Parallelization and Global PID for PandaROOT

V.Suyam Jothi, M. Babai ,
J.Messchendorp

02-03-2009

KVI Groningen, Netherlands

Overview

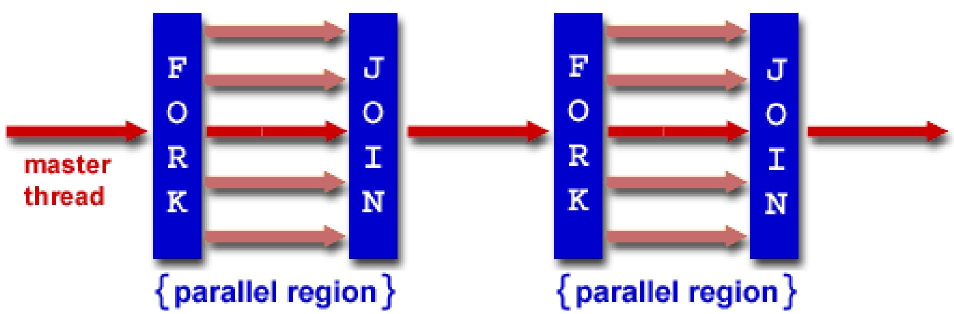
- Parallelization
  - Methods
  - Issues
- Global PID
  - Correlated parameters
  - MVA tools
  - Cross validation
- Summary & outlook

Parallelization

- Open Multi Processing (openMP)
  - M.Babai
- Message Passing Interface (MPI)
  - J.Messchendorp
- Parallel ROOT Facility (PROOF)
  - Klaus Goetzen
- GRID
  - Dan Protopopescu
- Graphical card (GPU)
  - Mohammad Al-Turany

OpenMP

- Shared memory architecture (multi core machine)
- Master thread forks to multiple threads in parallel region
- Standard included in gcc 4.2 and higher

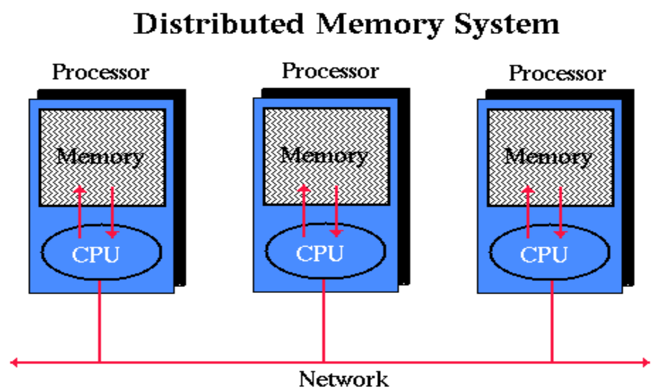


OpenMP in PandaROOT

- Implementation in different modules
  - PndEmcMakeBump – works
  - Lot of effort to make dependencies thread safe
- Track fitting – can be made parallel
  - But dependencies are not thread safe
- Message – Developers have to think about thread safety of the modules ?

MPI in PandaROOT

- Distributed memory architecture
  - Standard in High performance computing
- Example of event level parallelization
- /pandaroot/PndTools/mpiTools - Johan
- http://panda-wiki.gsi.de/cgi-bin/view/Computing/PandaRootTools - Documentation

