

Advanced event analysis methods

Reinhard Schwienhorst



CPPM

CPPM D0 Seminar, June 20 2008

Outline

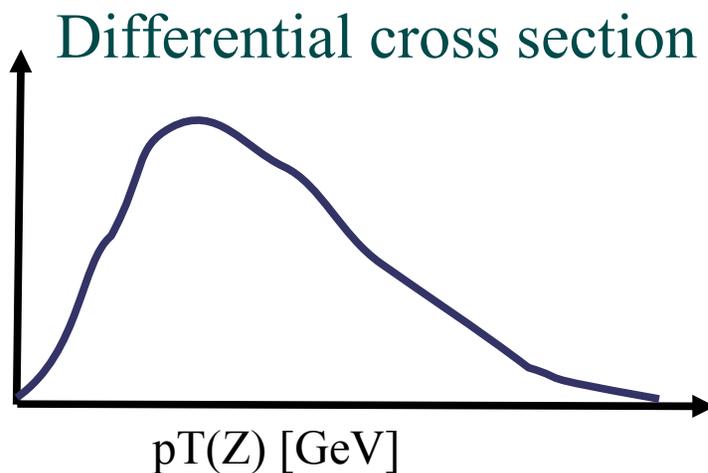
- Introduction
- Decision Tree
- Boosting
- Random Forest
- MiniBoone
- New developments
- Conclusions

Introduction

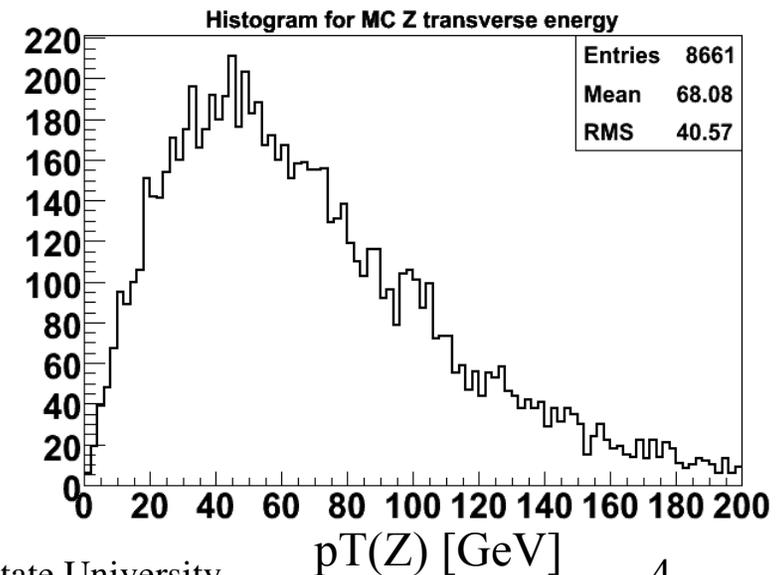
- I covered multivariate analysis techniques in my seminar talk here last year
- Today: Review and updates
 - Decision Trees,
 - Boosting,
 - Their application
 - Recent developments
- Examples mainly from D0

Monte Carlo Modeling

- In particle physics, we can calculate cross sections and differential cross sections for many processes
 - Remember your field theory 101 course?
- We approximate differential cross sections through Monte Carlo techniques
 - Rather than calculating full multi-dimensional differential cross section, we generate events → detector simulation
 - Example: $pp \rightarrow HZ$



MC simulation

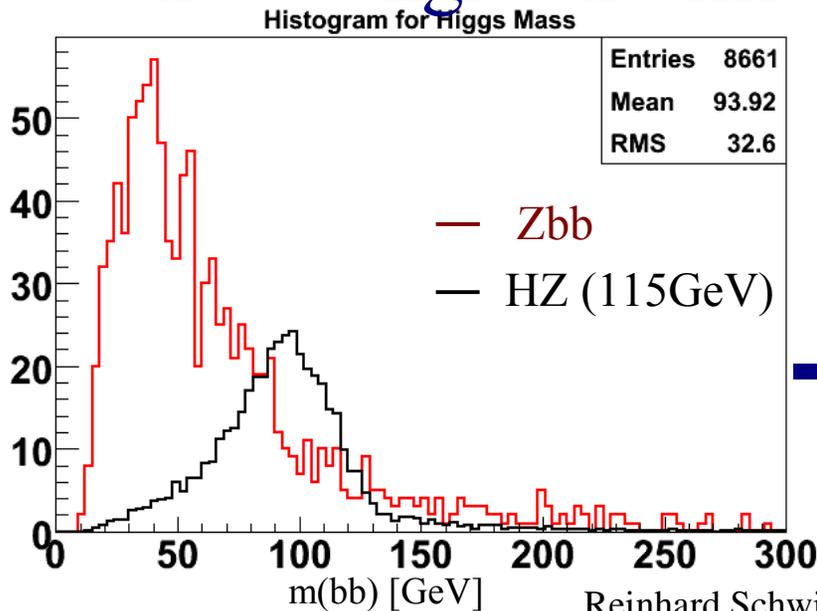


Bayesian limit

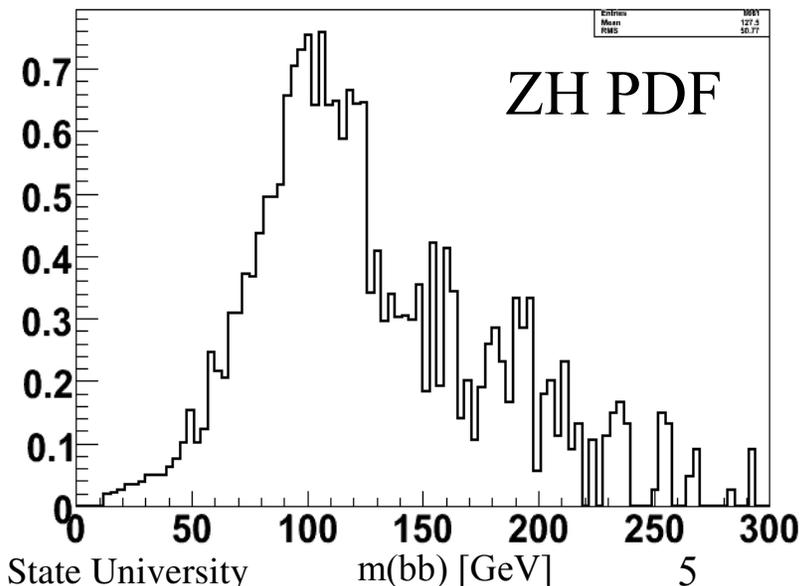
- For each analysis, there exists a fully optimized signal-background separation
 - Target function, also called Bayes discriminant or Bayesian limit

