of decision boundary





[Cowan]

Decision boundaries

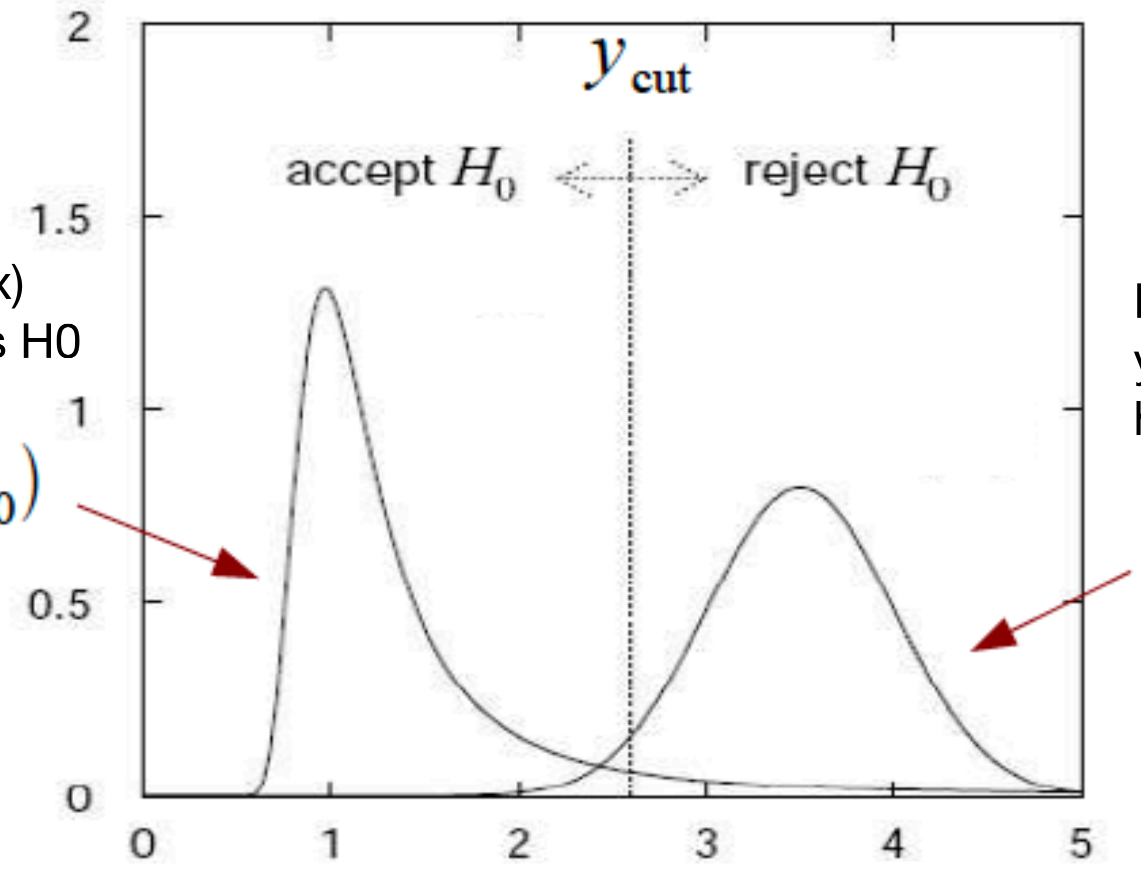
Decision boundary — a function of the data that allows separation between classes. Surface in the n-dimensional space of the features.

Can make $y(\vec{x})$ as a scalar number and determine it in a way that its distributions for the signal and background samples are maximally separated.

With such a dimensionality reduction, a “cut” on $y(\vec{x})$ offers a decision boundary

Distribution of y(x) under hypothesis H0

$f(y|H_0)$



Distribution of y(x) under hypothesis H1

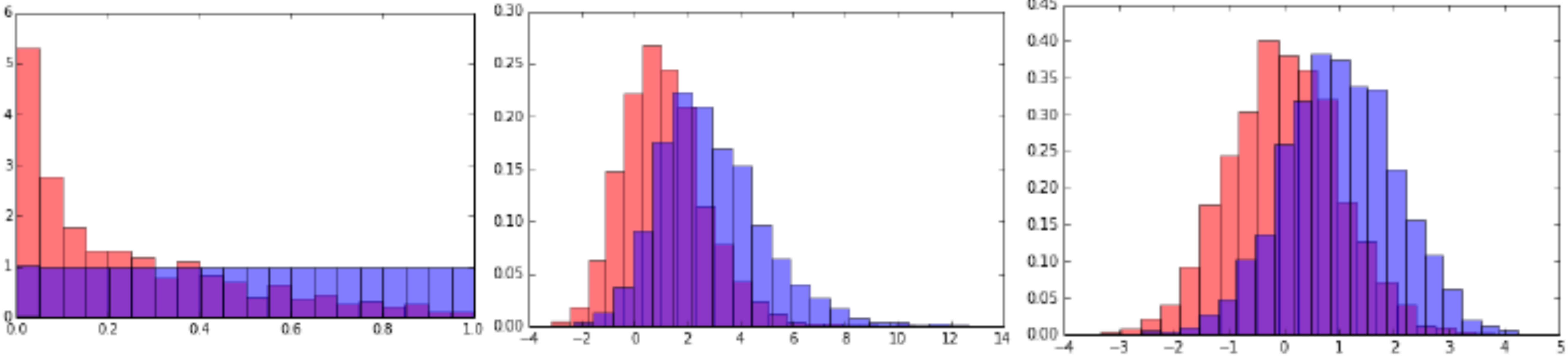
$f(y|H_1)$

y(x)

↑

Data

Binary classification performance



Rogozhnikov

Three classifiers separate “red” from “blue” classes of events.

Which one does it better?

ROC

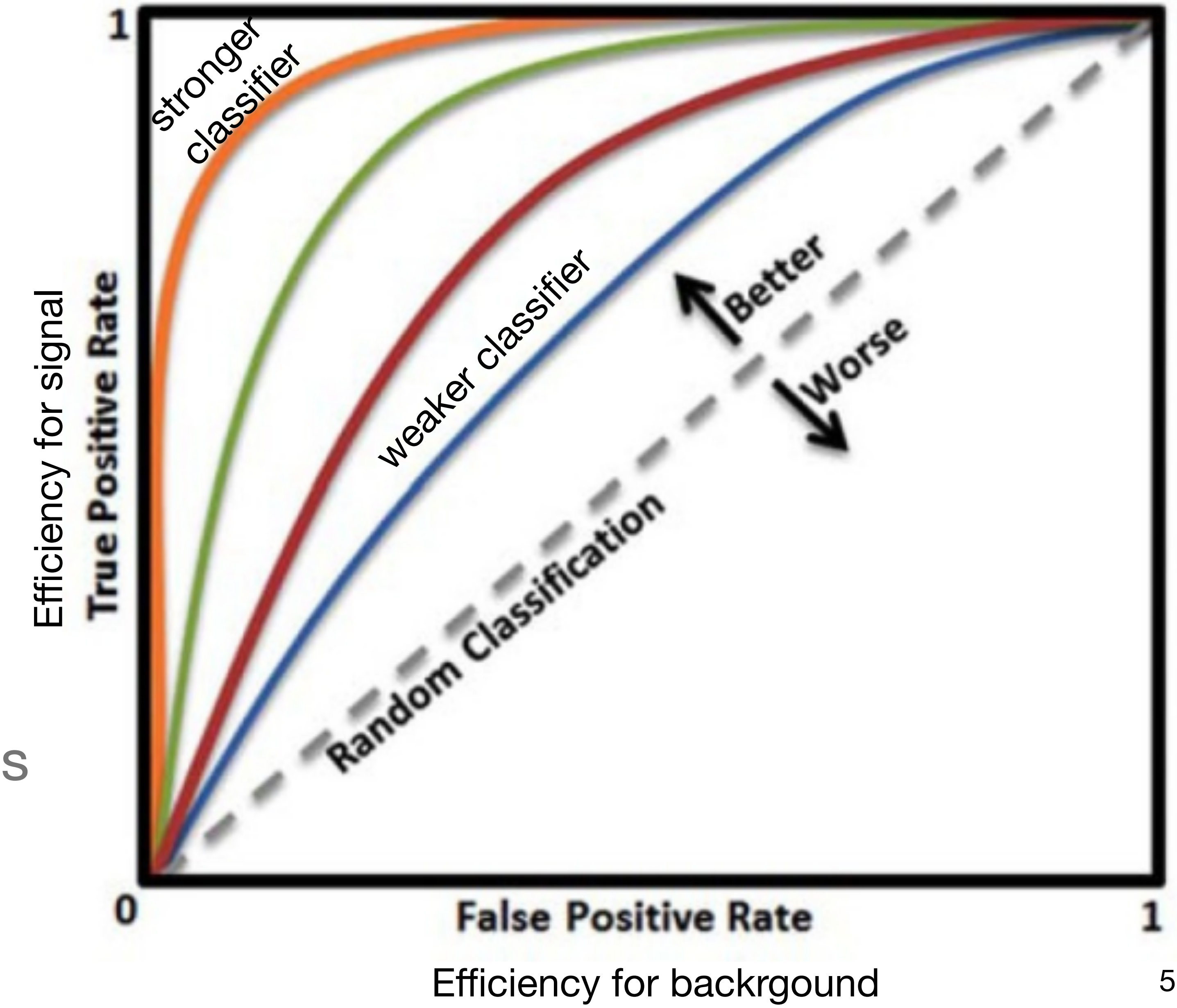
None: all have the same receiver operating characteristic (ROC) curve: signal classification efficiency vs background misclassification efficiency.

Standard measure of performance for binary classifiers.

Each point in the curve corresponds to a threshold in the classifier output.

Get as much top right as possible.

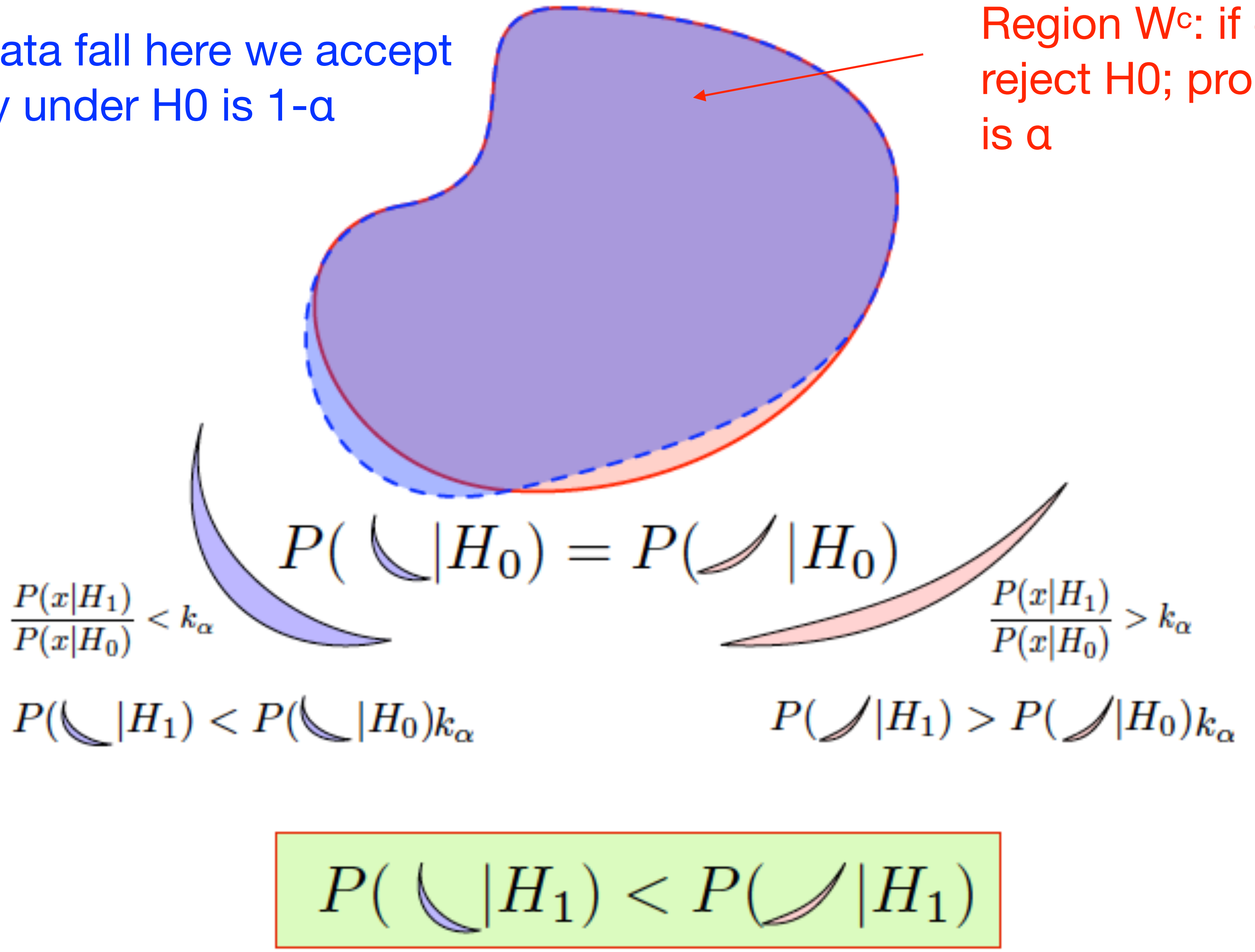
Is there any optimal variable that, given the information in data, allows separating two classes of events with minimum false positive rate at given true positive rate?



Yes

Region W: if data fall here we accept H0; probability under H0 is 1-α

Region W^c: if data fall there we reject H0; probability under H0 is α


$\frac{P(x|H_1)}{P(x|H_0)} < k_\alpha$
$\frac{P(x|H_1)}{P(x|H_0)} > k_\alpha$

The new region region has less power.

[Cranmer]

Neyman-Pearson Lemma — remember?

The optimal variable exist and it is the likelihood ratio

For any false positive rate (i.e., misclassification of true background events), the region W of acceptance of H_0, which minimizes the probability to accept H_0 when H_1 (or, to classify as background a true signal event) is true, is a contour (a cut, in 1D) of the likelihood ratio.

$$\frac{p(x|H_1}{p(x|H_0)} > k_\alpha$$

Therefore the optimal decision boundary is (where x can be multidimensional)

$$y(\vec{x}) = \frac{p(\vec{x}|H_1)}{p(\vec{x}|H_0)} = \frac{p(\vec{x}|s)}{p(\vec{x}|b)}$$

(or any monotonic function of it)

Problem

Rarely the densities p($\vec{x}$|H_1) and p($\vec{x}$|H_0) that are needed to evaluate the likelihood ratio for each event are known.