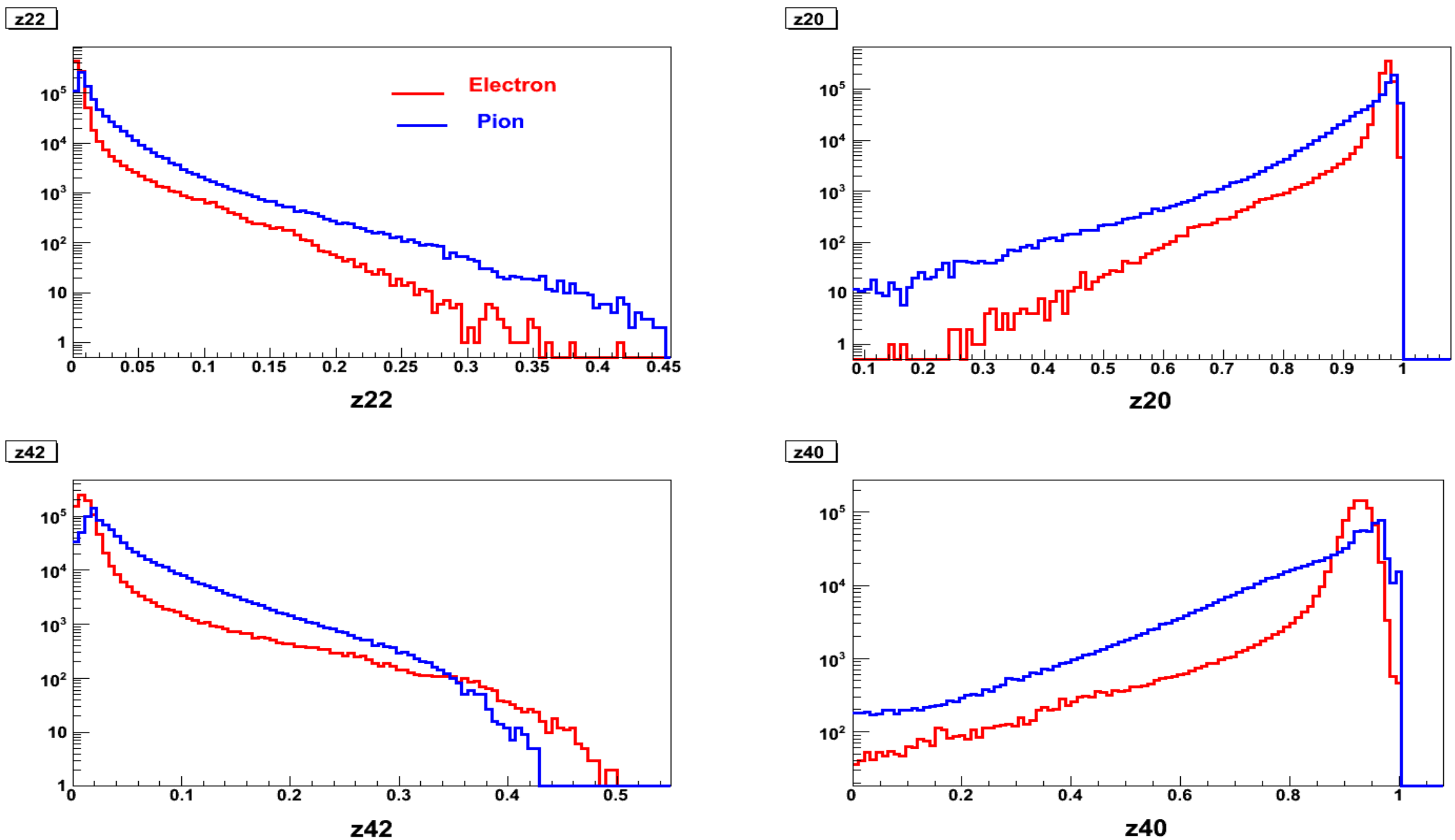


Global PID

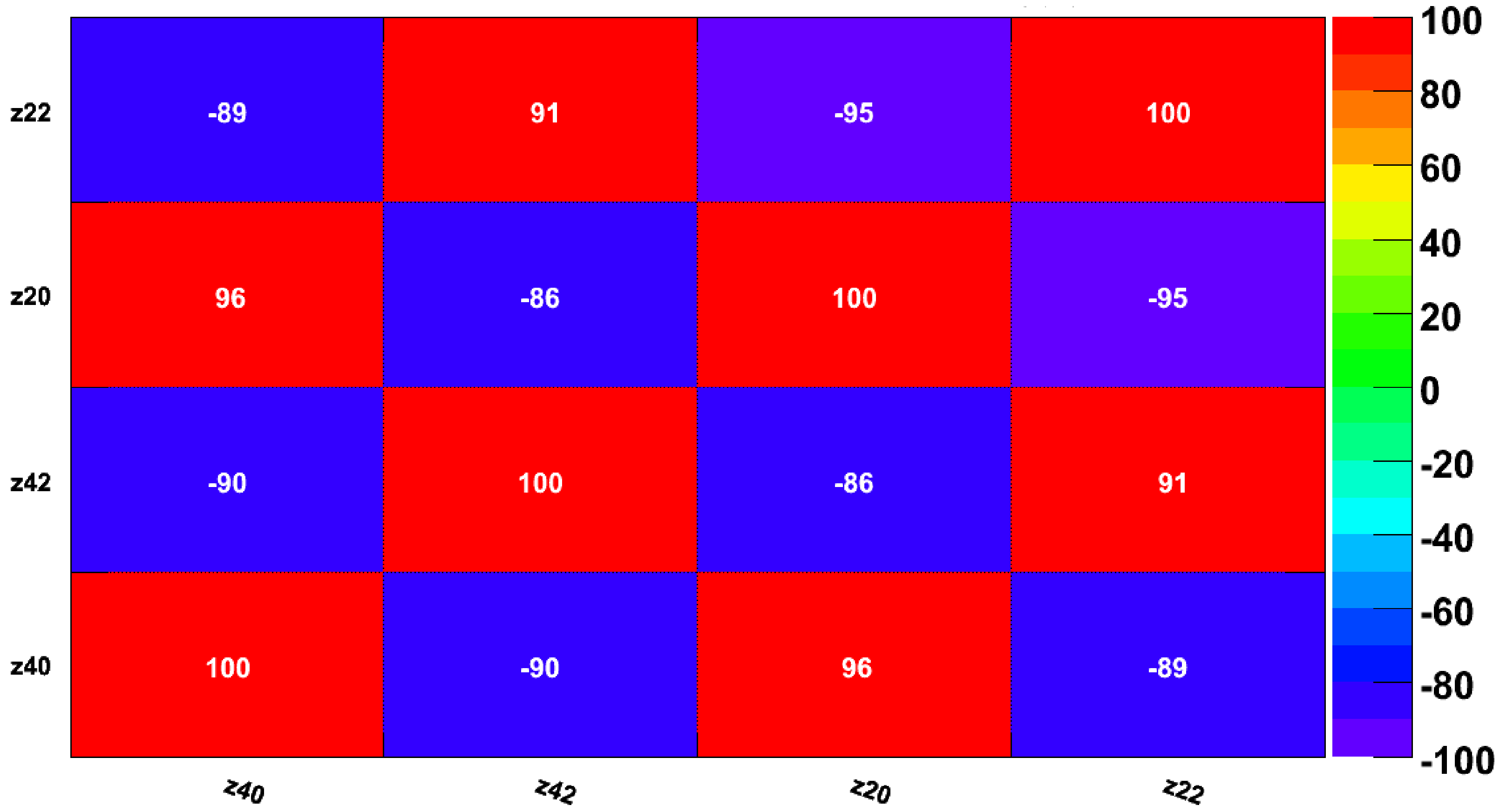
Global

- Global track – requires particle identification (electron, pion, kaon, muon,proton)
- Global PID tool – Classifies tracks
- Likelihoods for particle types
- Projective likelihood – first order solution
- But we have correlated parameters!