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

- For a single discriminating variable, this ratio of signal and background likelihoods is easy to calculate



S / (S+B)



Bayesian limit

- For each analysis, there exists a fully optimized signal-background separation
 - Target function, also called Bayes discriminant or Bayesian limit

$$B(\mathbf{x}) = \frac{L(S|\mathbf{x})}{L(S+B|\mathbf{x})}$$

- For a single discriminating variable, this ratio of signal and background likelihoods is easy to calculate
- Limited by MC statistics if 2d histogram
 - Example:
 - 100 bins in the histogram for each variable
 - 10000 bins in 2d
 - To get ~10% MC modeling uncertainty in each bin, need 100 events per bin
 - Require 1000000 MC events total
 - In D0, this takes a while to generate...

Bayesian limit

- For each analysis, there exists a fully optimized signal-background separation
 - Target function, also called Bayes discriminant or Bayesian limit

$$B(\mathbf{x}) = \frac{L(S|\mathbf{x})}{L(S+B|\mathbf{x})}$$

- For a single discriminating variable, this ratio of signal and background likelihoods is easy to calculate
- In case of more than two variables, this isn't possible anymore
 - 3 variables would require 10^8 MC events
 - 20 variables require 10^{42} MC events
 - More events than generated in total in the history of computers

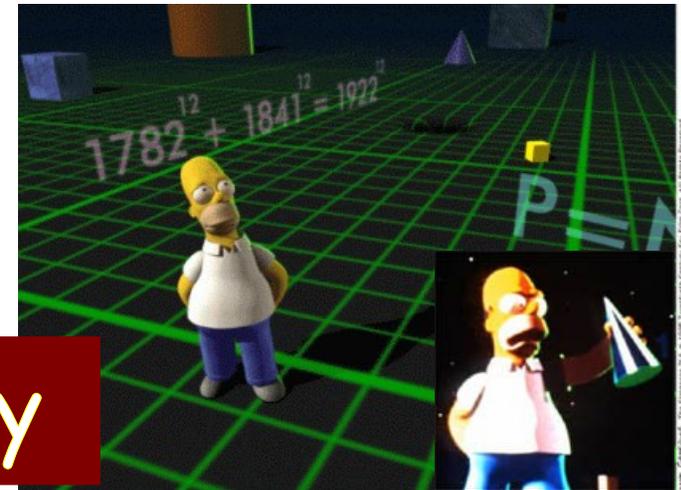
Bayesian limit

- For each analysis, there exists a fully optimized signal-background separation
 - Target function, also called Bayes discriminant or Bayesian limit

$$B(\mathbf{x}) = \frac{L(S|\mathbf{x})}{L(S+B|\mathbf{x})}$$

- For a single discriminating variable, this ratio of signal and background likelihoods is easy to calculate
- In case of more than two variables, this isn't possible anymore
 - Not enough MC statistic to compute a many-dimensional likelihood

Curse of dimensionality



The solution: Multivariate techniques

- Approximate the multi-dimensional likelihood
- Good methods do well even with finite MC statistics
 - Typical MC sample sizes are 10^6 events

Optimized event analysis

Optimized = {
Optimize signal-background separation
Exploit full event information
Event kinematics, angular correlations, ...
Take all correlations into account

Goal: Reach the Bayesian limit

- Requires detailed understanding of signal and background
 - Only applicable to searches for a specific signal or measurements of a specific process

Optimized event analysis

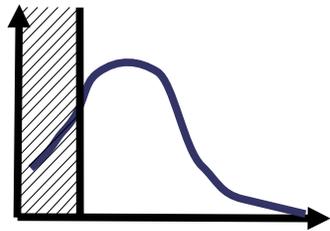
Optimized = {
Optimize signal-background separation
Exploit full event information
Event kinematics, angular correlations, ...
Take all correlations into account

Goal: Reach the Bayesian limit

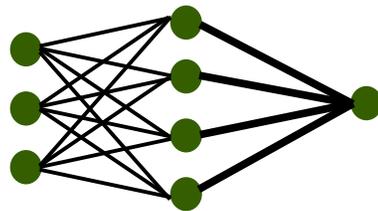
- Requires detailed understanding of signal and background
 - Only applicable to searches for a specific signal or measurements of a specific process
- Limited by background and signal modeling
 - MC statistics, MC model, background composition, shape, ...