Most of the supervised machine-learning classification task boils down to use the data to find the best approximation of the likelihood ratio

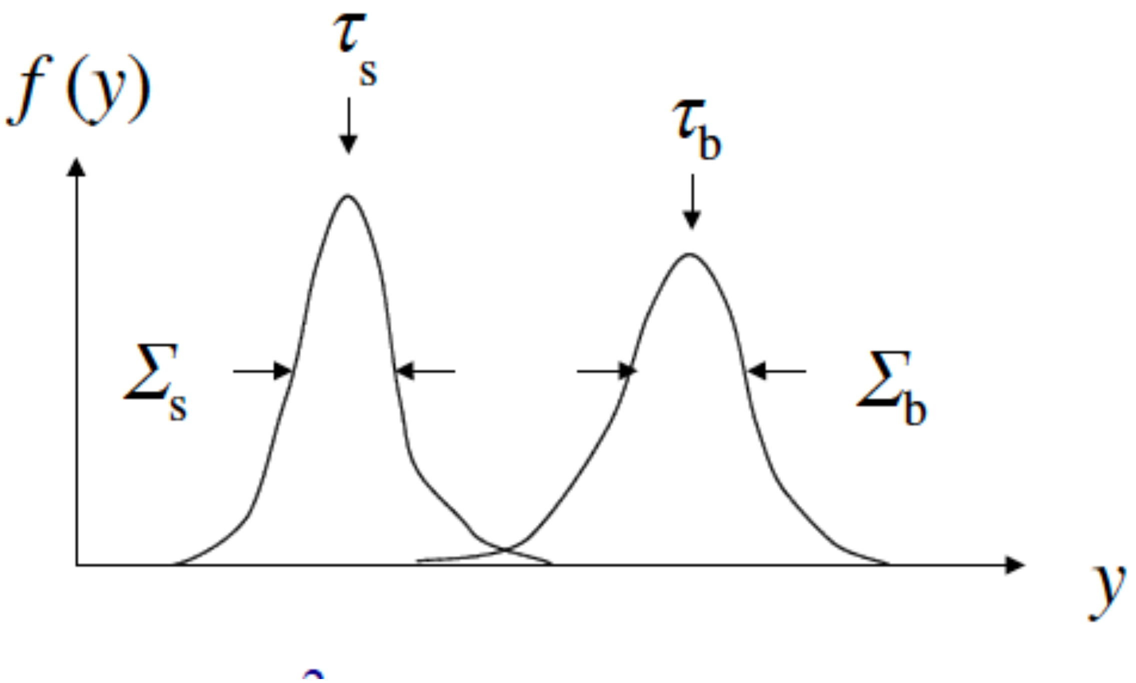
Guessing the density

Simple guess: assume y to be a linear function of the features

$$y(\vec{x}) = \sum_{i=1}^{n} w_i x_i$$

and find the coefficients w_i that maximize the separation between the distributions of y(x) on signal and background events:

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



Fisher’s discriminant

[Cowan]

mean, covariance of $\boldsymbol{x}$

We have

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

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

where $k = 0, 1$ (hypothesis)

and $i, j = 1, \ldots, n$ (component of $\boldsymbol{x}$)

$$J(\vec{w})=\frac{(\tau_{\mathrm{s}}-\tau_{\mathrm{b}})^2}{\Sigma_{\mathrm{s}}^2+\Sigma_{\mathrm{b}}^2}$$

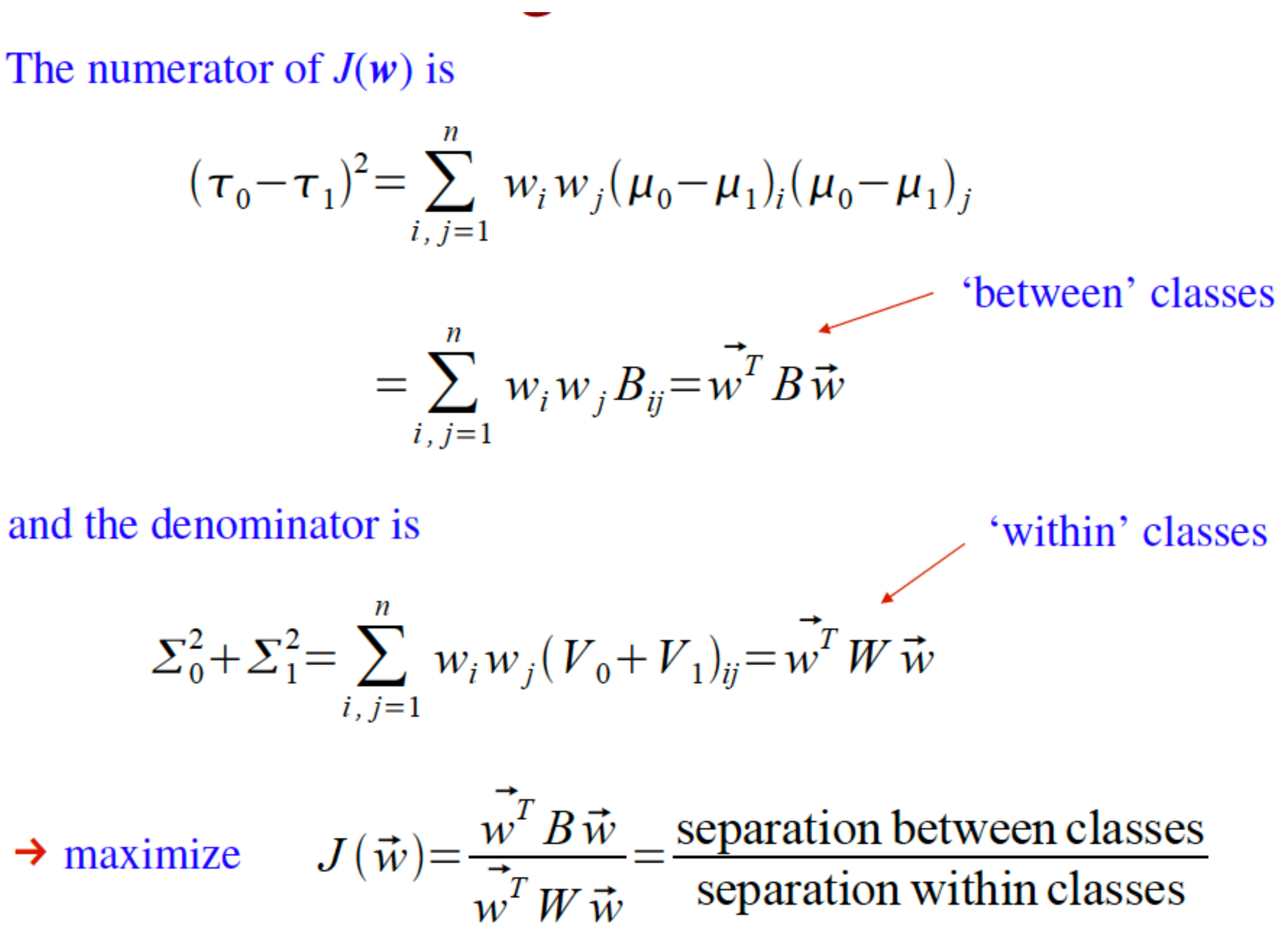
For the mean and variance of $y(\vec{x})$ we find

$$\tau_k=\int y(\vec{x})\, p(\vec{x}|H_k)\, d\vec{x}=\vec{w}^T \vec{\mu}_k$$

$$\Sigma_k^2=\int (y(\vec{x})-\tau_k)^s\, p(\vec{x}|H_k)\, d\vec{x}=\vec{w}^T V_k \vec{w}$$

Fisher’s discriminant

[Cowan]



The numerator of $J(\boldsymbol{w})$ is

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

‘between’ classes

$$= \sum_{i,j=1}^{n} w_i w_j B_{ij} = \vec{w}^T B \vec{w}$$

and the denominator is

‘within’ classes

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

→ maximize $\quad J(\vec{w}) = \dfrac{\vec{w}^T B \vec{w}}{\vec{w}^T W \vec{w}} = \dfrac{\text{separation between classes}}{\text{separation within classes}}$

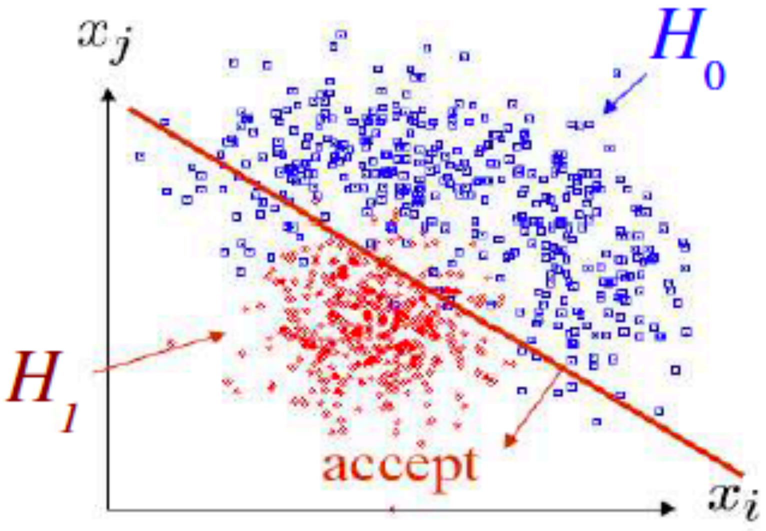
Fisher's discriminant

[Cowan]

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

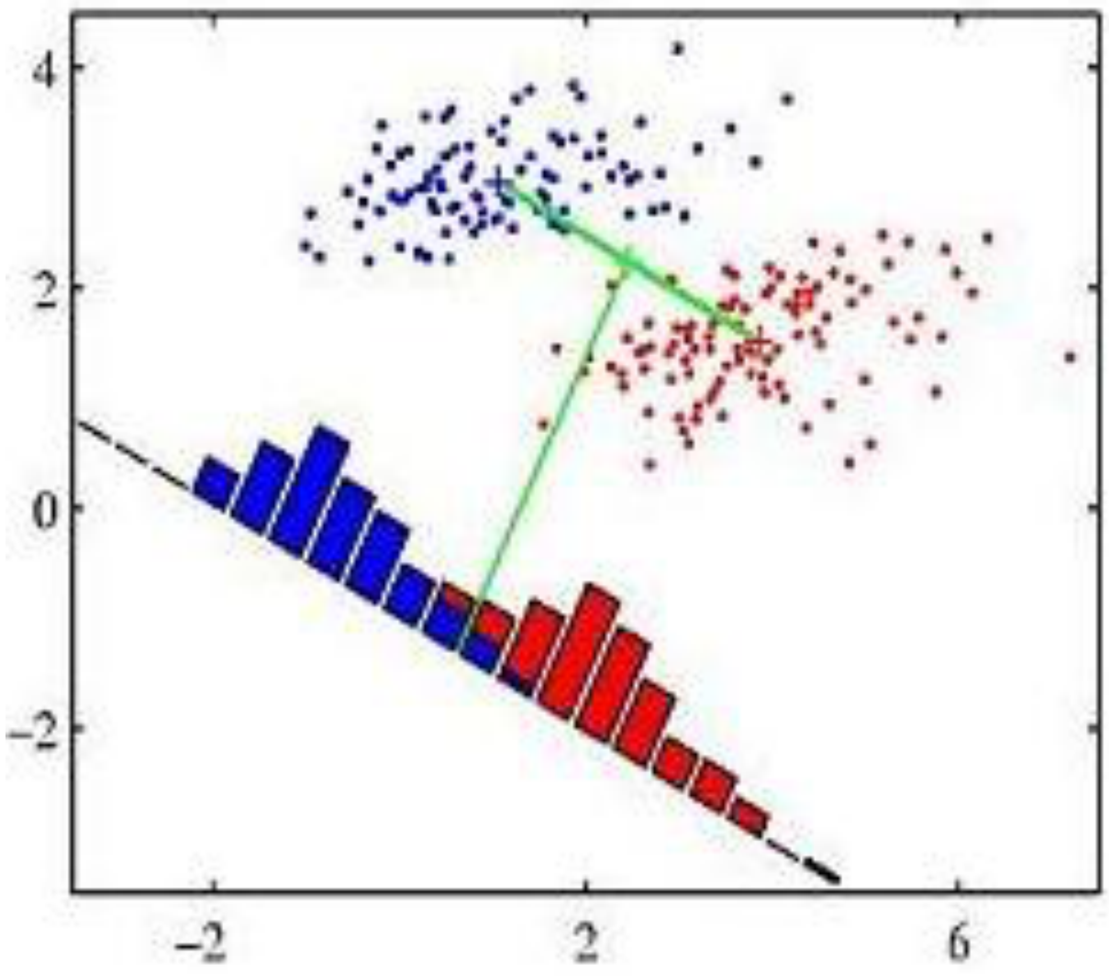
$$\boxed{y(\vec{x}) = \vec{w}^T \vec{x} \quad \text{with } \vec{w} \propto W^{-1}(\vec{\mu}_0 - \vec{\mu}_1)}$$

The resulting weights define the linear decision boundary such that the projection of the points along the tangent of the boundary produces maximally separated distributions.