Zernike Moments from EMC



Correlation Matrix of zernike moments



Multi Variate Analysis tools

- 1 – dim cuts
- 2 – dim cuts (banana cuts)
- 3 – dim cuts (separating planes)
- What is the solution in the higher dimensions?
- How can one draw cuts in higher dimensions? - Multi Variate Analysis

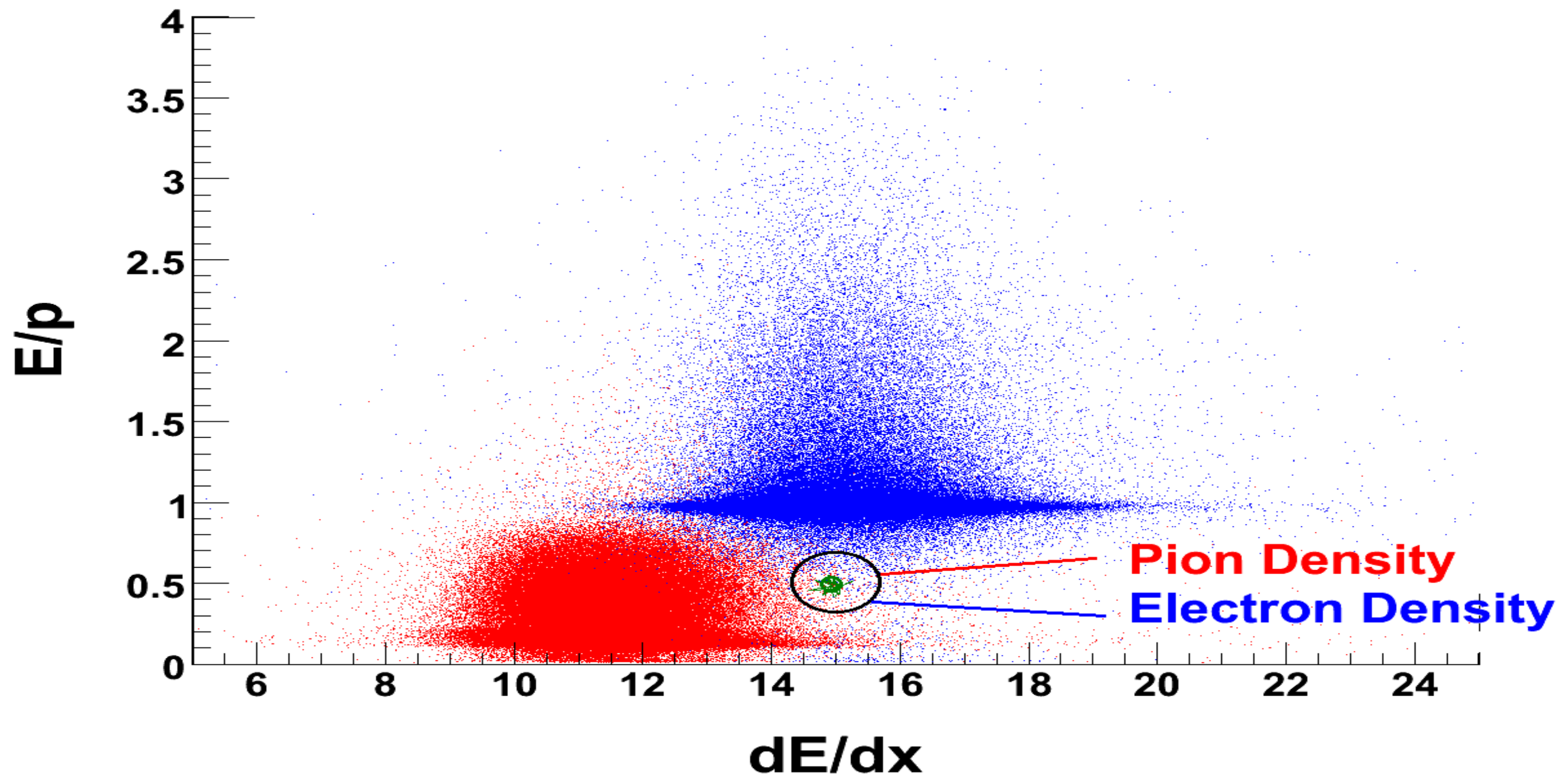
Multi Variate Analysis

- K-Nearest Neighbors – density estimator
  - Large Statistics
- Boosted Decision Tree – Statistical learning
- Learning Vector Quantization – M.Babai
  - /pandaroot/PndTools/MVA/
- Neural Network – Bertram
  - electron/pion seperation