Event analysis techniques

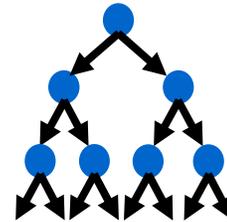
Cut-Based



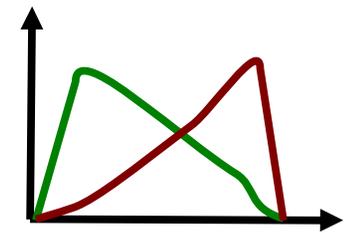
Neural networks



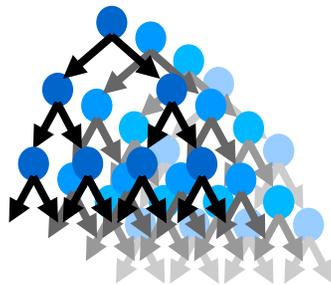
Decision trees



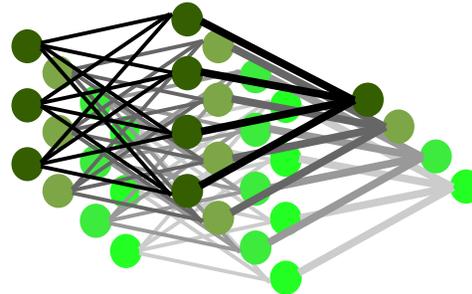
Likelihood



Boosted decision trees,
random forest



Bayesian neural networks



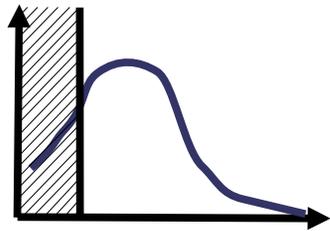
Matrix Elements

$$d^n \sigma_{hs} = \frac{(2\pi)^4 |\mathcal{M}|^2}{4 \sqrt{(q_1 q_2)^2 - m_2^2}} \times d\Phi_n$$

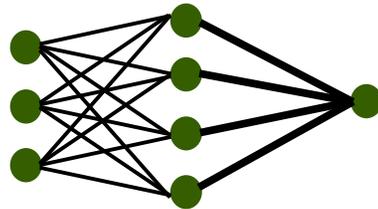
Many others: Kernel methods, support vector machines, ...

Event analysis techniques

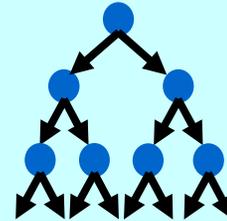
Cut-Based



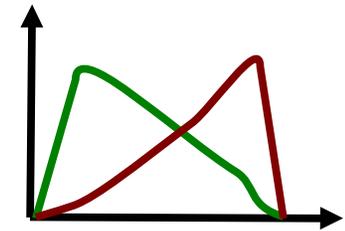
Neural networks



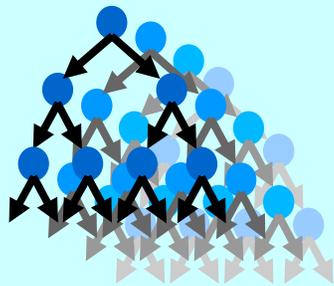
Decision trees



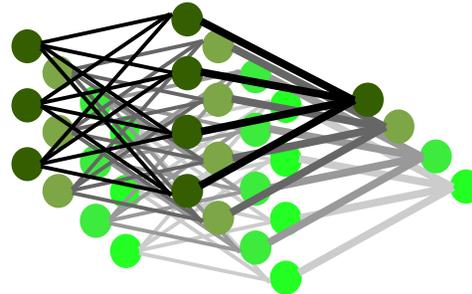
Likelihood



Boosted decision trees, random forest



Bayesian neural networks

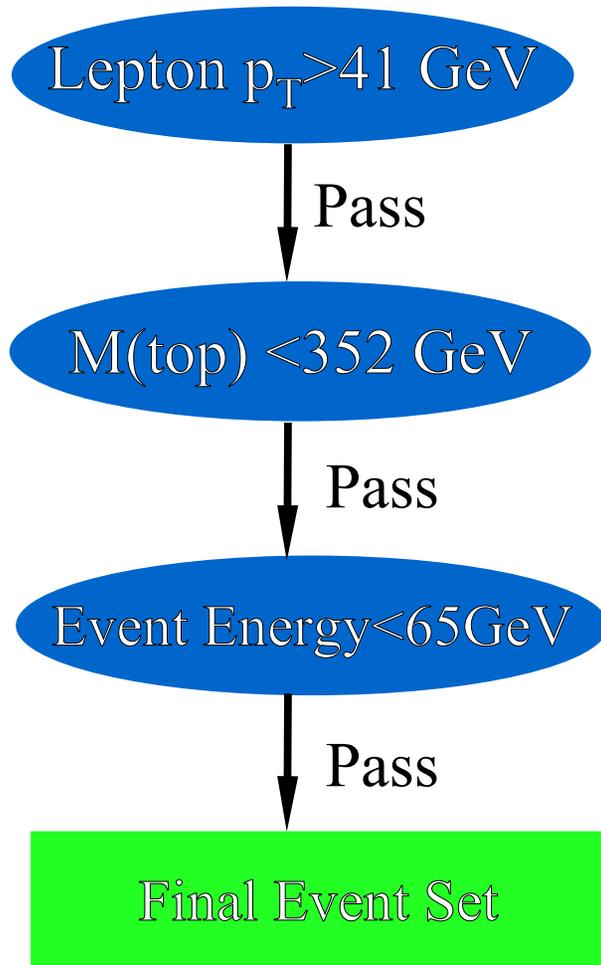


Matrix Elements

$$d^n \sigma_{hs} = \frac{(2\pi)^4 |\mathcal{M}|^2}{4 \sqrt{(q_1 q_2)^2 - m_2^2}} \times d\Phi_n$$

Many others: Kernel methods, support vector machines, ...

