Cross-Validation

TOM STEVENSON

16 SEPTEMBER 2016

MOTIVATION AND THE ISSUE

- Need confidence that the trained MVA is robust:
  - Performance on unseen samples accurately predicted.

- Validation techniques required for:
  - Model Selection:
    - Methods have at least one free parameter e.g.
      - BDT - #trees, min node size, etc.
      - SVM - kernel function, kernel parameters, cost, etc.
    - How are these parameters of models “optimally” selected?

  - Performance Estimation:
    - How does the chosen model perform?
    - Usually true error rate is used (misclassification rate for the entire dataset).

Queen Mary University of London

MOTIVATION AND THE ISSUE

- For an unlimited dataset these issues are trivial, simply iterate through parameters and find model with lowest error rate.
- In reality datasets are smaller than we would like.
- Naïvely use whole dataset to select and train classifier and to estimate error.
  - Leads to overfitting/overtraining as classifier learns fluctuations in the dataset and performs worse on unseen data.
  - Overfitting more distinct for classifiers with large number of tuneable parameters.
  - Also gives overly optimistic estimation of error rate.

K-FOLD CROSS-VALIDATION

- May not be able to reserve a large portion of data for testing:
  - Hold-out method may not be viable.
- Use k-fold cross-validation:

Dataset

Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	. . .	Fold k

  - Split dataset into k randomly sampled independent subsets (folds).
  - Train classifier with k-1 folds and test with remaining fold.
  - Repeat k times.
- Advantage of using the whole dataset for testing and training.
- True error rate is then estimated using average error rate:

$$E = \frac{1}{k}\sum_{i=1}^{k} E_i.$$

Queen Mary University of London

IMPLEMENTATION IN TMVA

- Hyper parameter tuning simply set up and called with:

```
TMVA::HyperParameterOptimisation * hyper = new
            TMVA::HyperParameterOptimisation(dataloader,"ROCIntegral","Minuit");
TMVA::HyperParameterOptimisationResult * hresult = hyper->Optimise(mva,mva,"",folds);
```

- Data splitting done behind scenes in dataloader.
  - Specify number of sig/background events first in usual way.
- Runs OptimiseTuningParameters for each combination of folds.
- Returns one set of hyper parameters per fold.
  - Working on splitting the training sample so validation set can be used to test performance.
- Looking at integrating CV into OptimiseTuningParameters.

Queen Mary University of London

IMPLEMENTATION IN TMVA

- Cross Validation set up and called with:

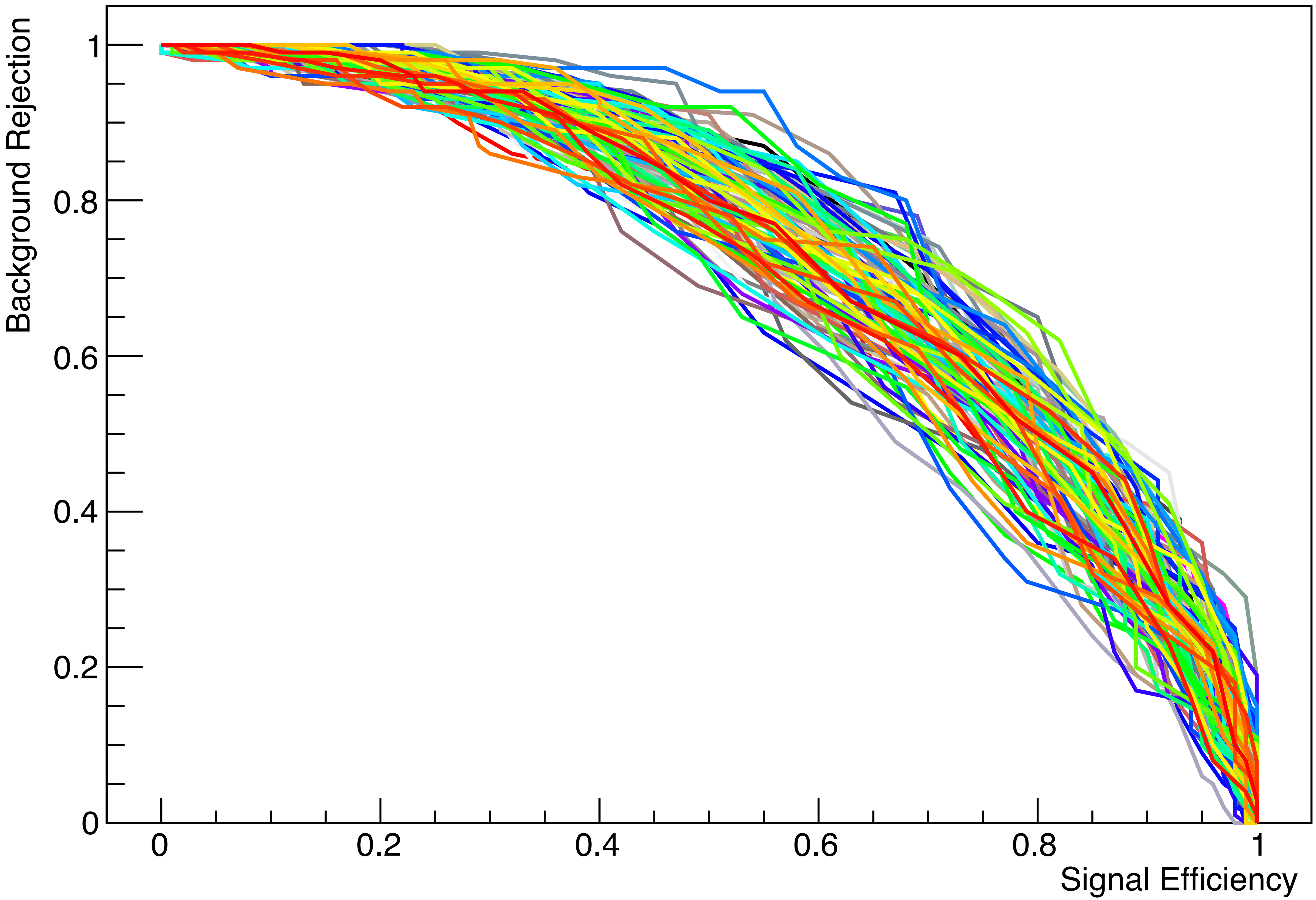
```
TMVA::CrossValidation * cv = new TMVA::CrossValidation(dataloader);
TMVA::CrossValidationResult * result = cv->CrossValidate(mva,mva,"",folds);
```