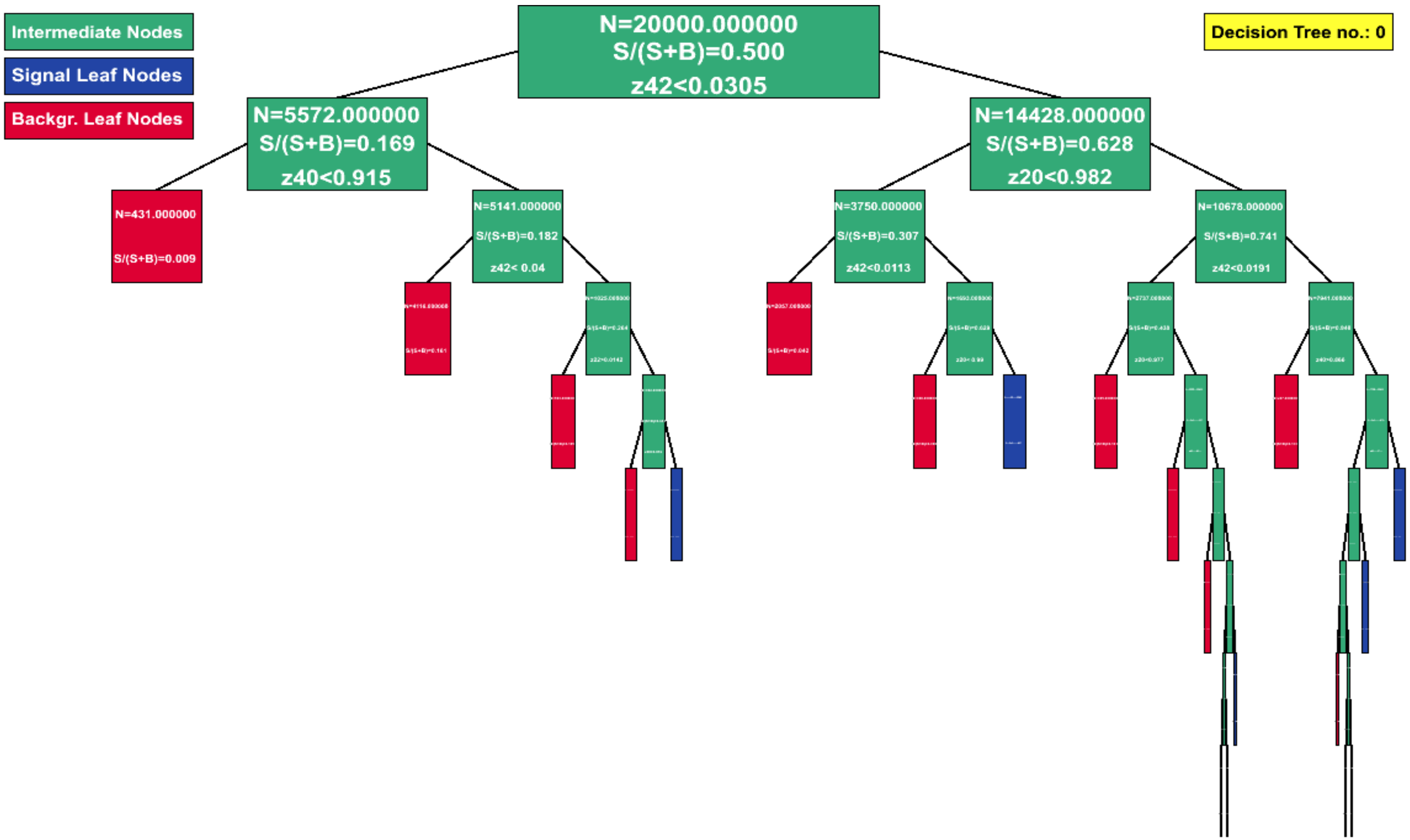
Multi Variate Analysis

- K-Nearest Neighbors – density estimator
  - Large Statistics
- Boosted Decision Tree – Statistical learning
- Learning Vector Quantization – M.Babai
- Neural Network – Bertram
  - electron/pion seperation