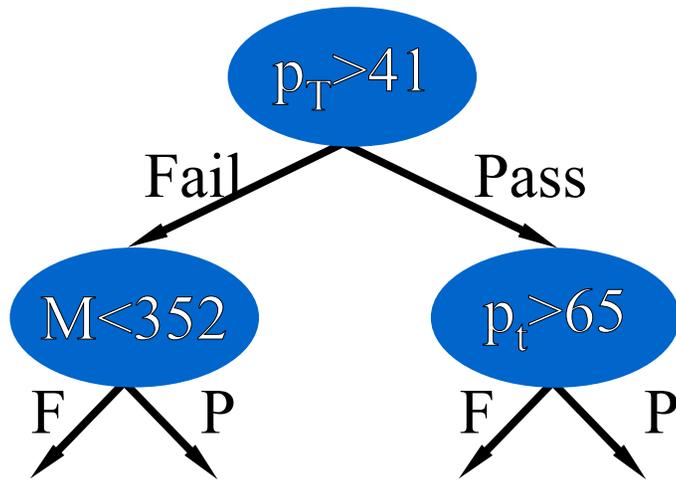
Cut-based analysis



In the final event set

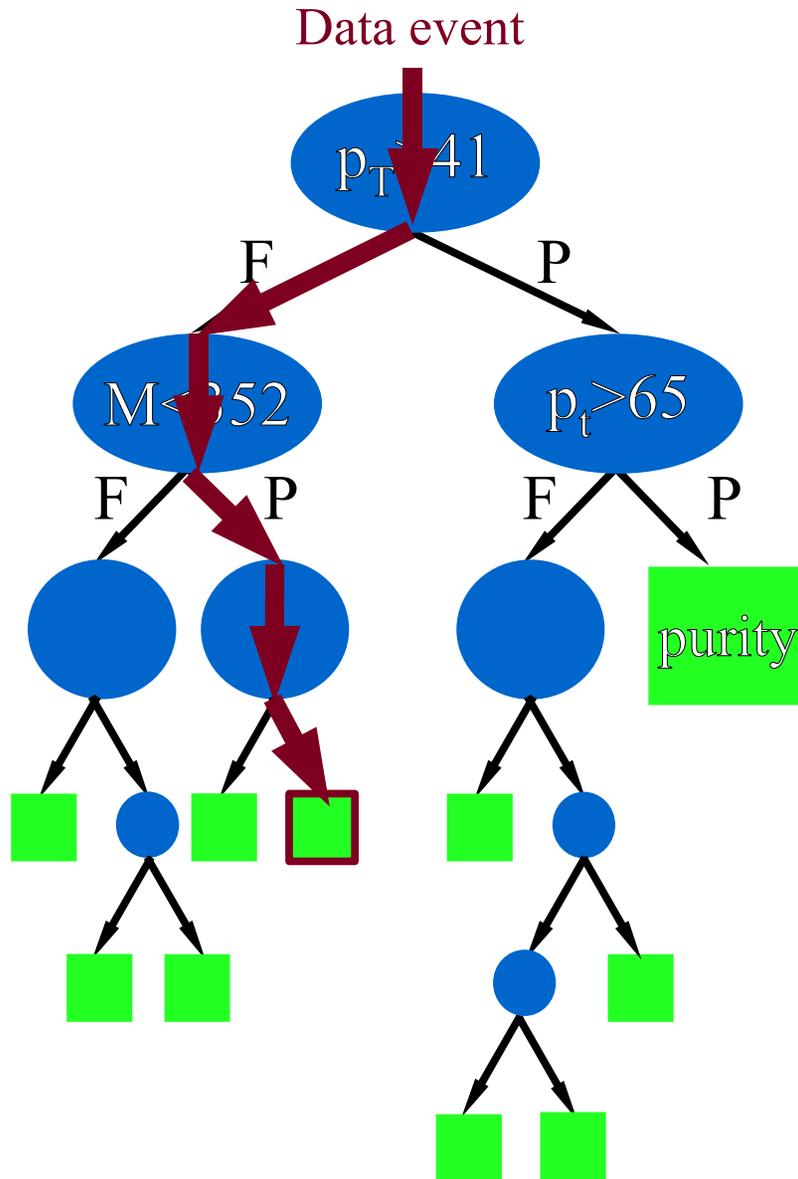
- Estimate background yield
- Compare to data
$$N_{\text{obs}} = N_{\text{data}} - N_{\text{B}}$$
- Calculate signal acceptance
$$\sigma = N_{\text{obs}} / (A * L)$$

Including events that fail a cut



- Create a tree of cuts
- Divide sample into “pass” and “fail” sets
- Each node  corresponds to a cut (branch)

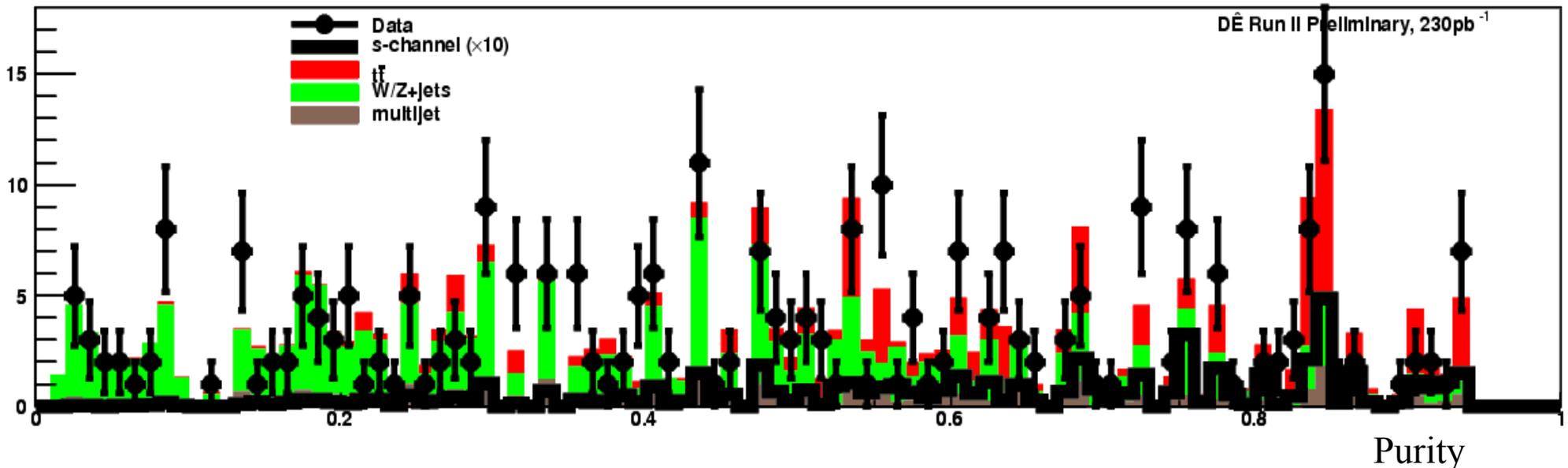
After training is finished: send data through the tree