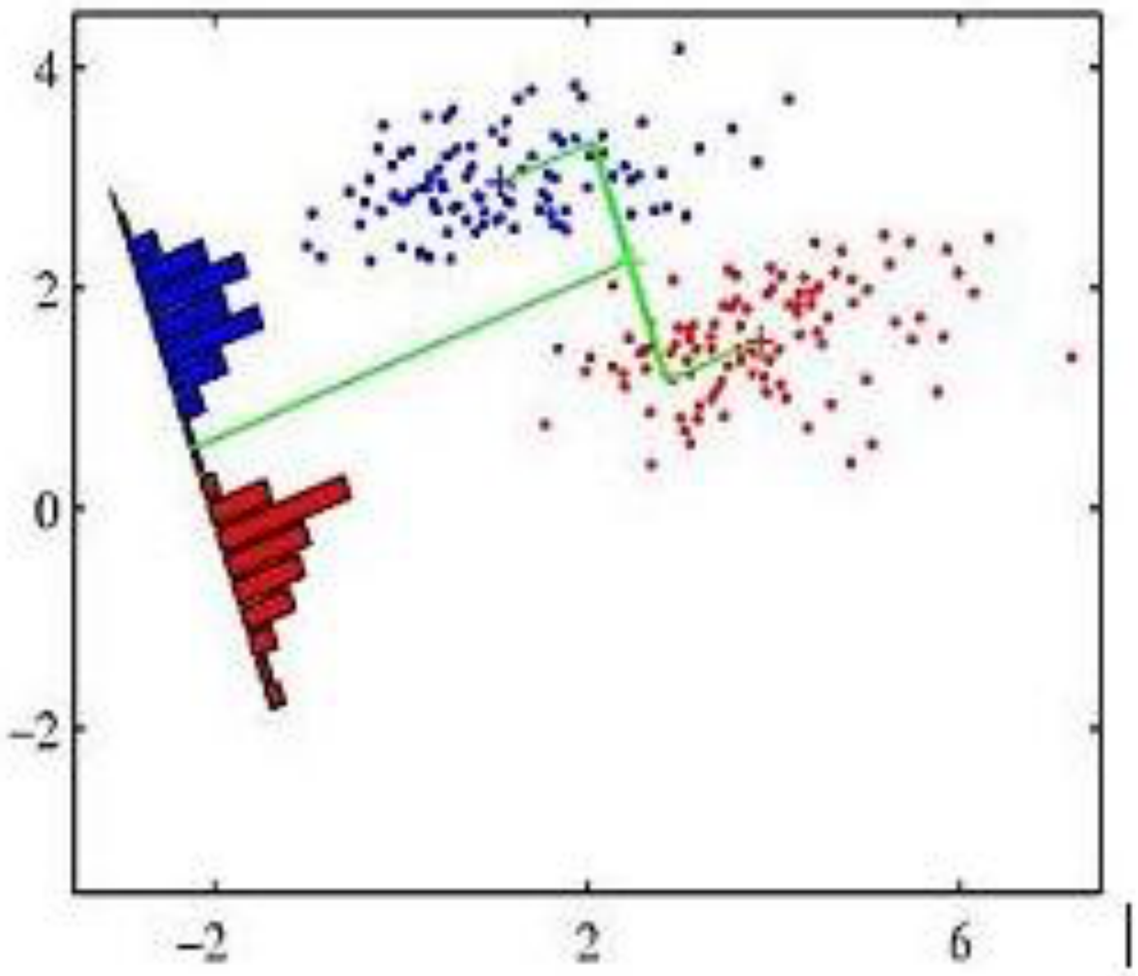


Fisher’s discriminant

Suboptimal separation



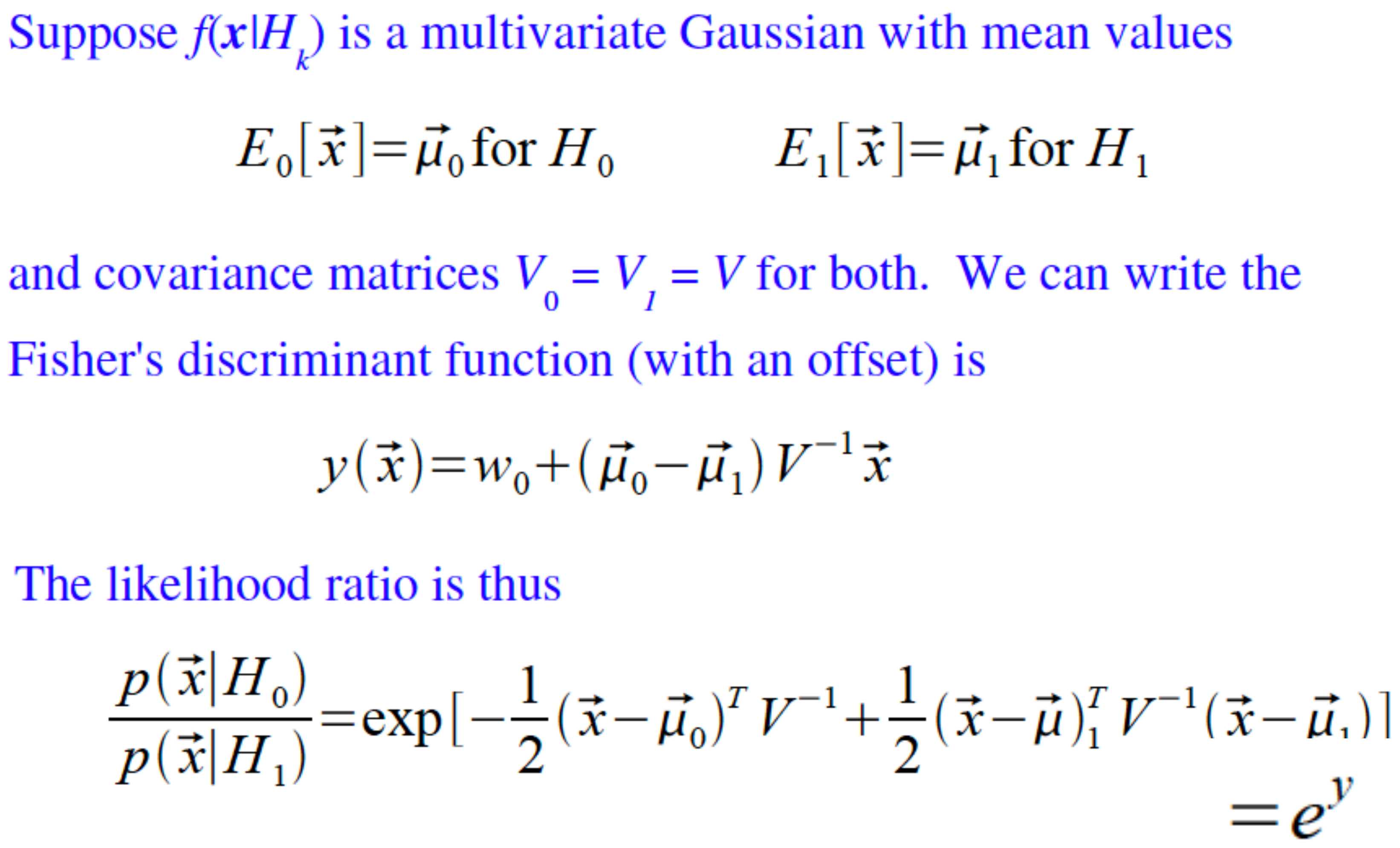
Fisher discriminant



Fisher’s discriminant

For Gaussian data with equal covariance the Fisher discriminant offers the optimal decision boundary.

[Cowan]



Suppose $f(\boldsymbol{x}|H_k)$ is a multivariate Gaussian with mean values

$$E_0[\vec{x}]=\vec{\mu}_0 \text{for } H_0 \qquad E_1[\vec{x}]=\vec{\mu}_1 \text{for } H_1$$

and covariance matrices $V_0 = V_1 = V$ for both. We can write the Fisher's discriminant function (with an offset) is

$$y(\vec{x})=w_0+(\vec{\mu}_0-\vec{\mu}_1)V^{-1}\vec{x}$$

The likelihood ratio is thus

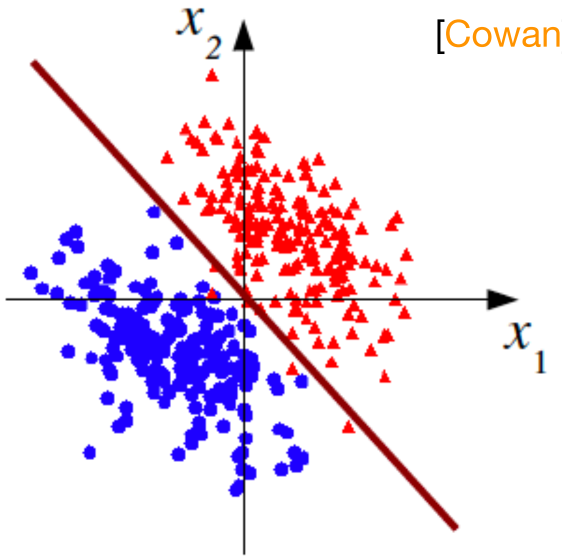
$$\frac{p(\vec{x}|H_0)}{p(\vec{x}|H_1)}=\exp[-\frac{1}{2}(\vec{x}-\vec{\mu}_0)^T V^{-1}+\frac{1}{2}(\vec{x}-\vec{\mu})_1^T V^{-1}(\vec{x}-\vec{\mu}_1)]$$

$$=e^y$$

The Fisher’s discriminant is a monotonic function of the likelihood ratio and is therefore optimal (for Gaussian data with equal covariance)

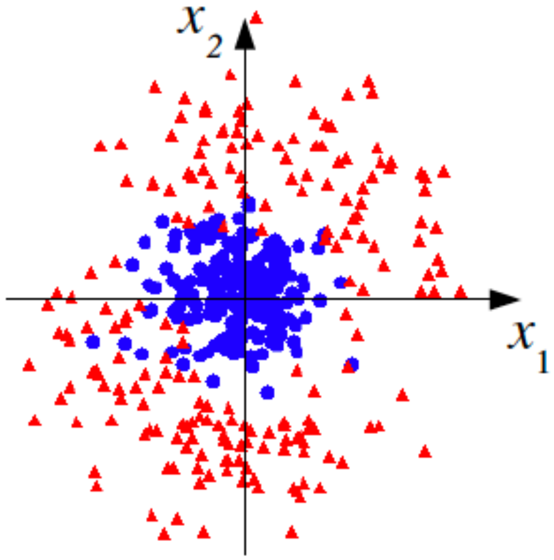
Limitations of linear boundaries

A linear decision boundary is optimal when the classes of events to be separated are distributed as multivariate Gaussians with same covariance and differing mean



[Cowan]

When data are non-Gaussian, linear decision boundaries can fail.



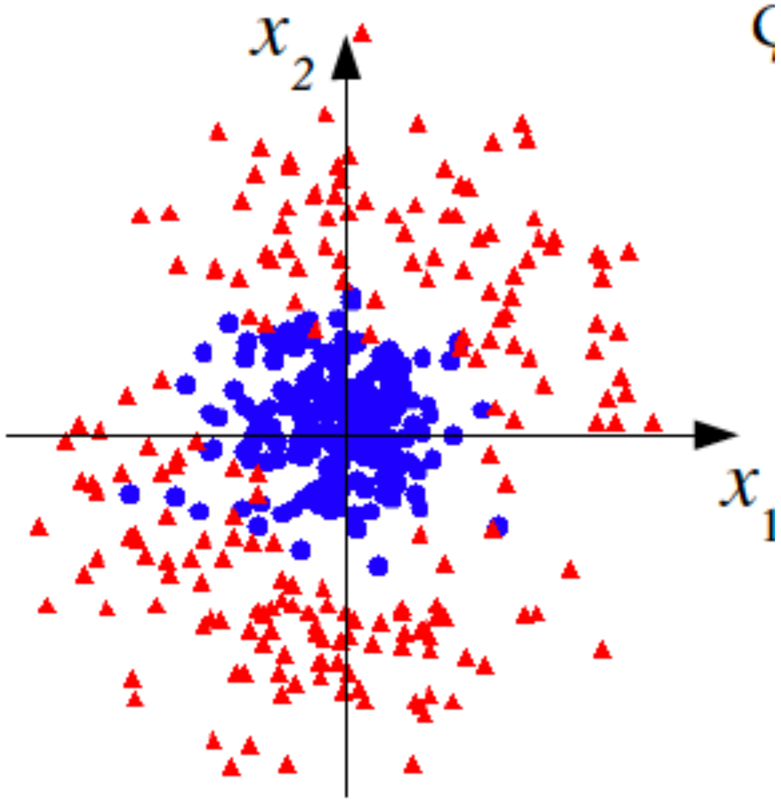
Limitations of linear boundaries

Occasionally in simple problems, a nonlinear transformation that maps the feature space into variables that are more likely to be linearly separable is evident

$x_1 \ldots\ldots x_n \;==>\; \phi_1(\vec{x})\ldots\phi_n(\vec{x})$

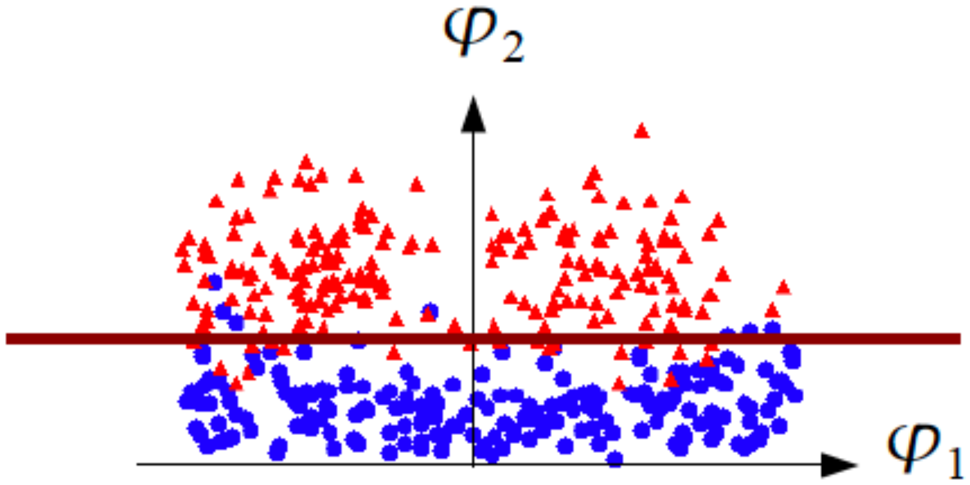
[Cowan]

x_1, x_2



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

$\phi_2(x_1, x_2) = (x_1^2 + x_2^2)^{1/2}$



In general, the functions of the feature space $\vec{\phi}(\vec{x},\vec{w})$ depend also on free parameters $\vec{w}$.

Nonlinear discriminants

In general, the set of basis functions in the feature space that allows/optimizes the classification is not evident.

A number of approaches offer algorithms to identify and parametrize such basis functions to offer effective classification.

Among the most commonly used nonlinear discriminants in HEP are artificial *neural networks* (some similarities with neuronal functionality)

Used in HEP since the early 80's — quite some time after the initial works by McCulloch and Pitts (1943) and Rosenblatt (1962).

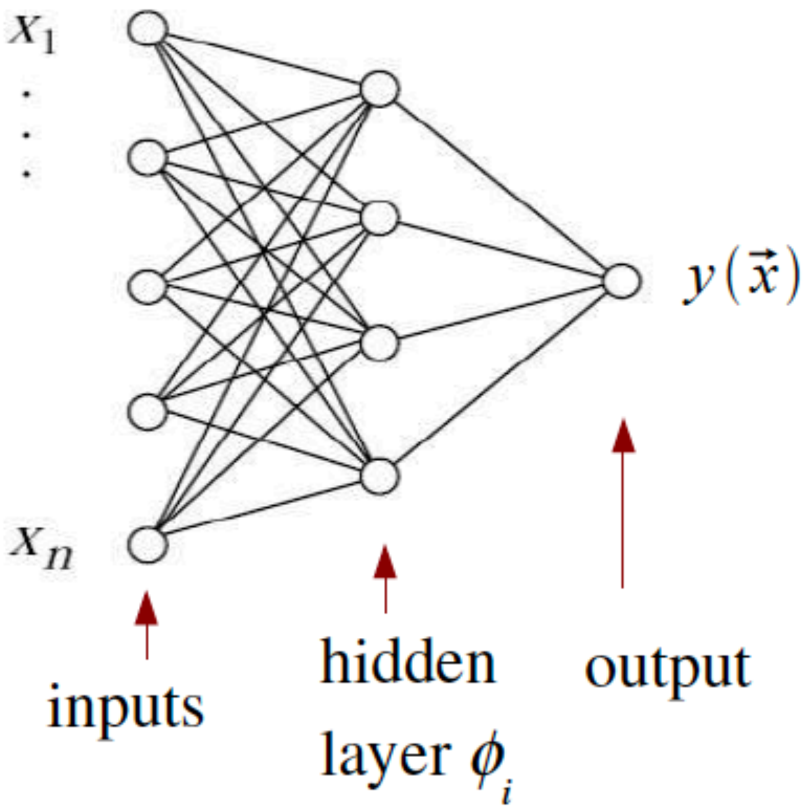
Artificial Neural Networks

Define a number of input “nodes” (driven by the dimensionality of feature space $\vec{x}$) and an output $y(\vec{x})$. i.e., a scalar variable where a single cut defines a decision boundary.

Choose a number (1 to few) of intermediate “hidden” layers. In each, choose a number of nodes. More layers/nodes imply more model parameters (N).

Each node connects to the downstream nodes. The intensities of the connections are tunable weights w

Choose a monotonic nonlinear function that expresses the “excitation” of each node in response to input from the upstream nodes (e.g., $h(s) = 1/ (1+ e^{-s})$



$$N = d \times N^{(1)} + K \times N^{(L)} + \sum_{k=1} N^{(k)} \times N^{(k+1)}$$

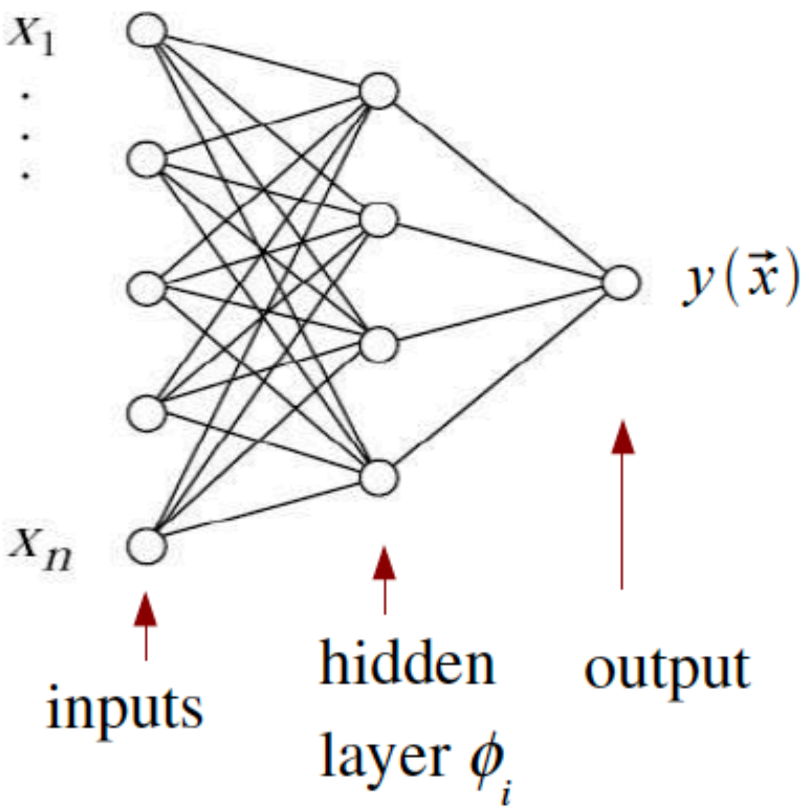
- L is the number of hidden layers
- $N^{(k)}$ is the number of nodes for layer k
- d is the input vector size, K is the output size

Artificial Neural Networks

Superscript for weights indicates layer number

$$\varphi_i(\vec{x})=h\left(w_{i0}^{(1)}+\sum_{j=1}^{n} w_{ij}^{(1)} x_j\right)$$

$$y(\vec{x})=h\left(w_{10}^{(2)}+\sum_{j=1}^{n} w_{1j}^{(2)} \varphi_j(\vec{x})\right)$$



When the ANN receives some input data, in each node, the weighted inputs incoming from the preceding nodes are fed to the activation function, which outputs to the resulting activation intensity to the following nodes.