K Nearest Neighbors



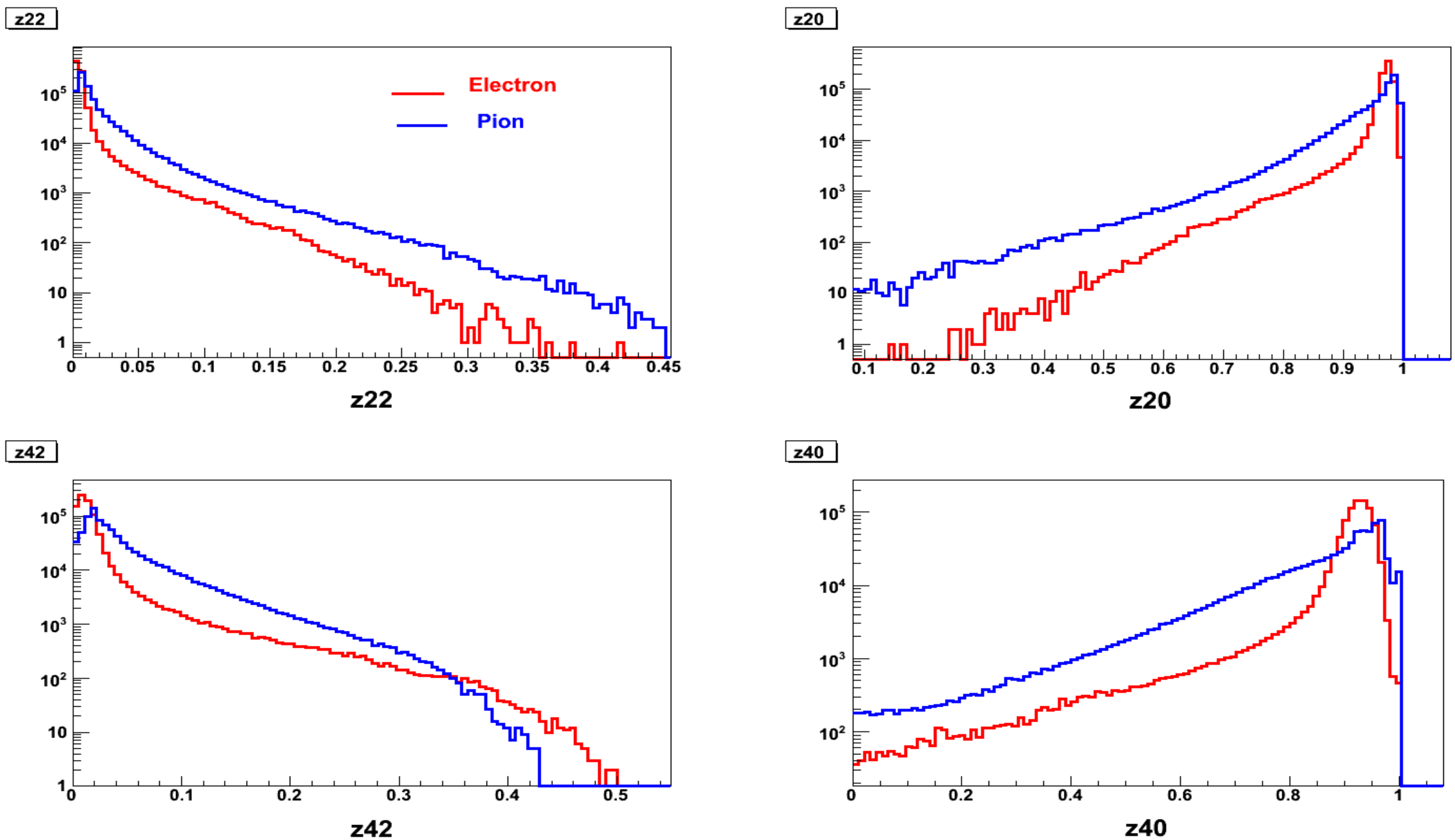
Boosted Decision Tree



Simulation and Analysis

- Full Simulation – PandaROOT
  - electron,pion 10^6 events each in KVI cluster
- Geant3 – Transport model
- Full reconstruction chain
- Tracking – lhetrack
- Momentum 1 - 2 GeV
- TMVA analysis