- Send one data event through the tree at a time
- Each event ends up in a specific leaf 
- Assign the purity value that was calculated during training to this data event
 - Each leaf has a unique purity
 - Leafs will either be high purity (lots of signal) or low purity (lots of background) or in between
- Also send separate samples of signal and background MC through the tree
 - Background estimate

Decision tree output

- Train on signal and background models (MC)
 - Stop and create leaf when $N_{MC} < 100$
- Compute purity value for each leaf
- Send data events through tree
 - Assign purity value corresponding to the leaf to the event
- Result approximates a probability density distribution

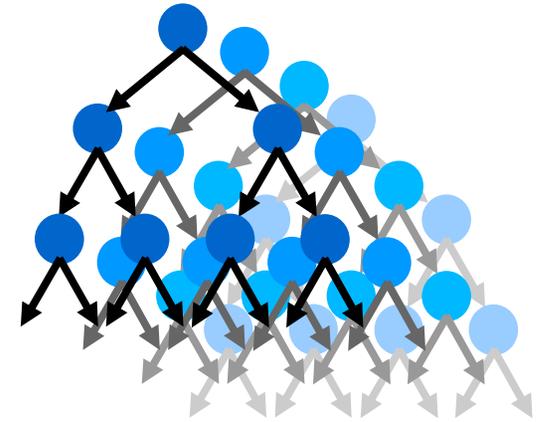


Boosting



Boosting

- A general method to improve the performance of any weak qualifier
 - Decision trees, neural networks, ...

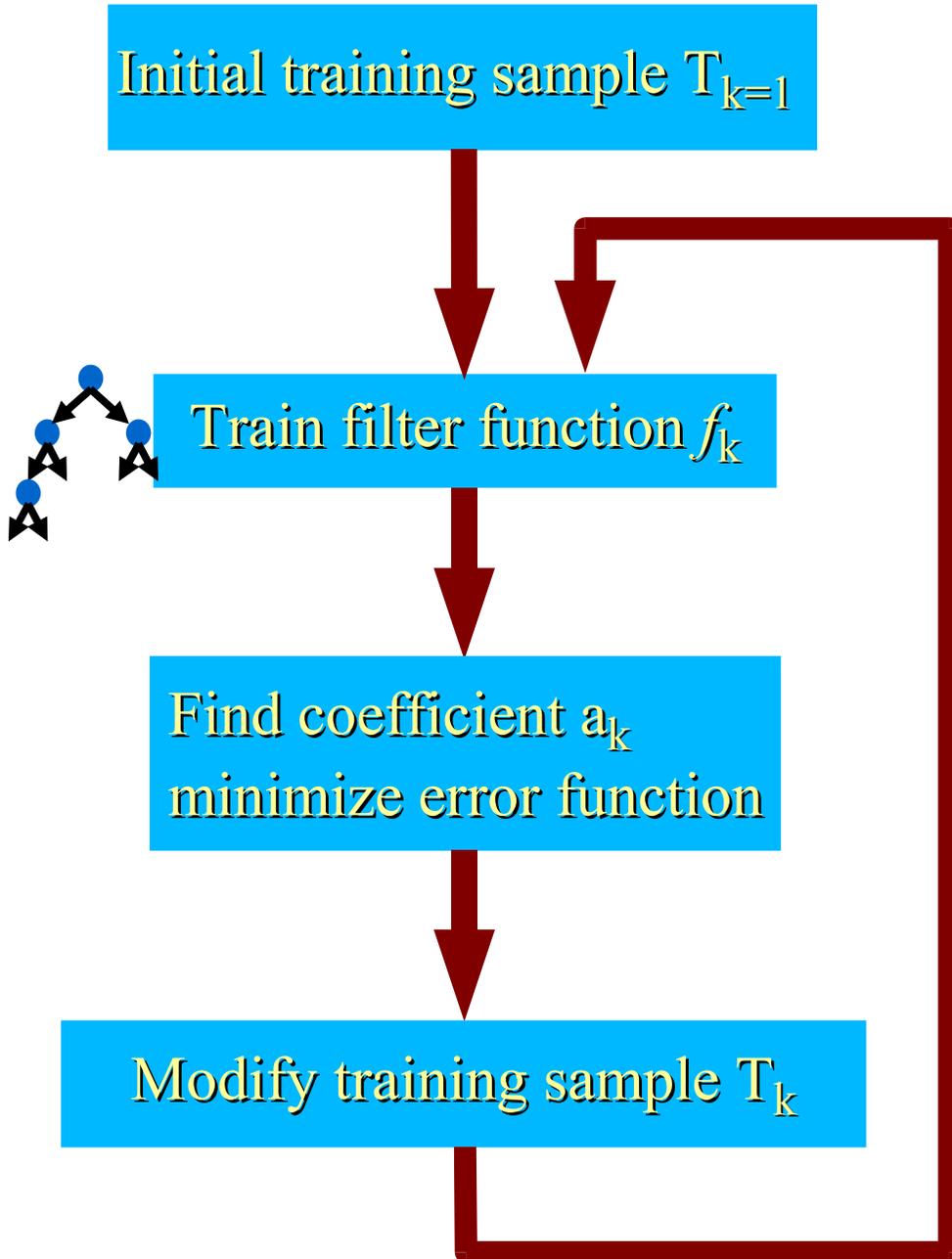


- Linear combination of many filter functions

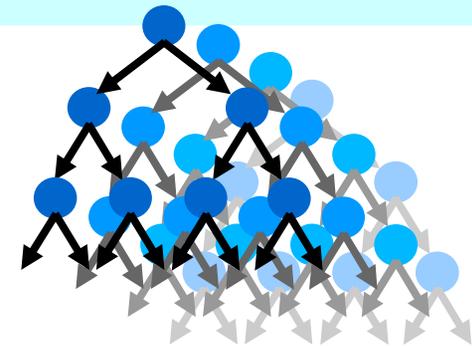
$$F(\mathbf{x}) = \sum_k a_k f_k(\mathbf{x})$$