- CrossValidationResult currently contains some of metrics in EvaluateAllMethods metric in Factory.
  - ROC Integral
  - Separation
  - Significance
  - Efficiencies at different working points.
  - Working on adding more.

EXAMPLE

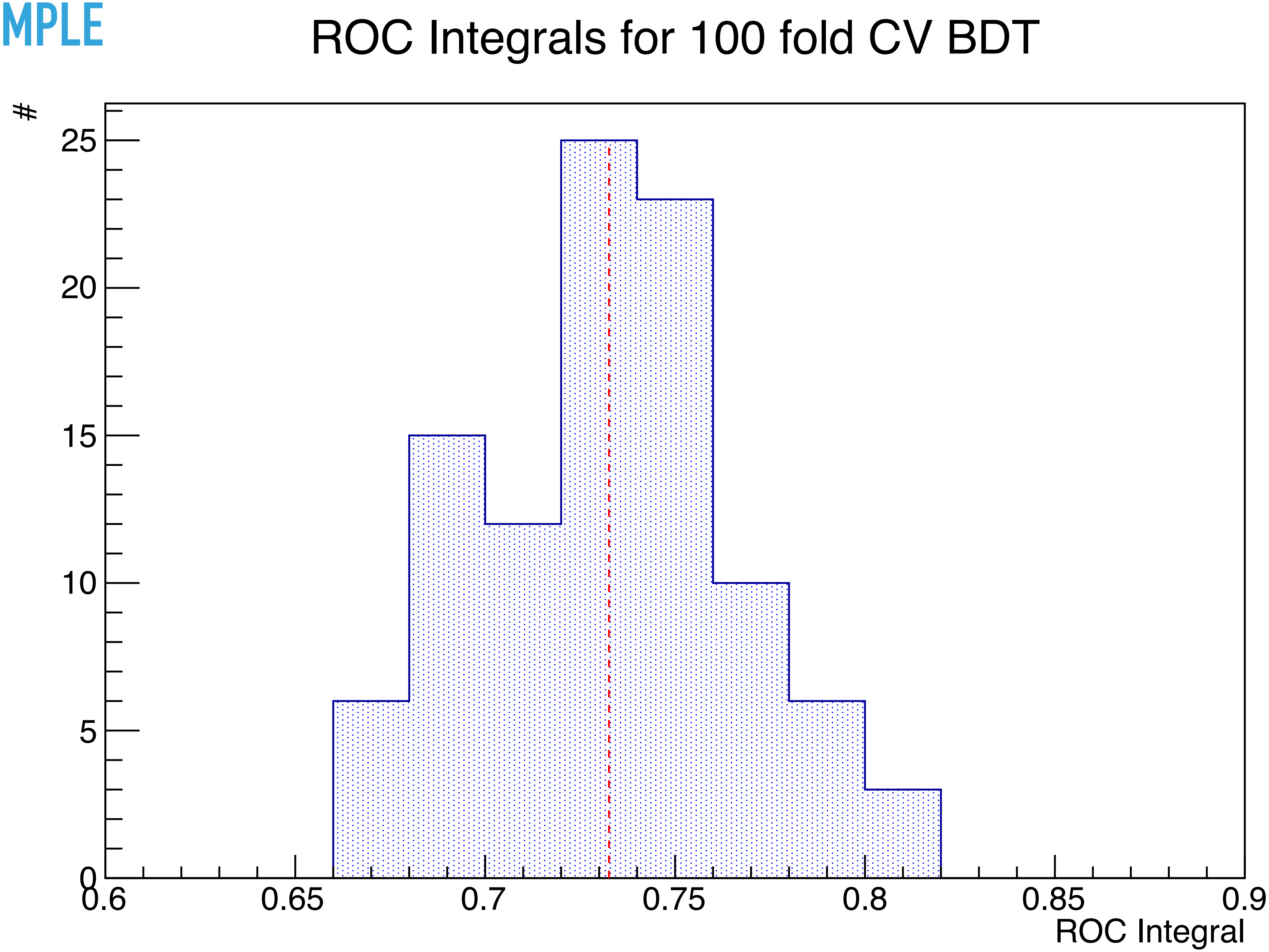
- Dataset:
  - Higgs example set
  - 20000 sig & bkg events.
  - 4 variables:
    - m_bb, m_wwbb, m_wbb, m_jj
- “Out-of-the-box” BDT
- 100 fold cross-validation.

EXAMPLE


Background Rejection
1
0.8
0.6
0.4
0.2
0
0
0.2
0.4
0.6
0.8
1
Signal Efficiency

Queen Mary University of London

EXAMPLE



Queen Mary University of London

SUMMARY

- Basic functionality for cross-validation and hyper-parameter optimisation integrated into TMVA.
- Adding more metrics.
- Investigating other ways to compare performance of classifiers.
- Currently not running in parallel but this will be a welcome improvement.

BACKUP