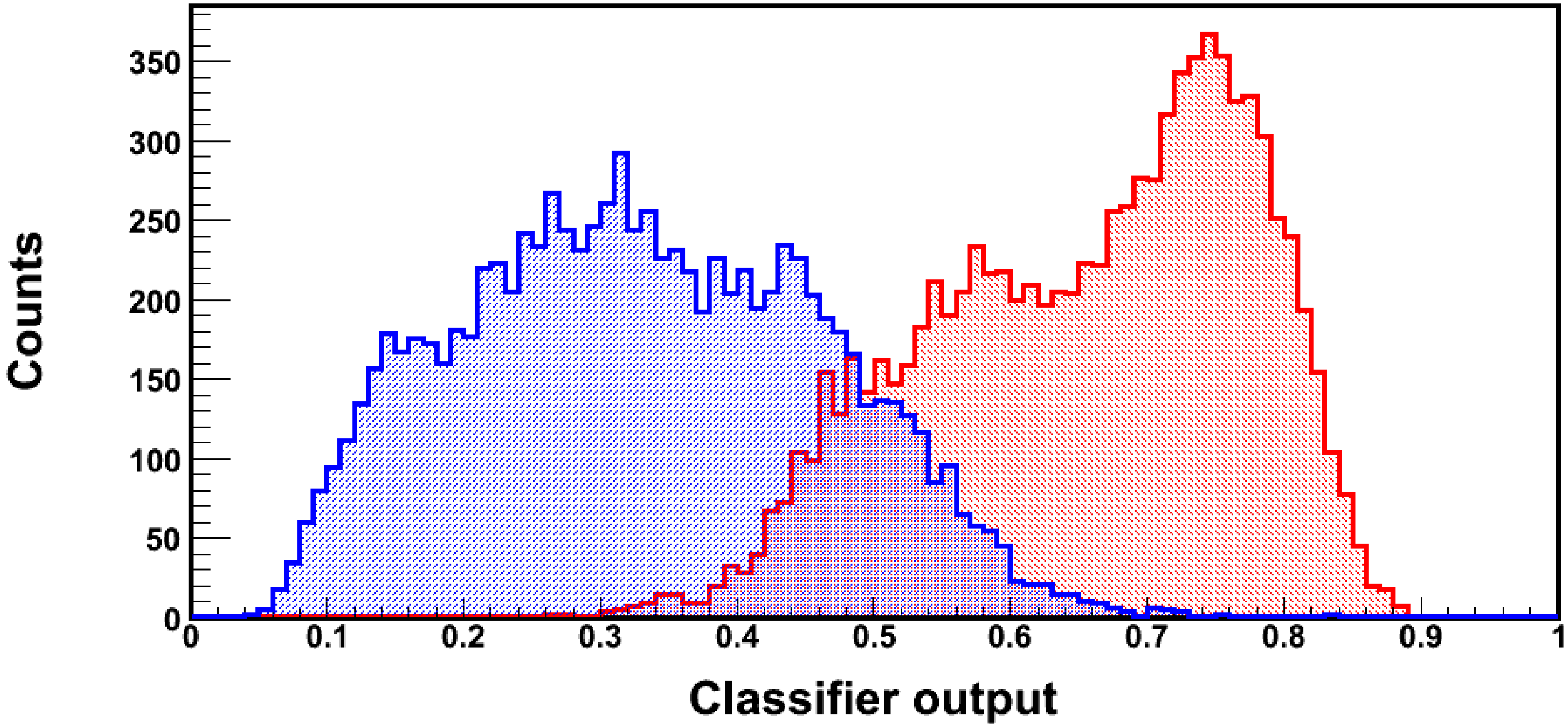
Zernike Moments from EMC



Cross validation of KNN and BDT

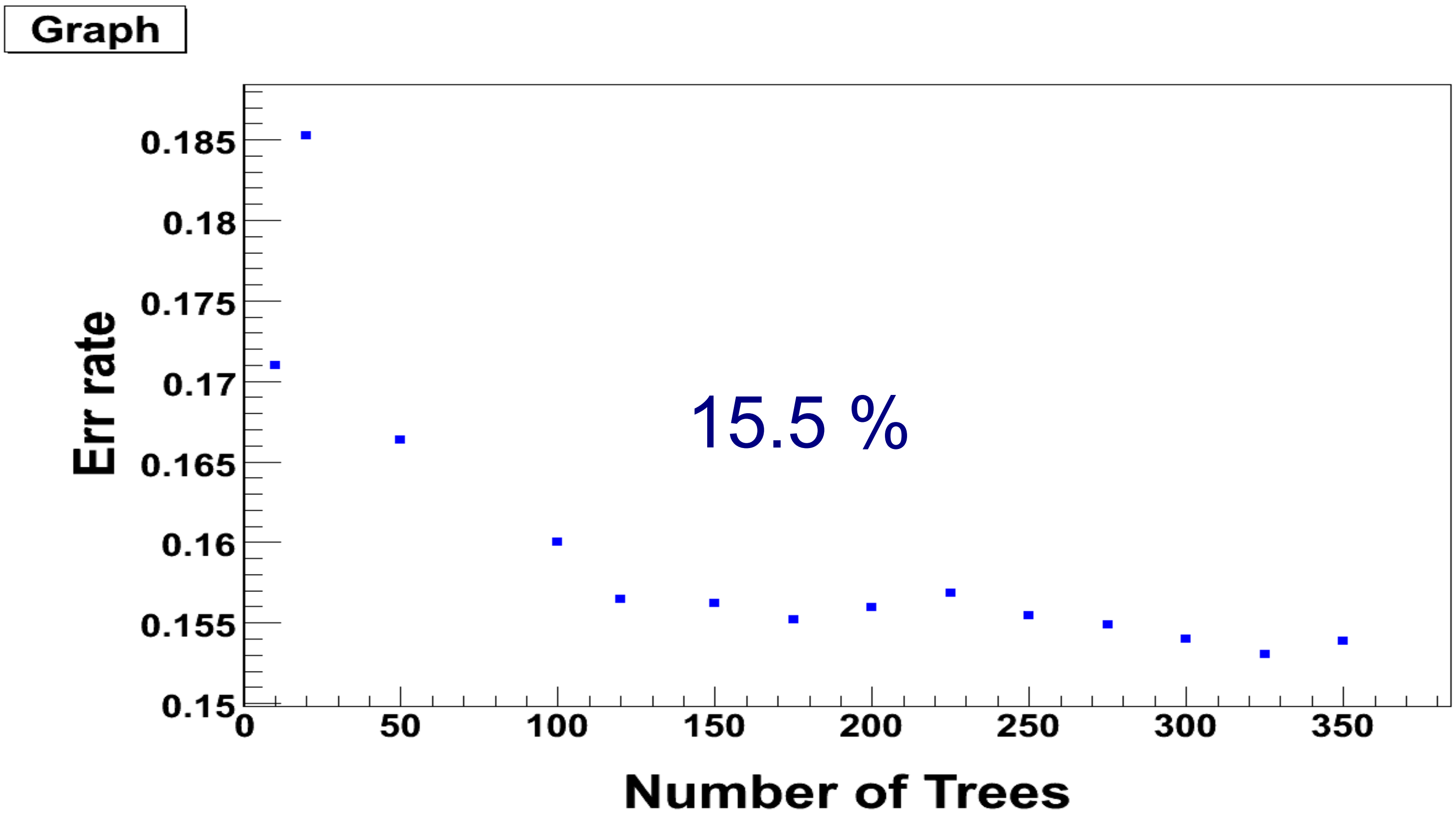
- Performance Optimization(Err rate)
- Learning time
- Classification time
- Resources issues(File size)