- a_k : coefficient, typically result of minimization of error function

Boosting procedure



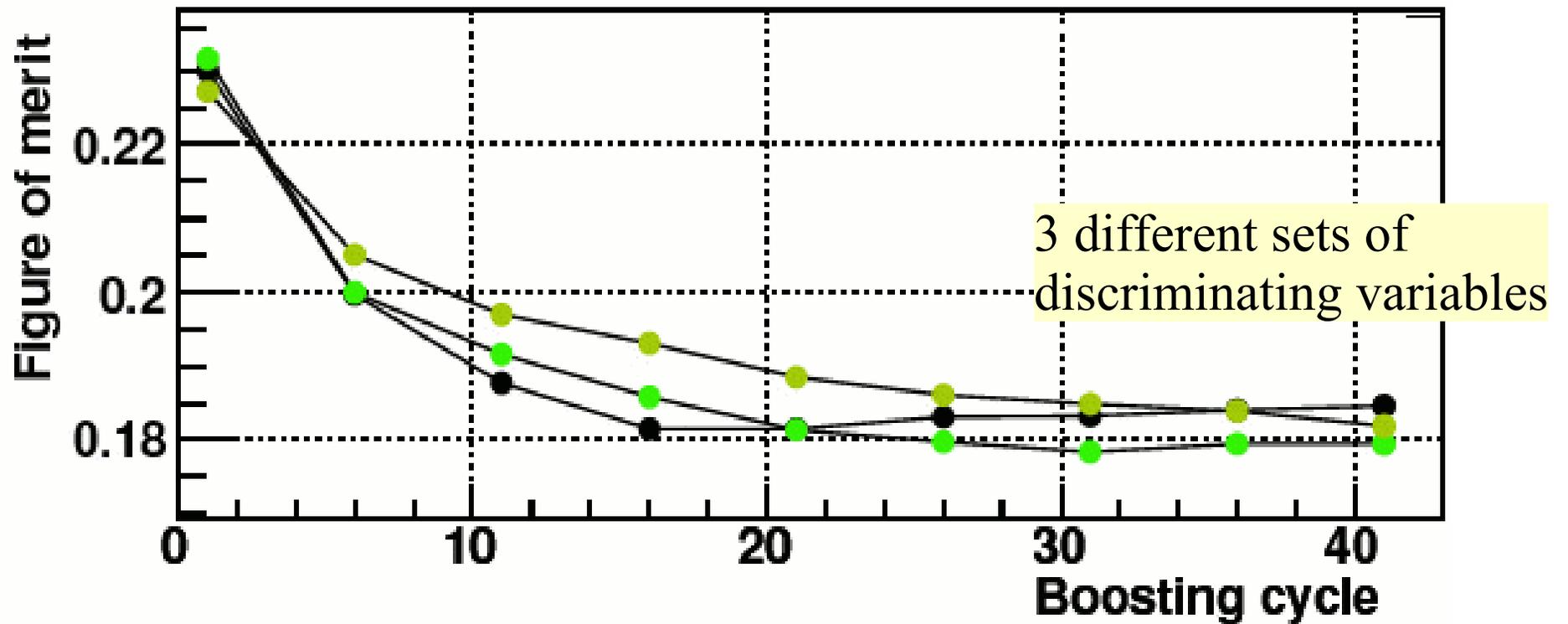
$$F_k = \sum a_k f_k$$



Adaptive boosting

- In each iteration, update coefficient a_k
 - From minimizing error function
 - coefficients decrease at each iteration
- Update weight for each event in training sample T_k
 - Figure out which events have been misclassified
 - Signal events should have purity ≥ 0.5
 - Background should have purity < 0.5
 - Increase event weight for those events that have been misclassified
 - Large weight to background events with large purity and signal events with small purity