The classification performance depends on the value of the weights. These are optimized during the training phase.

Training

Before the ANN can classify unknown events, the ANN is fed events of known classification (simulation, typically) so that the it can "learn". Each event "a" comes with its set of features $\vec{x}_a = (x_{1.....}x_n)$ and its true class $t_a = 0$ or 1.

The set of optimal weights w is obtained by minimizing an error function that quantifies how much the classification achieved by the ANN departs from the true classification (known for the training sample). Error-function example

$$E(\boldsymbol{w}) = \frac{1}{2}\sum_{a=1}^{N} |y(\vec{x}_a, \boldsymbol{w}) - t_a|^2 :$$

Minimizing E(w) over the space of weights to obtains their optimal values

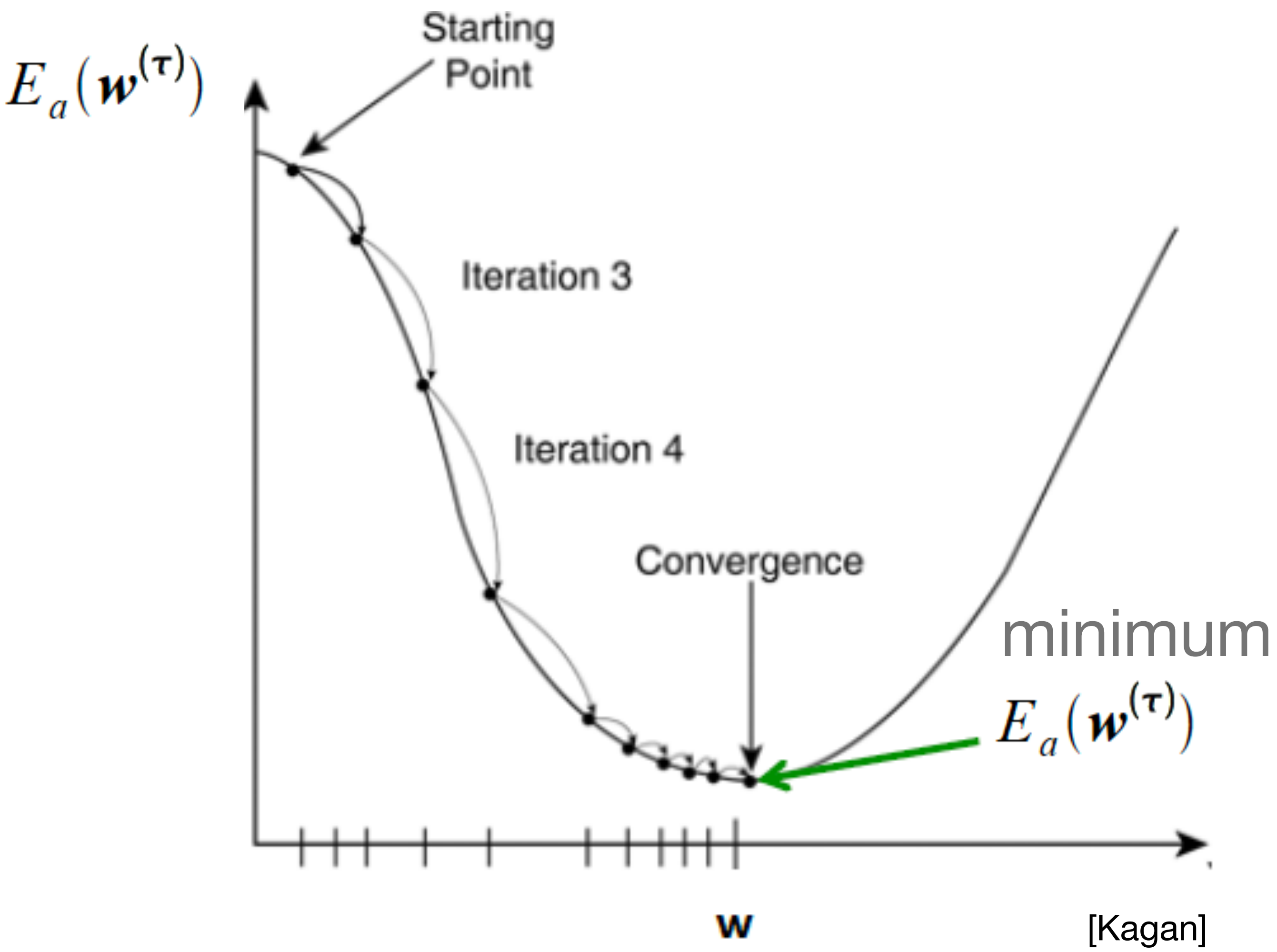
Gradient descent

The error function is minimized numerically, e.g., by using the gradient descent method: start from an initial guess (or random choice) and make a step in the direction of maximum decrease.

[Cowan]

$$\boldsymbol{w}^{(\tau+1)} = \boldsymbol{w}^{(\tau)} - \eta \nabla E_a(\boldsymbol{w}^{(\tau)})$$

Update w for each training event a.


$E_a(\boldsymbol{w}^{(\tau)})$
Starting Point
Iteration 3
Iteration 4
Convergence
minimum
$E_a(\boldsymbol{w}^{(\tau)})$
w

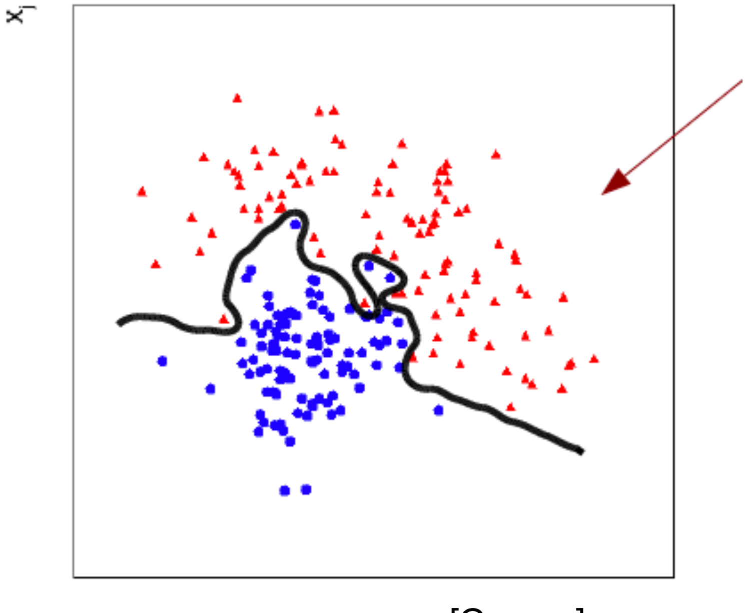
[Kagan]

Error backpropagation: determines the derivatives needed to calculate the gradient directions at each node using a recursive rule.

Overtraining

The number of inputs and inner nodes should be optimized for the problem .

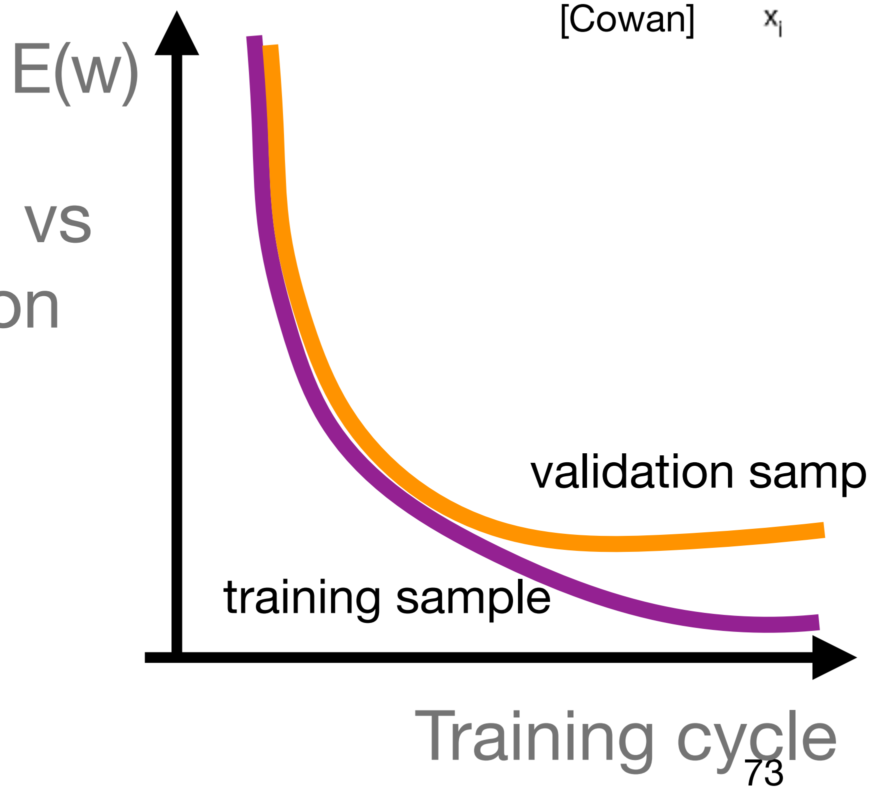
Too many nodes (i.e., free parameters) yield outputs conforming too closely the training data.



[Cowan]

Overtraining: decision boundary follows the details of the statistical fluctuations yielding an unrealistically small error rate on the training data.

Evaluate classification performance on a independent validation sample. Different behavior of the error function vs training cycle between the training sample and a validation sample indicates the onset of overtraining.

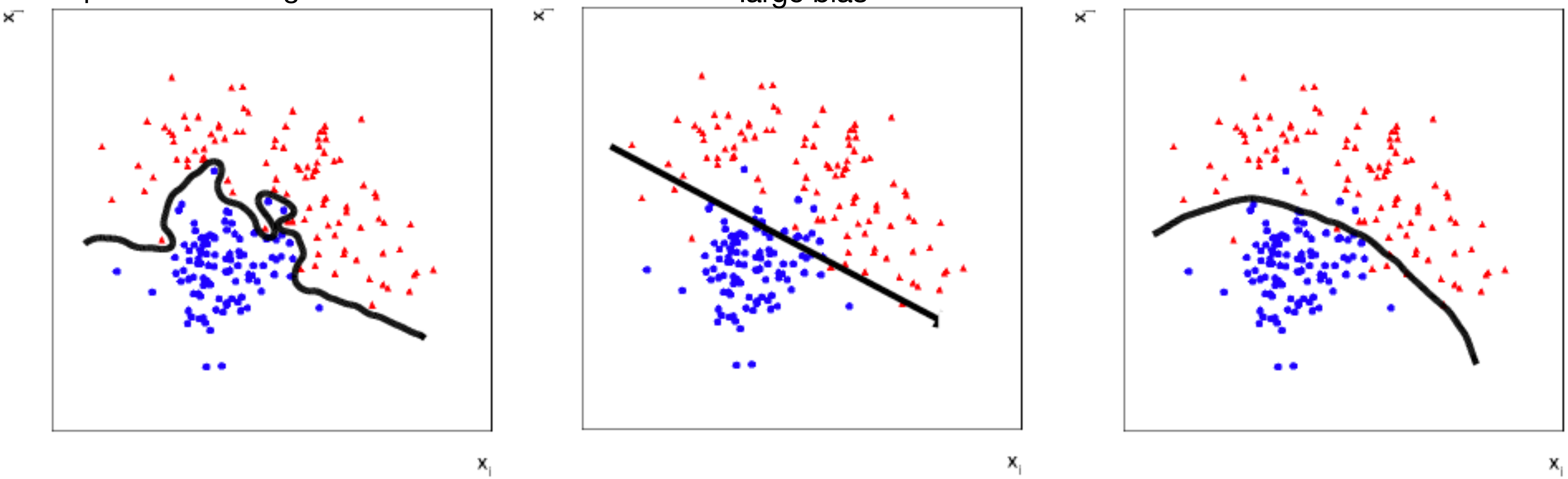


After the ANN architecture is optimized, the expected performance should be evaluated in a test sample (other than the training and validation samples).

Bias-variance tradeoff

At given training sample size, the higher the ANN complexity (more hidden layers, more nodes) the larger the statistical errors of the ANN parameters, since information from the same data is used to determine a larger number of parameters. This yields overtraining and high variance.

However, with the too few nodes, the ANN struggles to expoit the nonlinearities, yielding bias.



Too many model parameters: large variance

Too few model parameters: large bias

Tradeoff

[Cowan]

Bias-variance tradeoff

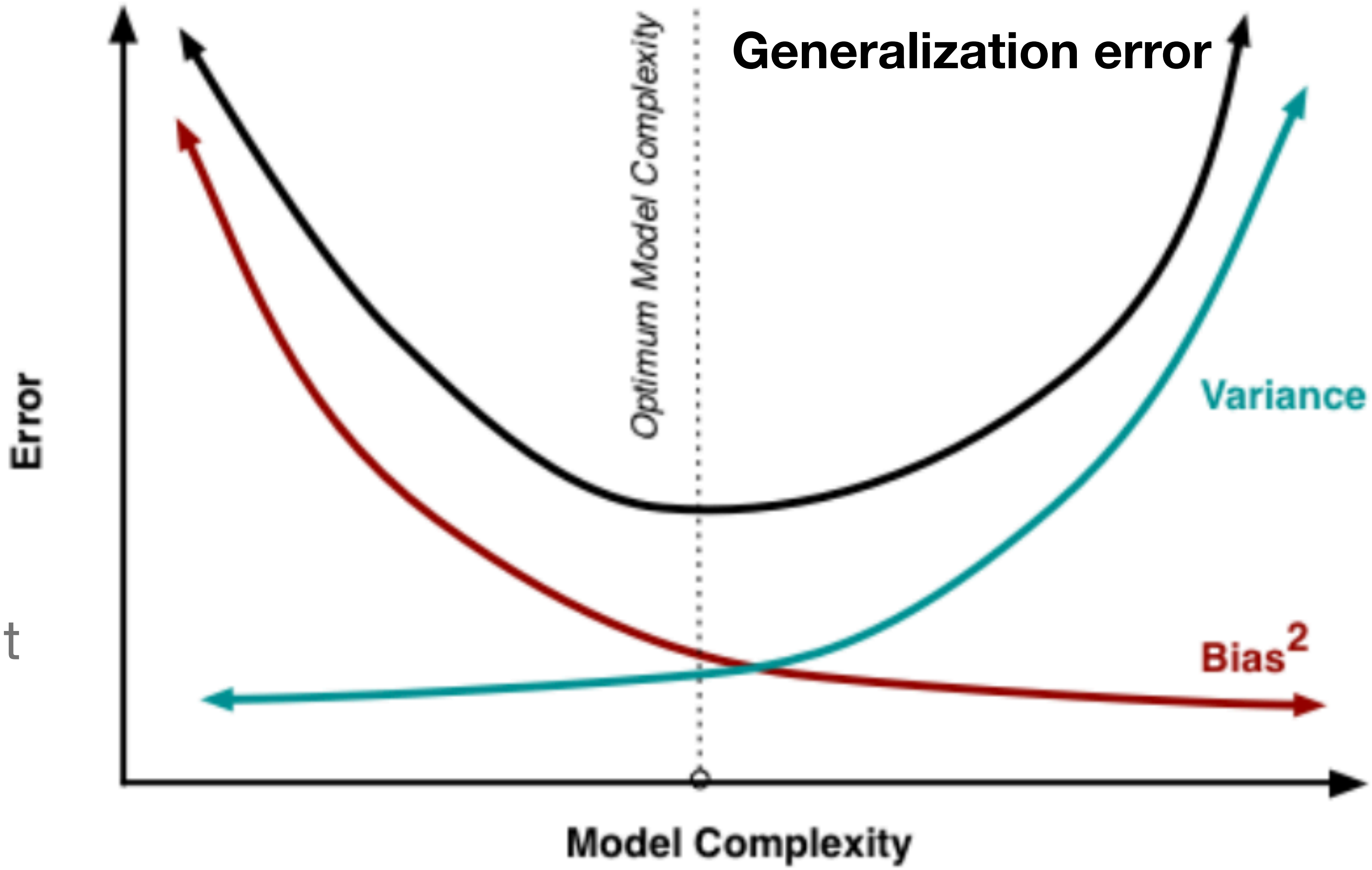
Evaluate the generalization error of your procedure by splitting data set in three samples:

1 training sample: fit values of model parameters

2. validation sample: check performance on independent data and optimize it by tuning # of parameters

3. Test sample final evaluation of performance, with all parameters fixed

[Kagan]



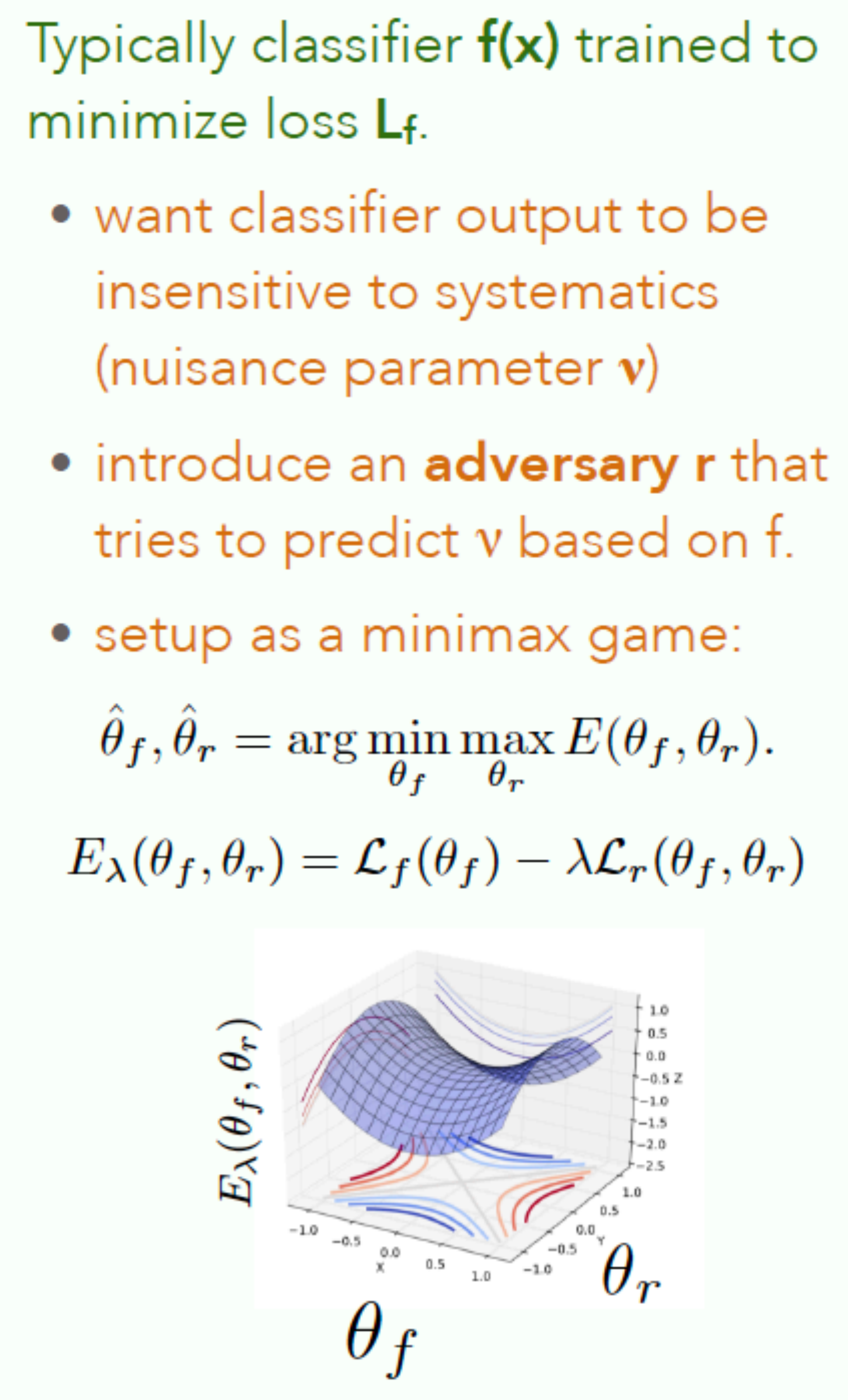
Regularization

One approach for avoiding overfitting/overtraining is to incorporate in the cost function a penalty for large weights.

$$\tilde{E}(\boldsymbol{w}) = E(\boldsymbol{w}) + \frac{\lambda}{2}\boldsymbol{w}^T\boldsymbol{w}$$

Increasing the regularization parameter λ drives the weights to zero, thus effectively reducing the number of nodes unless they are consistently supported by the training data

Advanced usages — adversarial networks



Typically classifier **f(x)** trained to minimize loss **L$_f$**.

- want classifier output to be insensitive to systematics (nuisance parameter **ν**)
- introduce an **adversary r** that tries to predict ν based on f.
- setup as a minimax game:

$$\hat{\theta}_f, \hat{\theta}_r = \arg\min_{\theta_f}\max_{\theta_r} E(\theta_f, \theta_r).$$

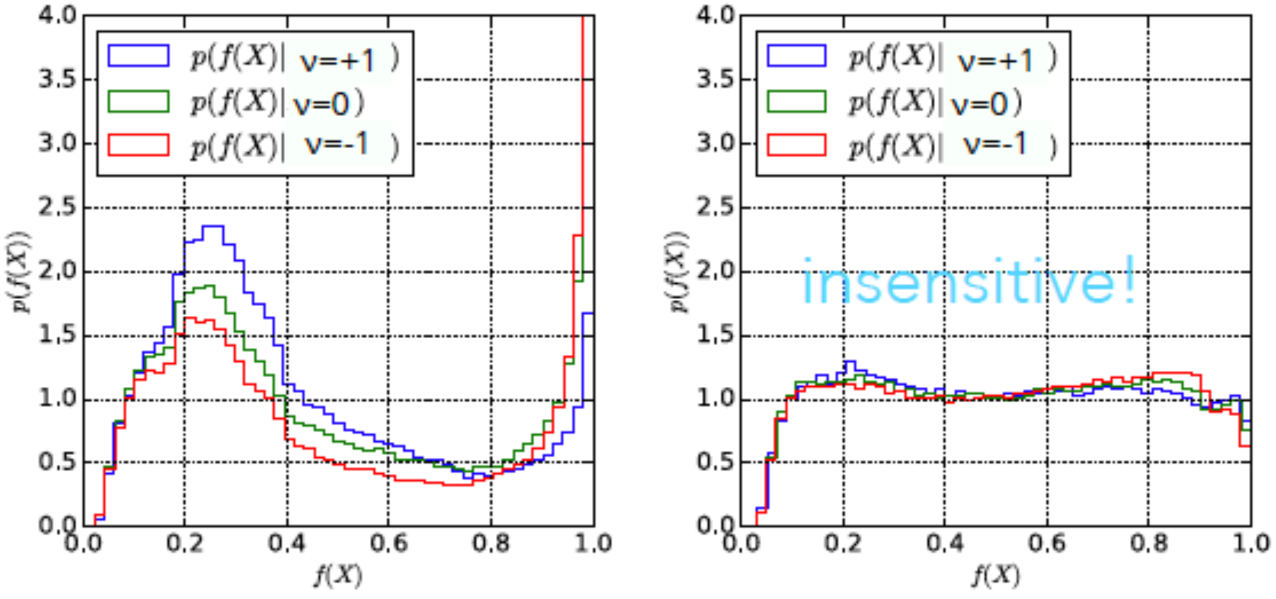
$$E_\lambda(\theta_f, \theta_r) = \mathcal{L}_f(\theta_f) - \lambda \mathcal{L}_r(\theta_f, \theta_r)$$

$E_\lambda(\theta_f, \theta_r)$
θ_f
θ_r
X
Y
Z
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
−2.5

Make the classifier output less sensitve to systematic effects or less biasing toward variables that may need to be fit downstream in the analysis.

Similar techniques been applied to decision trees (more later)

Discriminating function for various values of ν



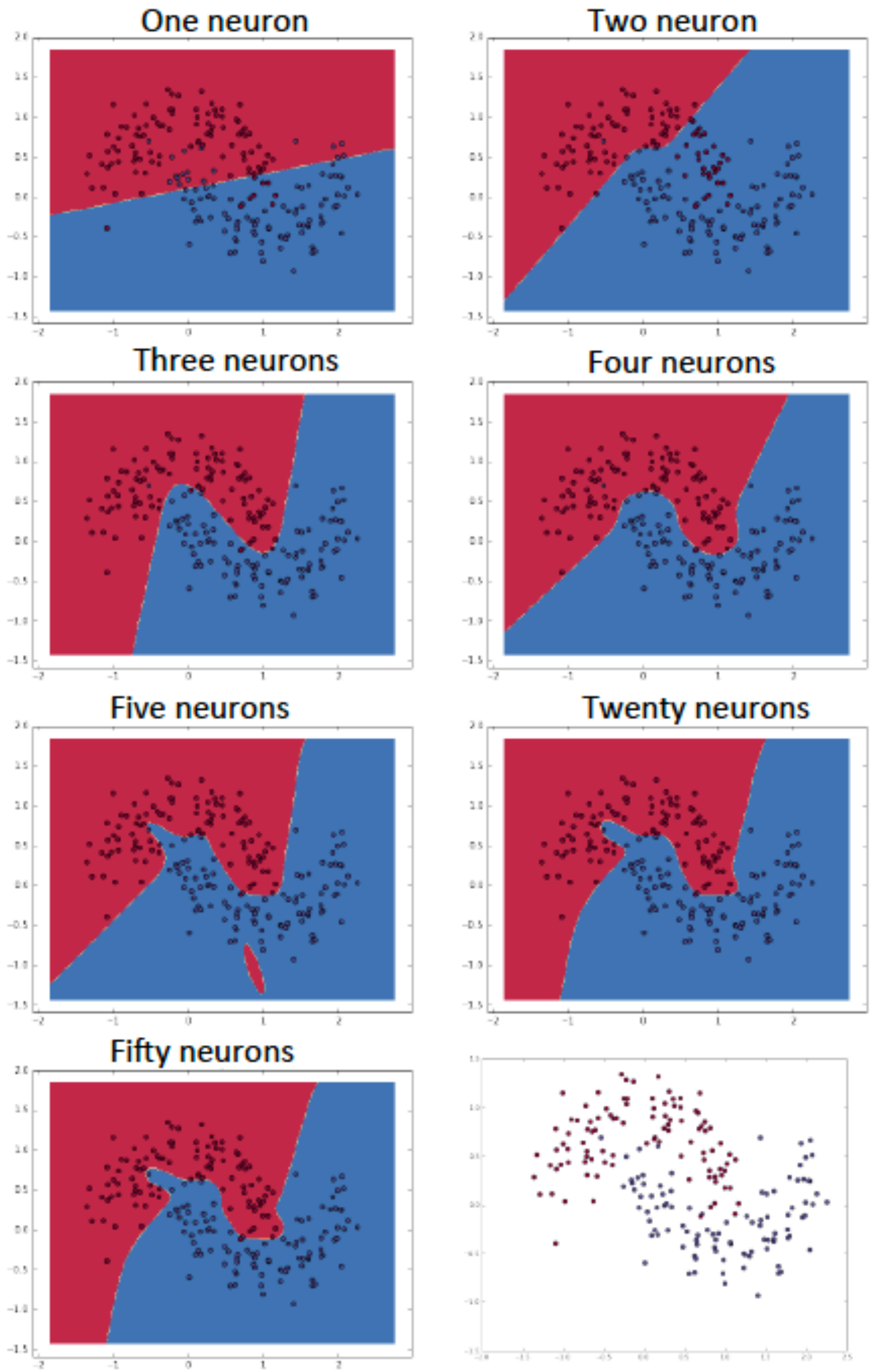
G. Louppe, M. Kagan, K. Cranmer, arXiv:1611.01046

Universal approximation theorem

A feed-forward multilayer perceptron with a single hidden layer and a finite number of nodes activated through any continuous nonpolynomial function can approximate arbitrarily well any continuous function

(Says nothing on how many nodes we need or how many data…)