MVA Output

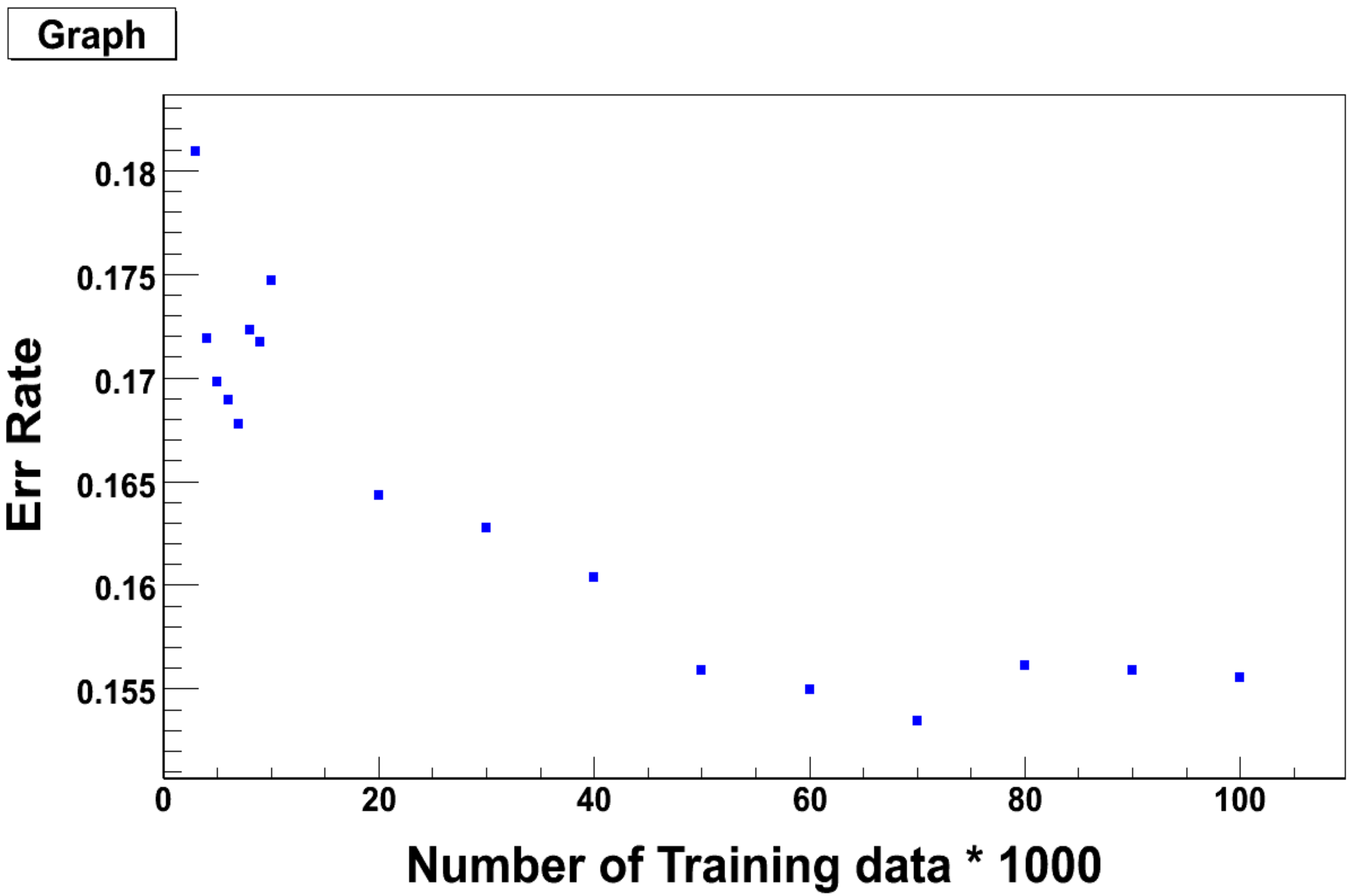


Counts
350
300
250
200
150
100
50
0
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
Classifier output

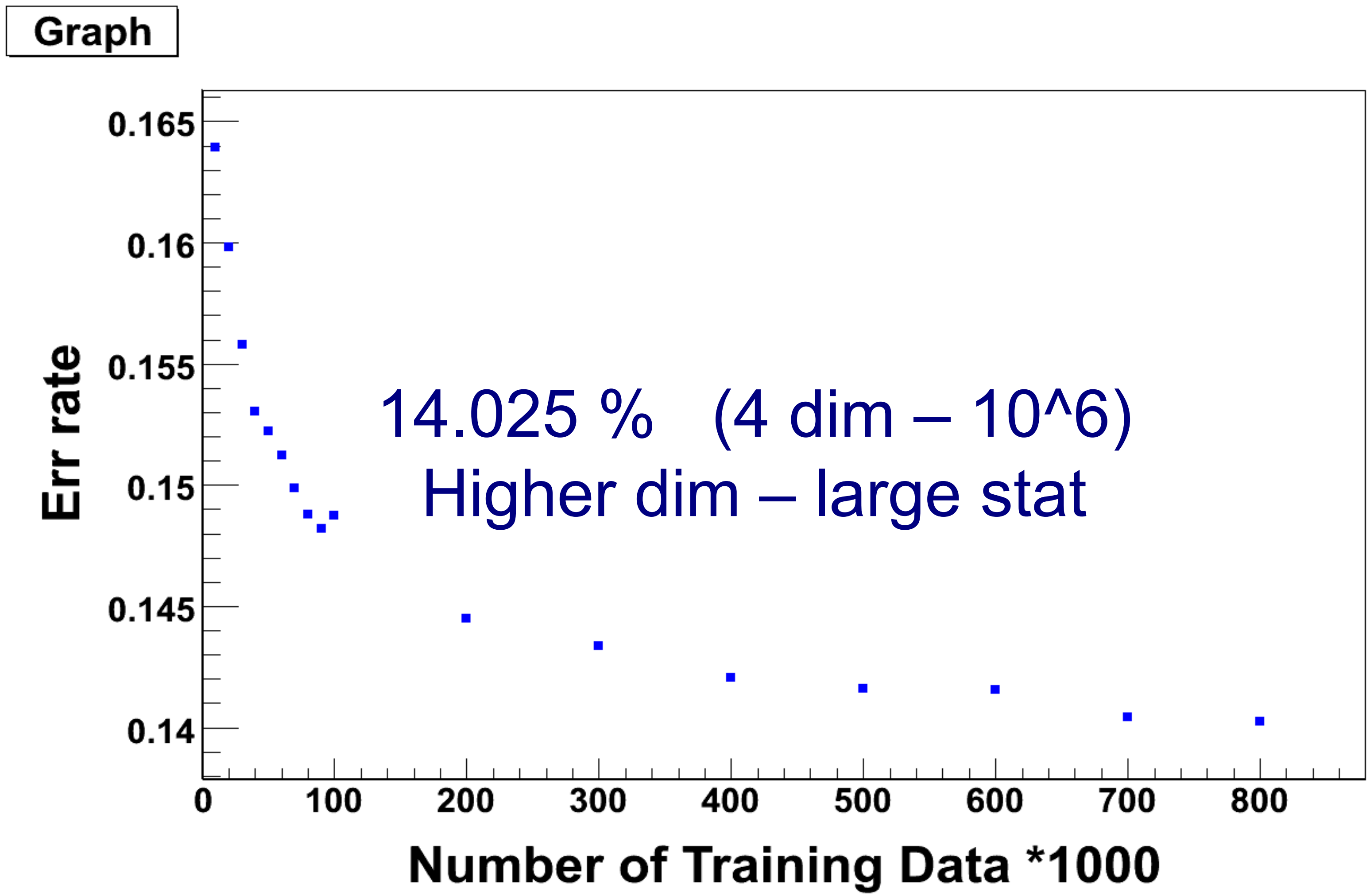
BDT Performance



BDT Performance



KNN Performance



Performance table

	BDT	KNN
Err Rate	15.5 % + overfiiting	14.025 %
Learning time	200 s + production time (1Hz)	20 s + production time (1Hz)
Classification time	0.016 s/track	0.02 s/track
File size	140 Mb(350 Trees)	250 Mb(10^6) 25Gb(10^8)