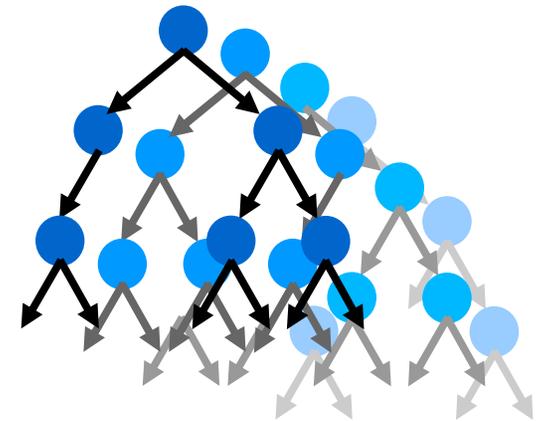
Boosting performance



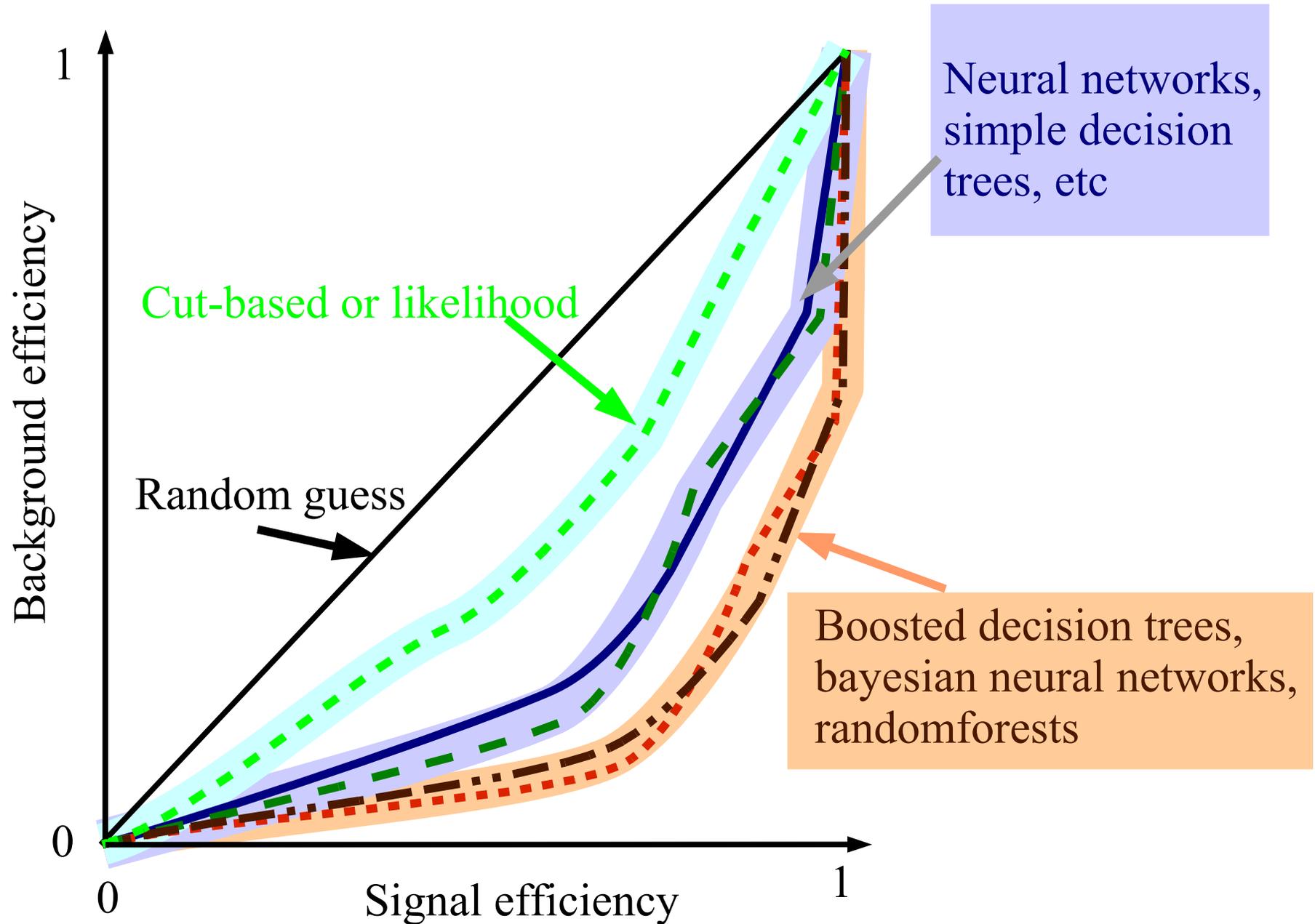
DØ single top
search
with decision trees

Random forest

- Average over many decision trees
 - Typically $O(100)$
- Each tree is grown using m variables
 - For N total variables, $m \ll N$
- Very fast algorithm
 - Even with large number of variables
- Very few parameters to adjust
 - Typically only m



Classifier Comparison



MiniBoone

- Short-baseline search for muon neutrino oscillation to electron neutrino
 - Goal: confirm or rule out LSND result
- One of the first analyses to use boosted decision trees
- 300 variables
 - Reconstructed objects, kinematic distributions, detector-based objects
- In competition with a simple likelihood method
 - Track-based likelihood, TBL
 - Linear combination of variables, ignoring correlations

MiniBoone

- Short-baseline search for muon neutrino oscillation to electron neutrino
 - Goal: confirm or rule out LSND result
- One of the first analyses to use boosted decision trees
- 300 variables
 - Reconstructed objects, kinematic distributions, detector-based objects
- In competition with a simple likelihood method
 - Track-based likelihood, TBL
 - Linear combination of variables, ignoring correlations
- The likelihood method was more sensitive

MiniBoone

- Short-baseline search for muon neutrino oscillation to electron neutrino
 - Goal: confirm or rule out LSND result
- One of the first analyses to use boosted decision trees
- 300 variables
 - Reconstructed objects, kinematic distributions, detector-based objects
- In competition with a simple likelihood method
 - Track-based likelihood, TBL
 - Linear combination of variables, ignoring correlations
- The likelihood method was more sensitive

What happened?

MiniBoone BDT vs TBL

- The MiniBoone likelihood method was more sensitive than the BDT
 - Even though they in principle had the same information, and the BDT should have been much more sensitive
- What happened?
- BDT was trained and likelihood was formed without systematic uncertainties
- Including systematic uncertainties at a later stage
 - BDT performance worsens significantly
 - Likelihood not affected as much

MiniBoone BDT vs TBL

- The MiniBoone likelihood method was more sensitive than the BDT
 - Even though they in principle had the same information, and the BDT should have been much more sensitive
- What happened?
- BDT was trained and likelihood was formed without systematic uncertainties
- Including systematic uncertainties at a later stage
 - BDT performance worsens significantly
 - Likelihood not affected as much