Universal approximation theorem

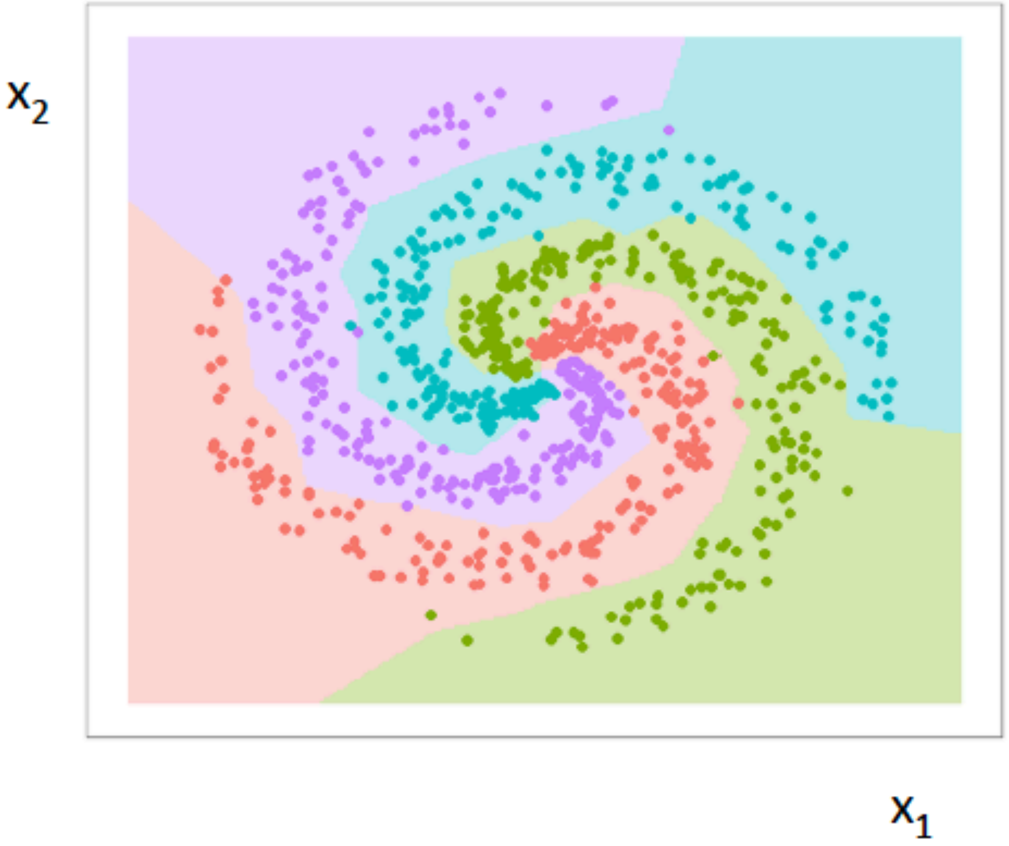


http://www.wildml.com/2015/09/implementing-a-neural-network-from-scratch/

2-class classification
1-hidden layer NN
L2 norm regularization

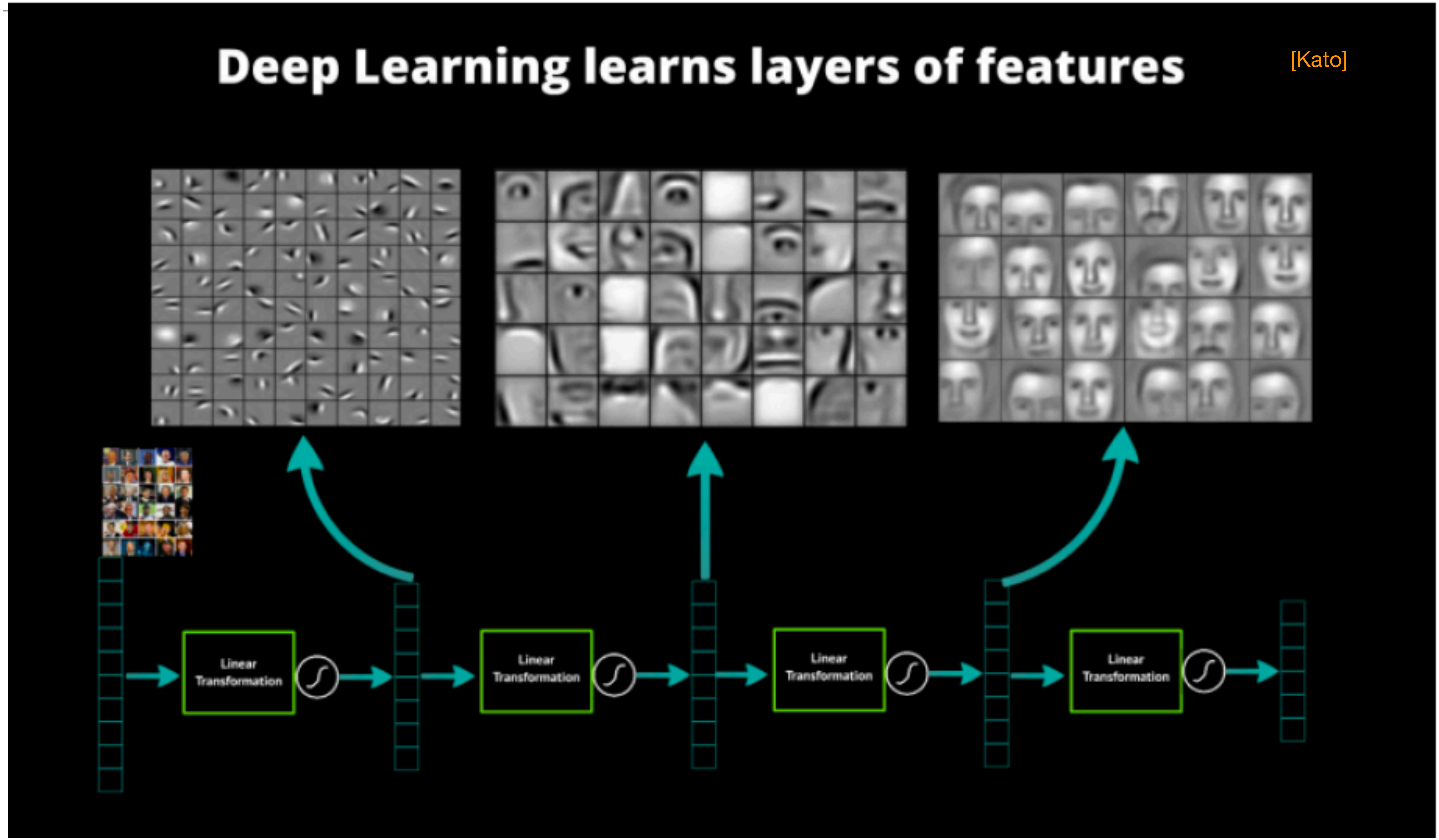
4-class classification
2-hidden layer NN
ReLU activations
L2 norm regularization

[Kagan]



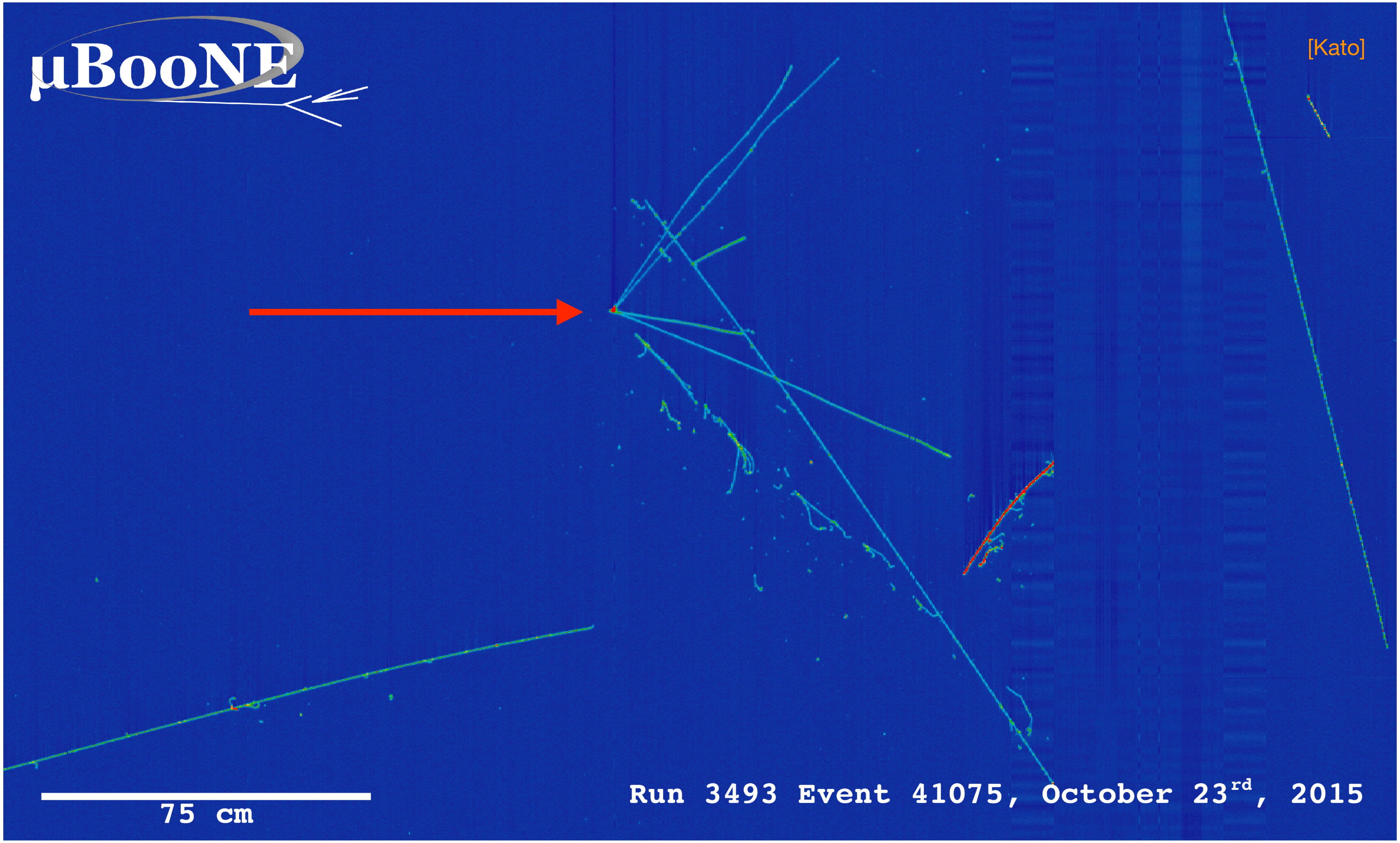
http://junma5.weebly.com/data-blog/build-your-own-neural-network-classifier-in-r

Advanced usages — convolutional networks



Promising for the analysis of LAr time-projection chamber “event pictures” from current and future neutrino experiments

Advanced usages — convolutional networks



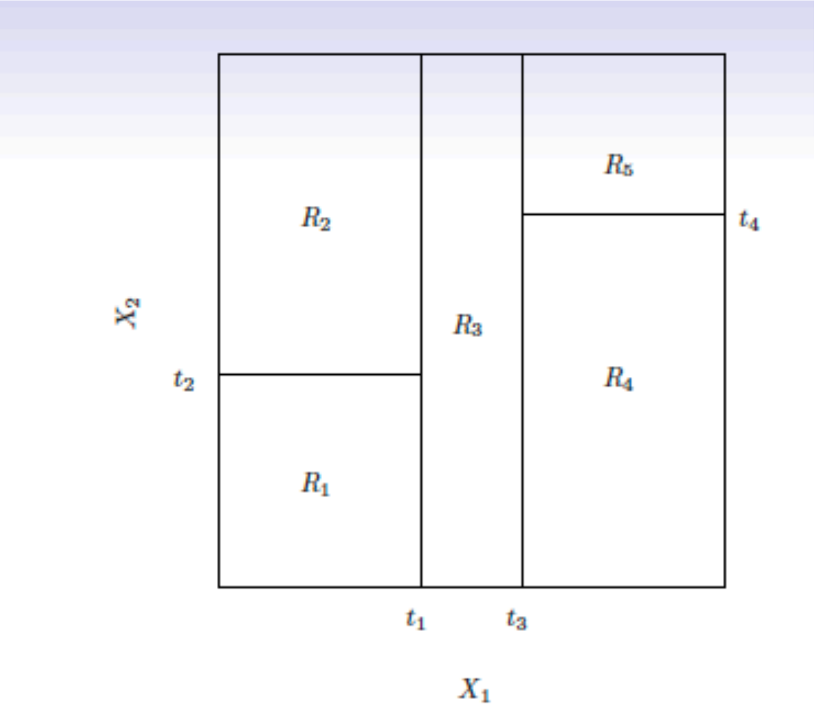
Promising for the analysis of LAr time-projection chamber "event pictures" produced in current and future neutrino experiments

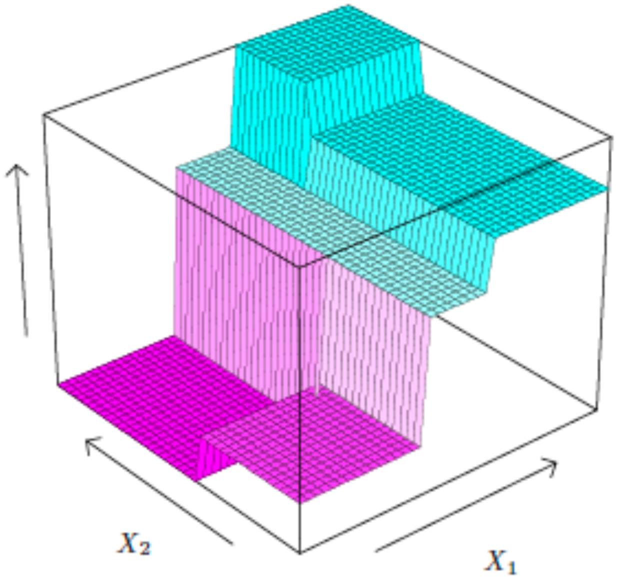
Decision trees

Algorithms that subdivide the space of the input variables (features) into a number of simple nonoverlapping regions (n-dimensional rectangles).

Each region will be labeled as “signal region” or “background region” according to the predominant category of training events that populate it after training.

Such labels are used to classify the test events.