Summary & Outlook

- Parallelization
  - Various parallel programing techniques are under consideration for PandaROOT
  - Thread safety !
- Global PID
  - MVA analysis necessary.
  - KNN & BDT studied for EMC shower parameters
  - GPID task ready (LVQ , KNN , BDT)
  - Physics benchmark study
  - Other application (photon/pi0 separation -Christian Geldmann)

Criterion for “Best” Tree Split

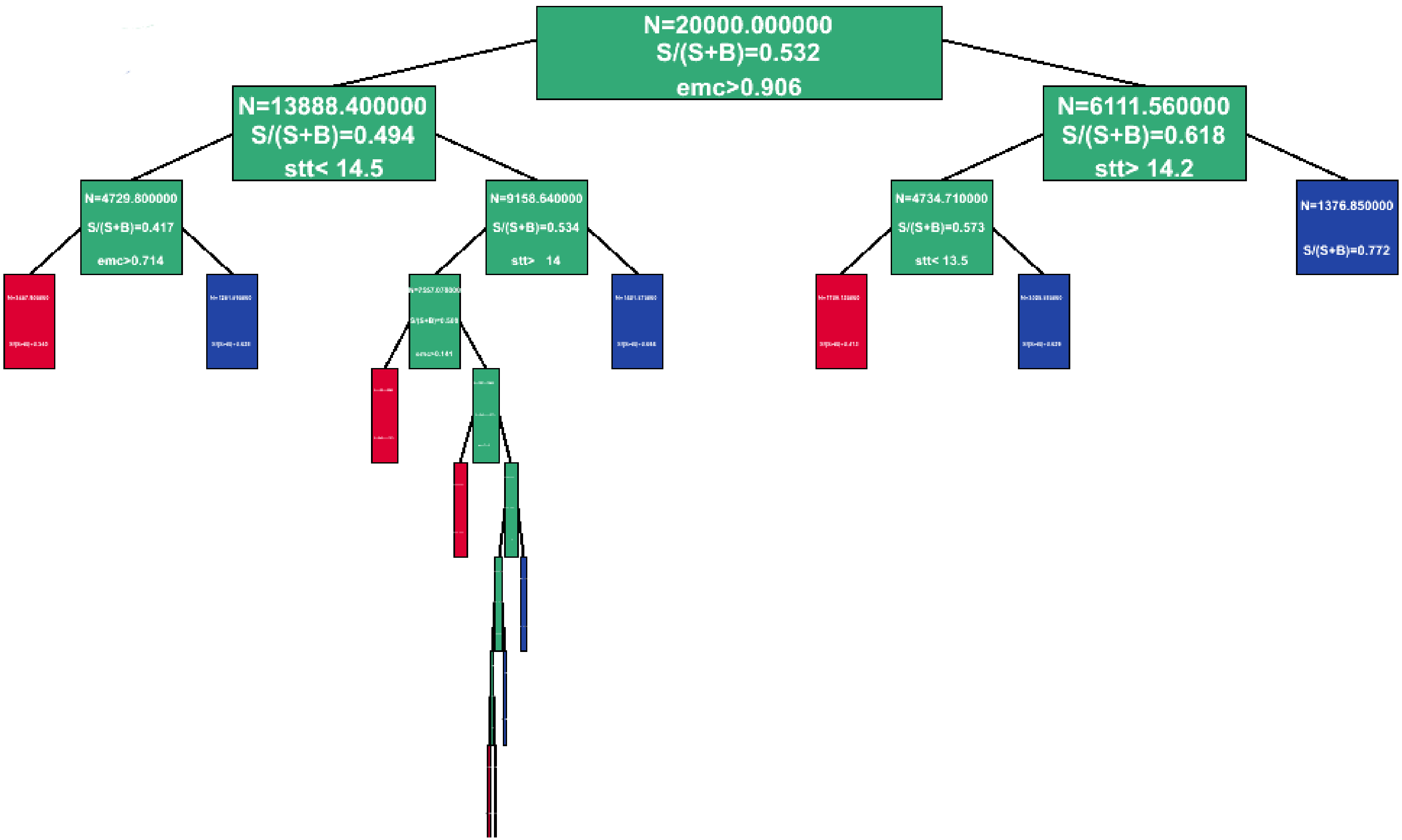
- Purity, *P,* is the fraction of the weight of a node (leaf) due to signal events.
- Gini Index: Note that Gini index is 0 for all signal or all background.

$$Gini = \left(\sum_{i=1}^{n} W_i\right) P(1-P)$$

- The criterion is to minimize

  Gini_left_node+ Gini_right_node.

Decision Tree



AdaBoost

Given: m examples $(x_1, y_1), \ldots, (x_m, y_m)$ where $x_i \in X, y_i \in Y = \{-1, +1\}$

Initialize $D_1(i) = 1$

For t = 1 to T

1. Train learner $\boldsymbol{h_t}$ with min error $\varepsilon_t = \Pr_{i \sim D_t}[h_t(x_i) \neq y_i]$

2. Compute the hypothesis weight $\alpha_t = \frac{1}{2} \ln\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right)$

3. For each example i = 1 to m

$$D_{t+1}(i) = \frac{D_t(i)}{Z_t} \times \begin{cases} e^{-\alpha_t} & \text{if } h_t(x_i) = y_i \\ e^{\alpha_t} & \text{if } h_t(x_i) \neq y_i \end{cases}$$

Output

$$H(x) = \operatorname{sign}\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right)$$

The goodness of h_t is calculated over D_t and the bad guesses.

The weight **<u>Ada</u>**pts. The bigger ε_t becomes the smaller α_t becomes.

Boost example if incorrectly predicted.

Z_t is a normalization factor.

Linear combination of models.

References

- Y.Freund and R.E. Schapire.
  A short introduction to boosting. *Journal of Japanese Society for Artificial Intelligence*, 14(5):771-780, September 1999.