Lesson: Include systematic uncertainties during multivariate optimization

Next generation multivariate analyses

- Deal with systematic uncertainties
 - Or at least mitigate their effects
 - Train without variables that are sensitive to systematics
 - Strong dependence on systematic is reduced
 - Performance is also reduced
 - Train on shifted samples
 - More stable with respect to systematics
 - Overall performance reduced
- Neuroevolution (example: CDF single top combination)
 - Train many neural networks, varying both the network parameters and the network structure
 - Pick one network to use based on performance with systematics
- Use something other than Gini in the DT training and boosting
 - Modern classifiers use event weights, flexible decision making
 - Could use S/B or S/\sqrt{B} or other variants (include systematics)

Signal and background modeling

- All training uses signal MC
 - Usually most backgrounds are also from MC
- Classifier performance depends critically on correct modeling of the data
 - Signal MC must model signal present in the data
 - Background MC must reproduce background in data

If signal model is wrong: search is not sensitive



If background model is wrong: find something that isn't there



- Can usually test background modeling in cross-check samples
 - Reverse one cut or sample with very loose cuts
- Cannot really test signal modeling

Modeling of variables

- All variables used in a multivariate filter must be modeled properly
 - And their correlations
- Since MC is used heavily, this prevents us from using many classes of variables
 - Most detector-variables are not well modeled
 - Tracks, hits, calorimeter energy distribution, muon hits
 - B-tagging is not well modeled
 - D0 b-ID uses tag-rate-functions to give probability to each MC jet
 - Even reconstructed objects aren't necessarily well modeled
 - Jet mass, MET significance, electron likelihood, muon quality, isolation, ...
 - Can only use few high-level variables in training
 - Jet and lepton pt, eta, phi, MET and their combinations

Training on data vs data

- ZH $nb\bar{b}$ is dominated by QCD multijet background (before tagging)

- Estimated from data

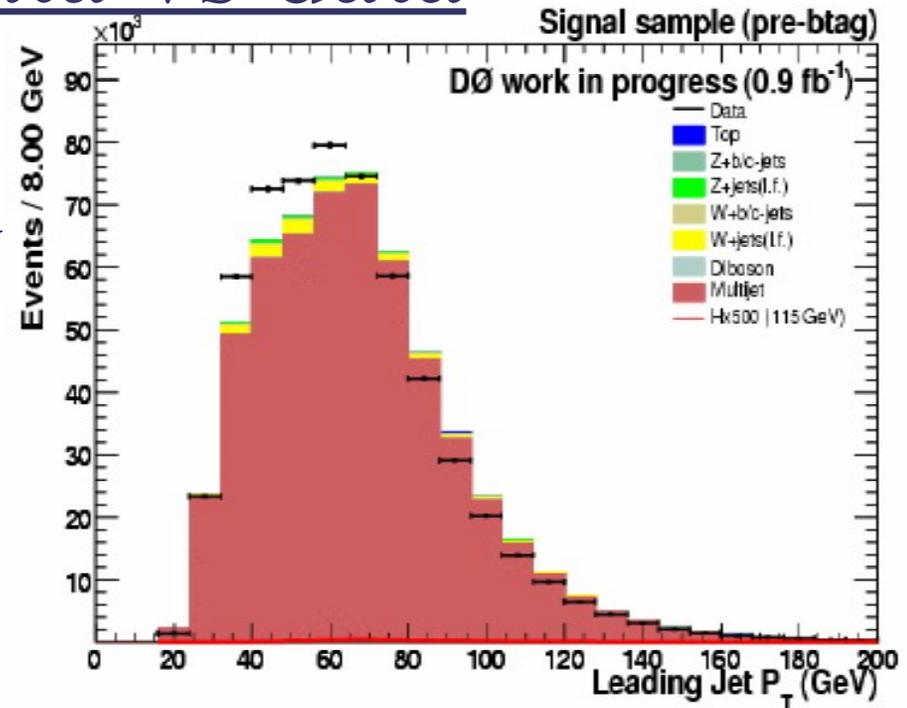
- Train data vs data

- Signal data vs QCD

- Improve QCD rejection
- Allows use of looser b-tagging, looser cuts
- Thus gains signal

- Signal data vs $Z \rightarrow \mu\mu + \text{jets}$ data where muons have been removed

- Signal-like sample



Training data vs data

- Purpose:
 - Effectively eliminate the largest backgrounds
- Procedure:
 - Train on half of data and evaluate on other half
 - Cut on classifier output
 - Estimate cut efficiency for MC from data
- Advantages:
 - Can use all variables, including b-tagging, detector variables, isolation, tracks, etc.
- Disadvantage:
 - How to estimate effect on signal MC?
 - How to guarantee training doesn't introduce bias

Conclusions

- Multivariate event analysis techniques are now a common tool in HEP
 - In the past mostly neural networks, now also decision tree-related methods
 - Glashow, MiniBoone, ATLAS, DZero
- Modern classification tools make life easy
 - Very few parameters to adjust
 - Can use many variables
 - Ranking of variables automatically provided
 - Implemented in several software packages
- Recently, experiments are taking the next step
 - Adapting classifiers specifically for HEP needs

Resources

- PhyStat code repository
<https://plone4.fnal.gov:4430/P0/phystat/>
- PhyStat 2007 conference
<http://phystat-lhc.web.cern.ch/phystat-lhc/>
- Jim Linnemann's collection of statistics links:
http://www.pa.msu.edu/people/linnemann/stat_resources.html
- Statistical analysis tool R
<http://www.r-project.org/>
- TMVA (multivariate analysis tools in root)
<http://tmva.sourceforge.net/>
- Neural Networks in Hardware
<http://neuralnets.web.cern.ch/NeuralNets/nnwInHep.html>
- Boosted Decision Trees in MiniBoone
<http://arxiv.org/abs/physics/0508045>
- Decision Tree Introduction
<http://www.statsoft.com/textbook/stcart.html>
- GLAST Decision Trees
<http://scipp.ucsc.edu/~atwood/Talks%20Given/CPAforGLAST.ppt>