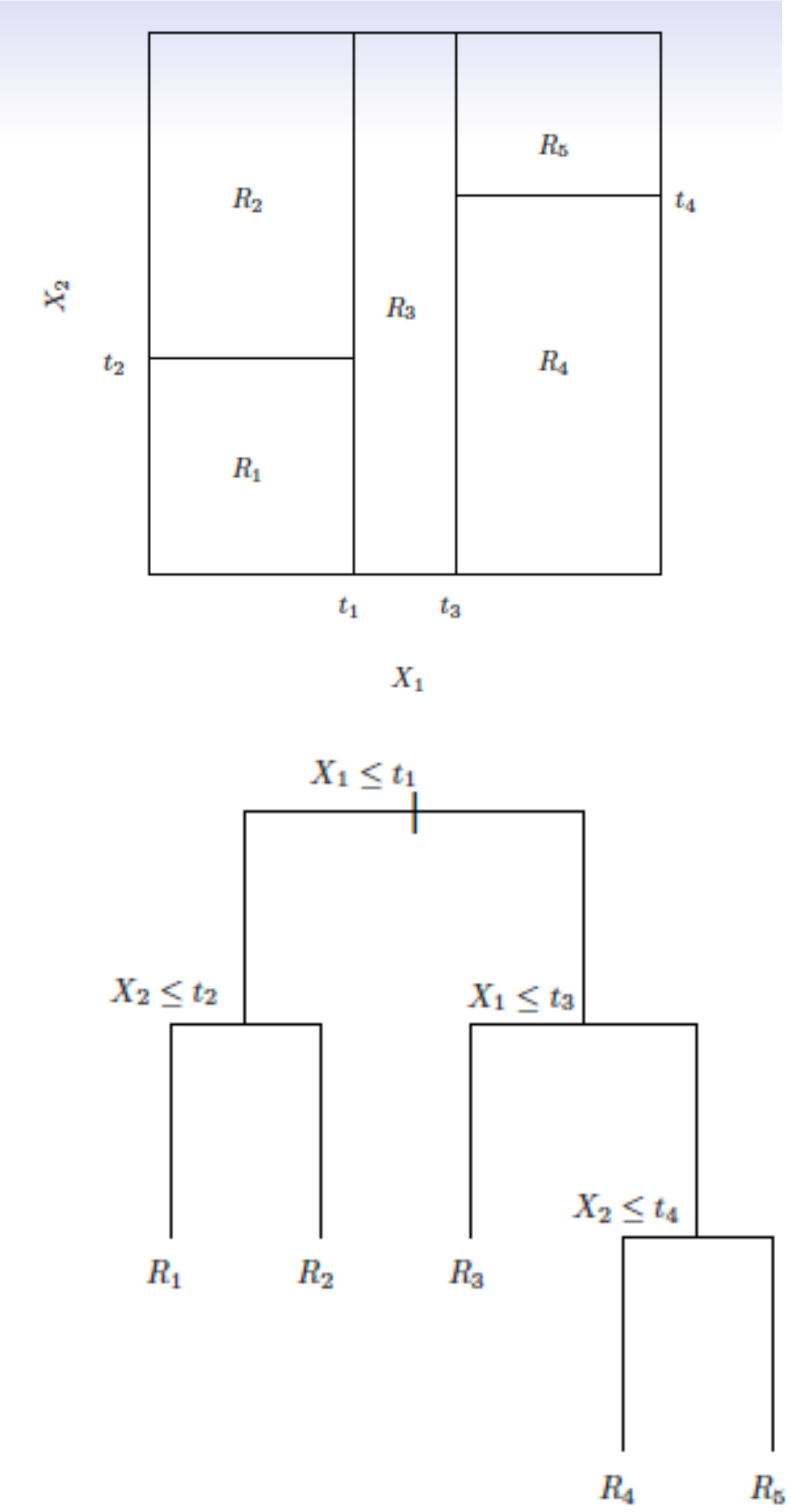


Decision trees

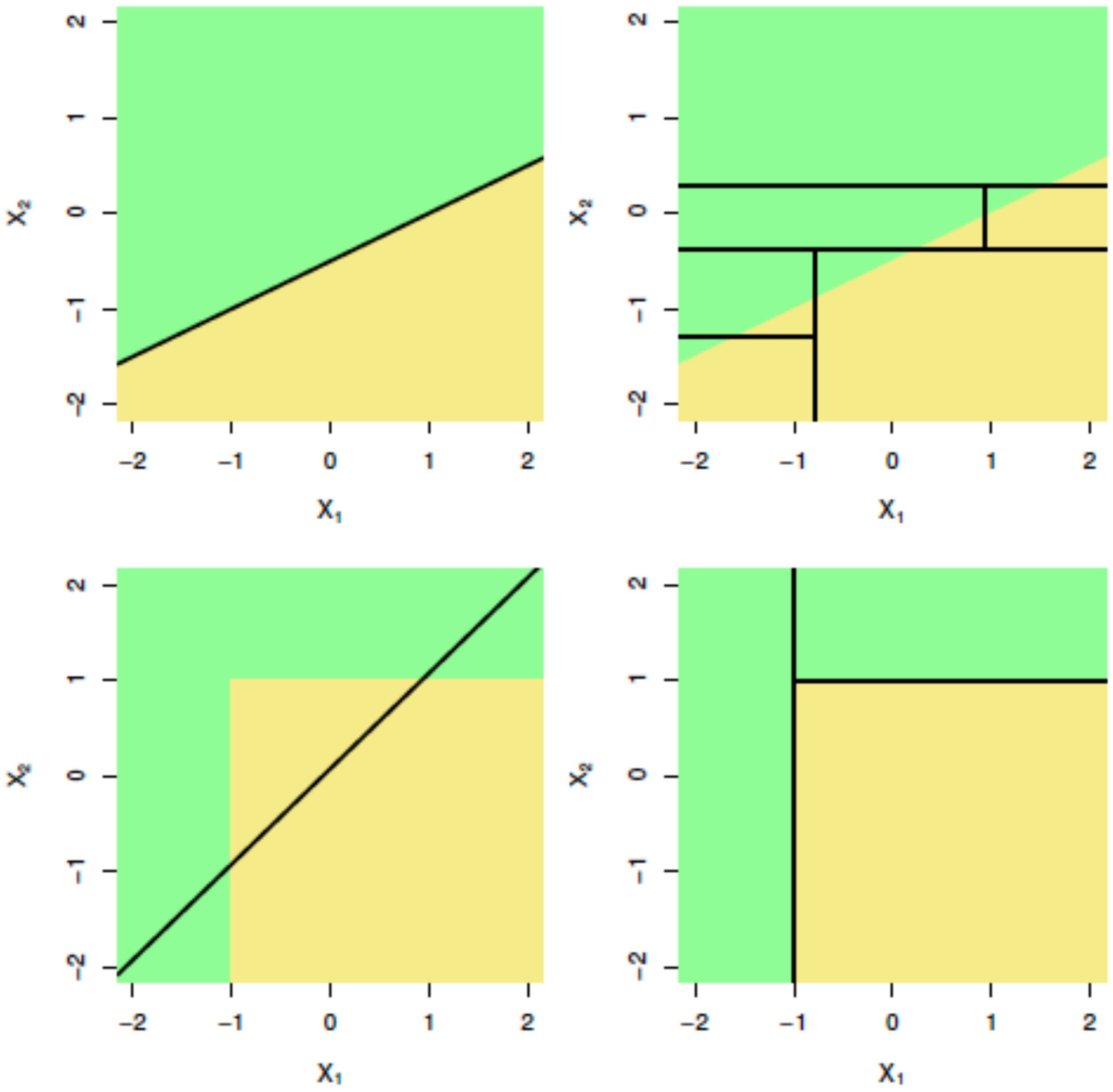
Subdivision of feature-space corresponds to a set of splitting rules.

Represent through an (inverted) tree-like structure made of a cascade of decision nodes, each associated to an input variable

Each node tests a single feature of the event at a time (e.g., cuts on a single variable) and routes the event on one of its downstream branches. partitioning the sample into subsamples of increasing purity until the final classification is reached (events accumulated in the terminal nodes (leafs).



Trees vs linear models



Building the tree —how the subregions are chosen

Recursive binary splitting.

At each building step, choose the input variable and the cut threshold on it that minimizes a cost function.

Misclassification rate, the fraction of events of the training sample in that region that do not belong to the most common class (not really used)

$$E = 1 - \max_k(\hat{p}_{mk}).$$

Here $\hat{p}_{mk}$ represents the proportion of training observations in the mth region that are from the kth class.

Typically use cross enthropy $$D = -\sum_{k=1}^{K} \hat{p}_{mk} \log \hat{p}_{mk}.$$ or functions of it (Gini)

Building the tree —how the subregions are chosen

Repeat the process of choosing the variable and cut value that minimizes the cost but do it restricting to one of the two regions previously identified.

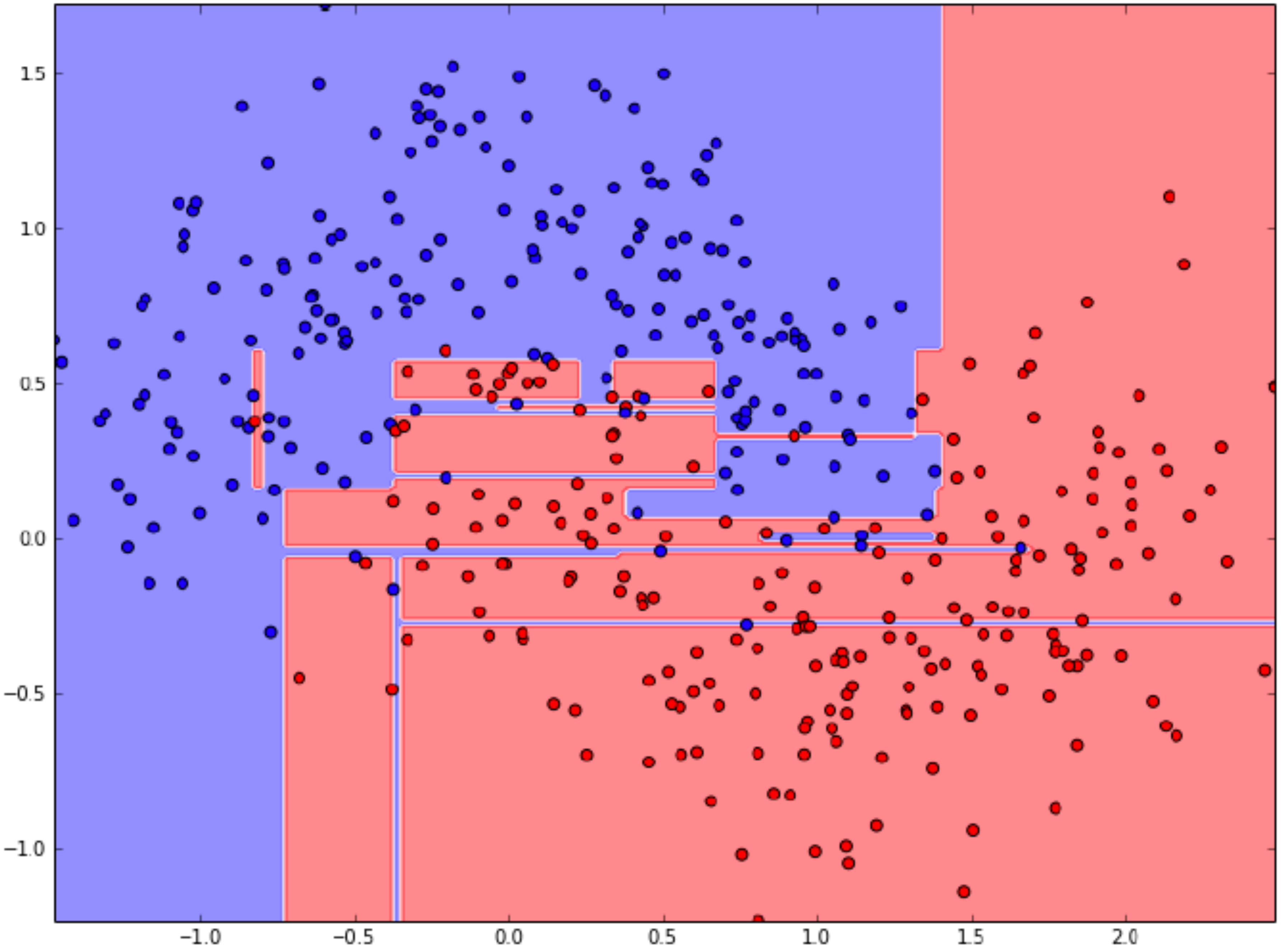
Without a stopping criterion, the training sample could end up being exactly classified, leading to strong overtraining. Typically impose stopping criteria like minimum number of entries in a node or achieved purity.

Still decision trees can grow very large and pruning is applied. For instance terminal leaves are recombined if their purity is compatible within statistical uncertainties.

Instabilities

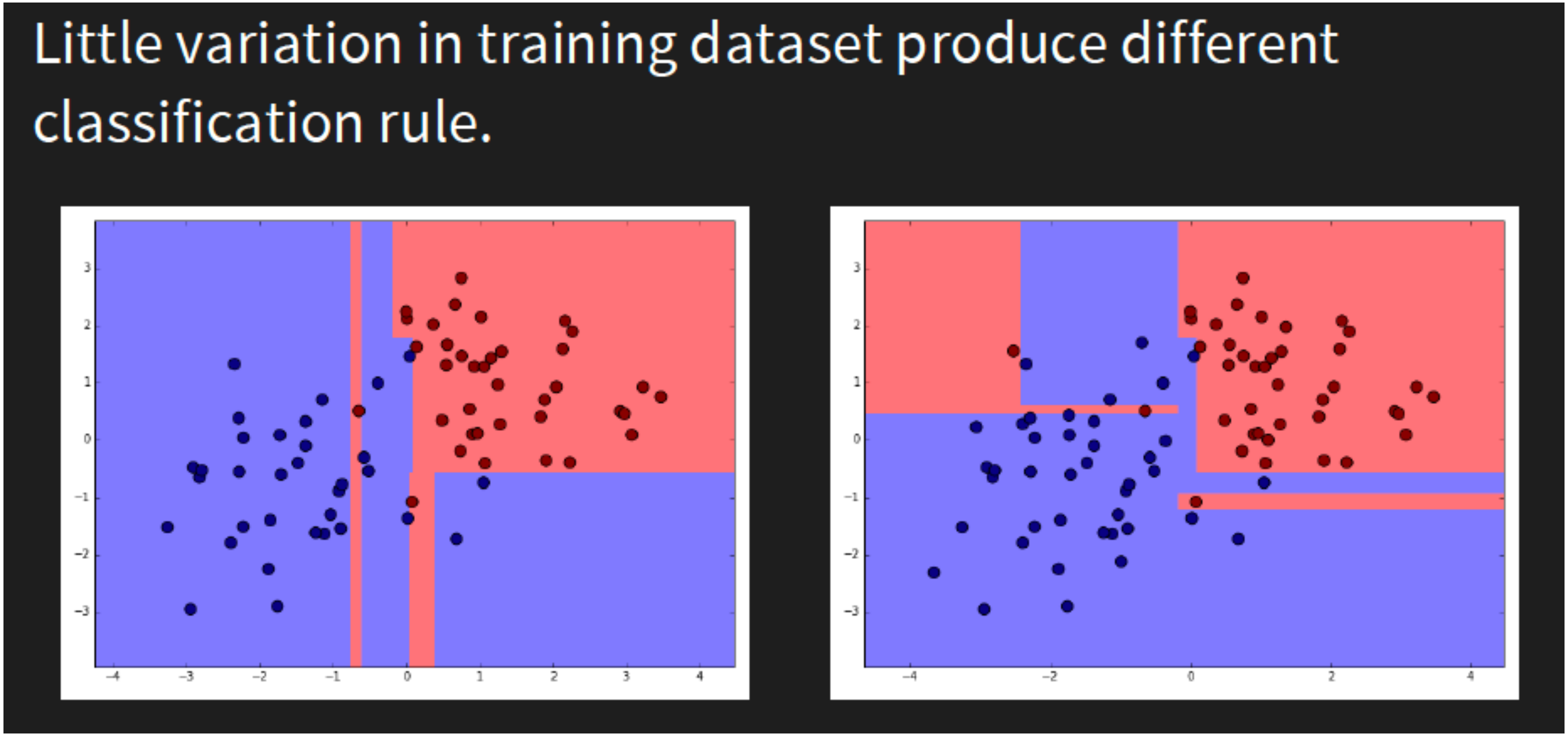
[Rogozhnikov]

Tree keeps splitting until each event is correctly classified



Instabilities

[Rogozhnikov]



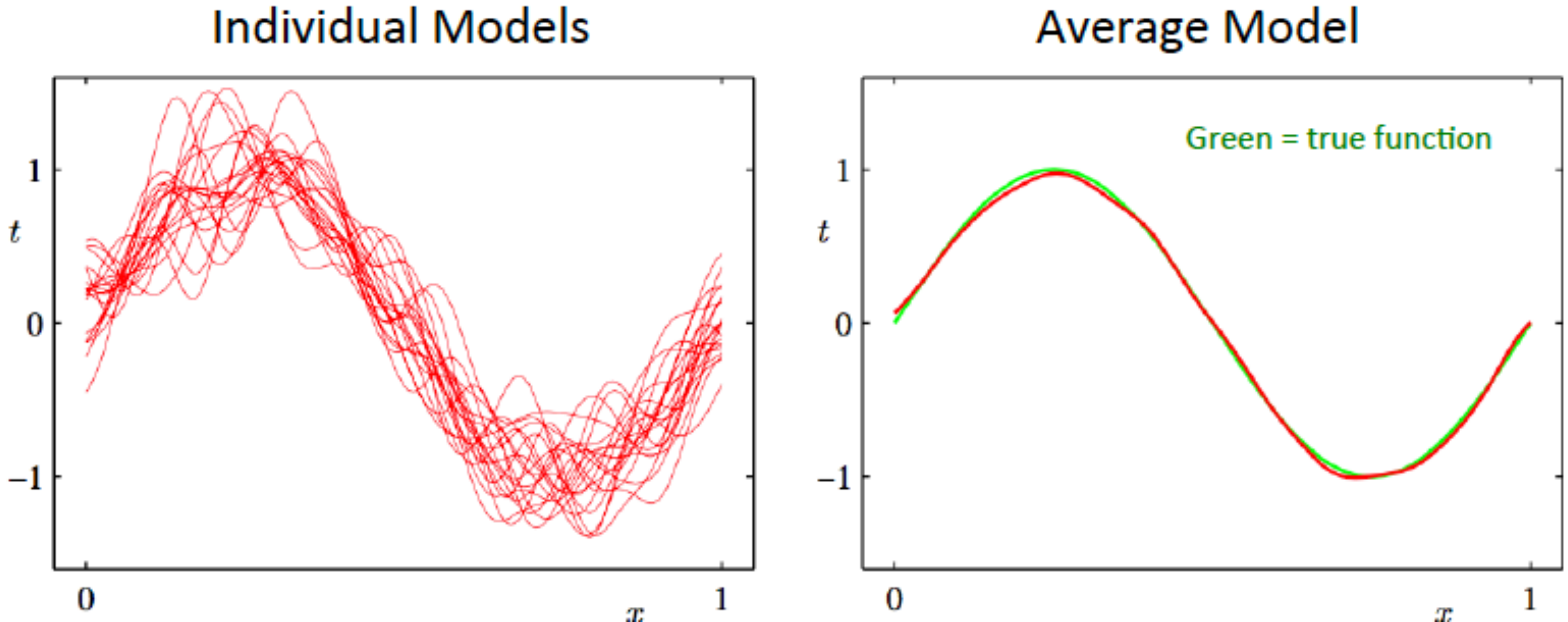
Little variation in training dataset produce different classification rule.

Improving trees

Typically classification performance of decision trees is not competitive with that from other machine-learning approaches.

By aggregating multiple decision trees, the classification performances is improved.

Combination of several weak learners with high variance goes a long way



[Bishop]

Improving trees - bagging/boosting

Generate N nearly independent training samples by resampling a single training sample. Build a tree based on each of the resampled training samples. For each test point in feature space, look at the classification of the N trees, and define the classification output of the tree as the most common output among the N trees (bagging)

Train N trees in sequence, giving in each more weight to the training examples misclassified in the previous tree. Take the weighted vote of the outputs to classify the examples. (boosting)

(not specific of decision trees, can be applied to other machine-learning methods).

Improving trees - random forests

Tweak the bagging algorithm decorrelates better the trees obtained by the resampled training data.

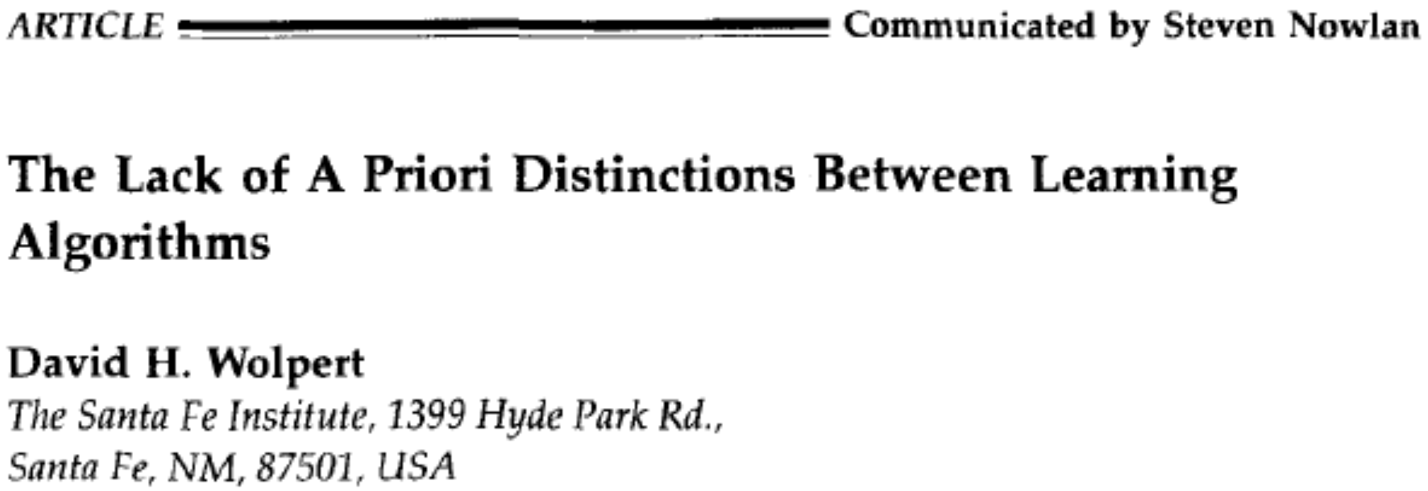
When considering a split in the building step, only a random subset of the whole set of features is available for choose the splitting-variable.
This “forces” the trees to develop differently thus reducing correlations among them and therefore the variance of the bagged trees

E.g, if one feature is much more discriminating than all others. A split based such feature would likely to be at the top of most the trees derived from the resampled training data. Such trees will therefore be similar and yield strongly cotrrelated outputs strongly. Since correlations do not reduce by averaging, the advantages of bagging will be lost.

Practical advice

[Kagan]

No free lunch theorem



ARTICLE **Communicated by Steven Nowlan**

The Lack of A Priori Distinctions Between Learning Algorithms

David H. Wolpert
The Santa Fe Institute, 1399 Hyde Park Rd.,
Santa Fe, NM, 87501, USA

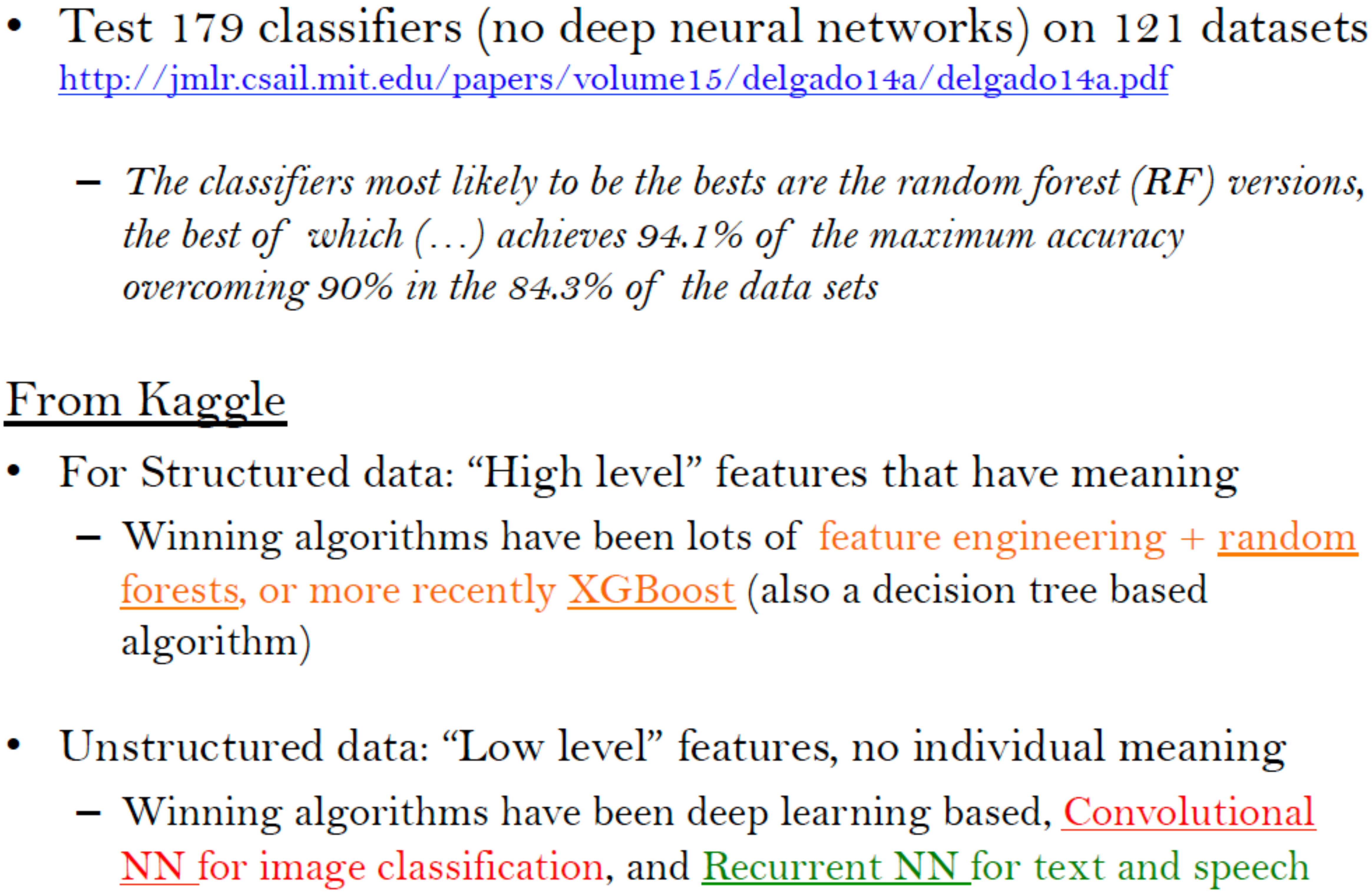
Many ML methods and tools out there to improve our reach in HEP-specific problems: linear, nearest neighbor, ANN, DeepNN, DT ensembles, support vector machines…

With no prior knowledge, general statements on performance are hard. Performance very much dependent on the details of the problem at hand.

The only shortcut to your trial and error is if your problem mirrors a similar problem somebody else has already explored.

Empirical heuristics

[Kagan]



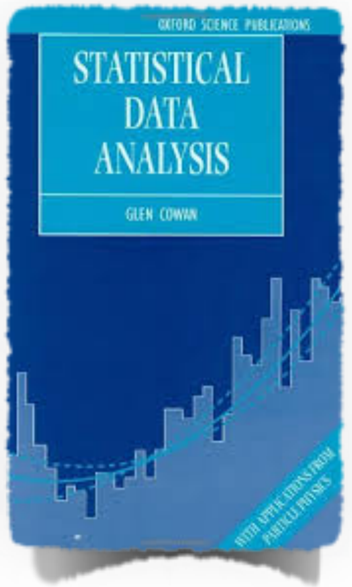
- Test 179 classifiers (no deep neural networks) on 121 datasets
  http://jmlr.csail.mit.edu/papers/volume15/delgado14a/delgado14a.pdf
  - *The classifiers most likely to be the bests are the random forest (RF) versions, the best of which (...) achieves 94.1% of the maximum accuracy overcoming 90% in the 84.3% of the data sets*

From Kaggle

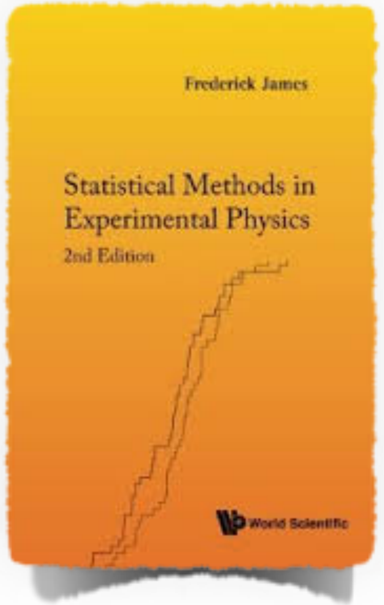
- For Structured data: "High level" features that have meaning
  - Winning algorithms have been lots of feature engineering + random forests, or more recently XGBoost (also a decision tree based algorithm)
- Unstructured data: "Low level" features, no individual meaning
  - Winning algorithms have been deep learning based, Convolutional NN for image classification, and Recurrent NN for text and speech

Thanks for your attention and your questions

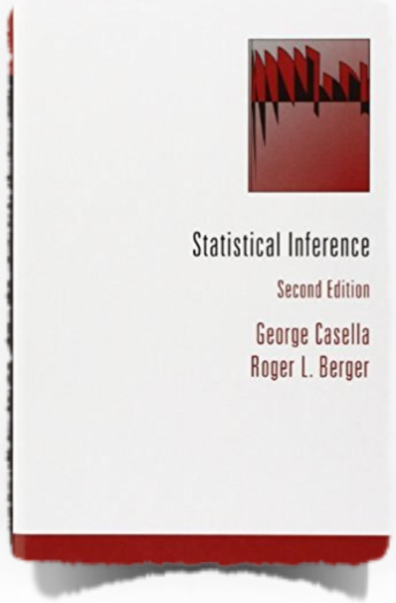
Further readings - books


OXFORD SCIENCE PUBLICATIONS
STATISTICAL DATA ANALYSIS
GLEN COWAN
WITH APPLICATIONS FROM PARTICLE PHYSICS

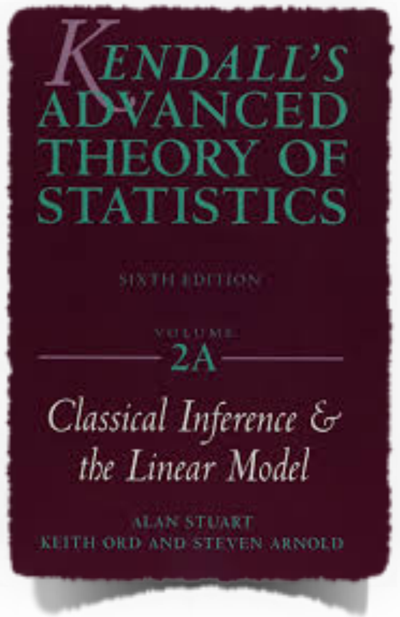
G. Cowan,"Statistical data analysis"



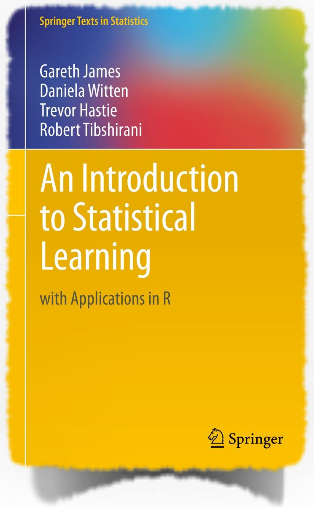
F. James, "Statistical Methods in Experimental Physics, data analysis"



G. Casella, R. Berger, "Statistical Inference

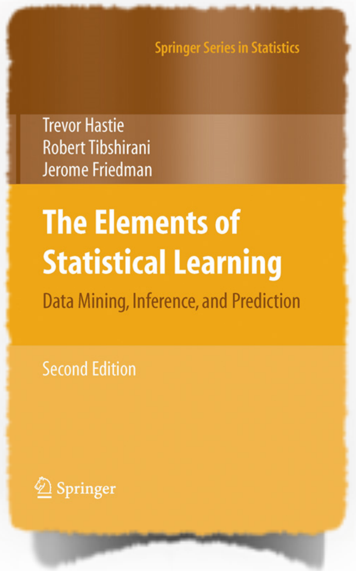

KENDALL'S ADVANCED THEORY OF STATISTICS
SIXTH EDITION
VOLUME 2A
Classical Inference & the Linear Model
ALAN STUART
KEITH ORD AND STEVEN ARNOLD

A. Stuart, et al "Kendall's Advanced Theory of Statistics Vol 2A"



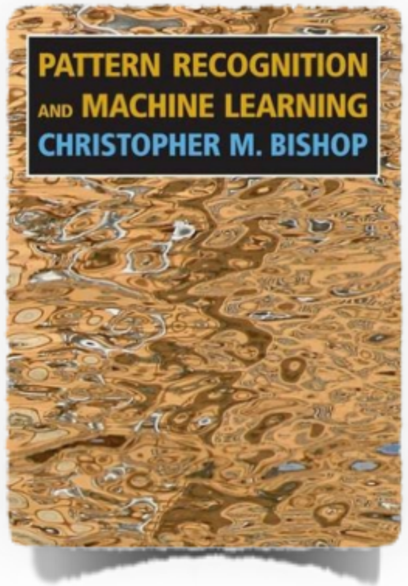
T. Hastie et al., "An Introduction to Statistical learning"

PDF and slides: www-bcf.usc.edu/~gareth/ISL/


Springer Series in Statistics
Trevor Hastie
Robert Tibshirani
Jerome Friedman
The Elements of Statistical Learning
Data Mining, Inference, and Prediction
Second Edition
Springer

T. Hastie et al., "The elements of statistical learning"

https://web.stanford.edu/~hastie/ElemStatLearn/



C. Bishop "Pattern recognition and machine learning"

Further readings — slides/docs

- Statistics@ http://hcpss.web.cern.ch/hcpss/ (Excellent lectures by K. Cranmer, G. Cowan, B. Cousins et al.)
- Lectures from Glen Cowan's page https://www.pp.rhul.ac.uk/~cowan/
- Terascale Stat School (especially 2015 F. James' lectures) https://indico.desy.de/conferenceDisplay.py?confId=11244
- T. Junk's lectures from www-cdf.fnal.gov/~trj/
- L. Lyons lectures: https://indico.cern.ch/event/431038/
- Notes from CDF's Statistics Committee public page https://www-cdf.fnal.gov/physics/statistics/
- B. Cousins' stuff: try to find his (CMS restricted) "Statistics in Theory - prelude to Statistics in Practice" lectures. Look at his statistics papers on inspire and the references he reccommends.
- Proceedings/docs from the PHYSTAT conferences and workshops, linked from phystat.org
- IML material (https://indico.cern.ch/category/8009/ and recent HEP-relevant resources linked from https://github.com/iml-wg/HEP-ML-Resources#lectures.