Queen Mary University of London

Support Vector Machines and Generalisation for HEP

Agni Bethani, Adrian Bevan,
Jon Hays, Tom Stevenson

ACAT 2016
Valparaíso, Chile

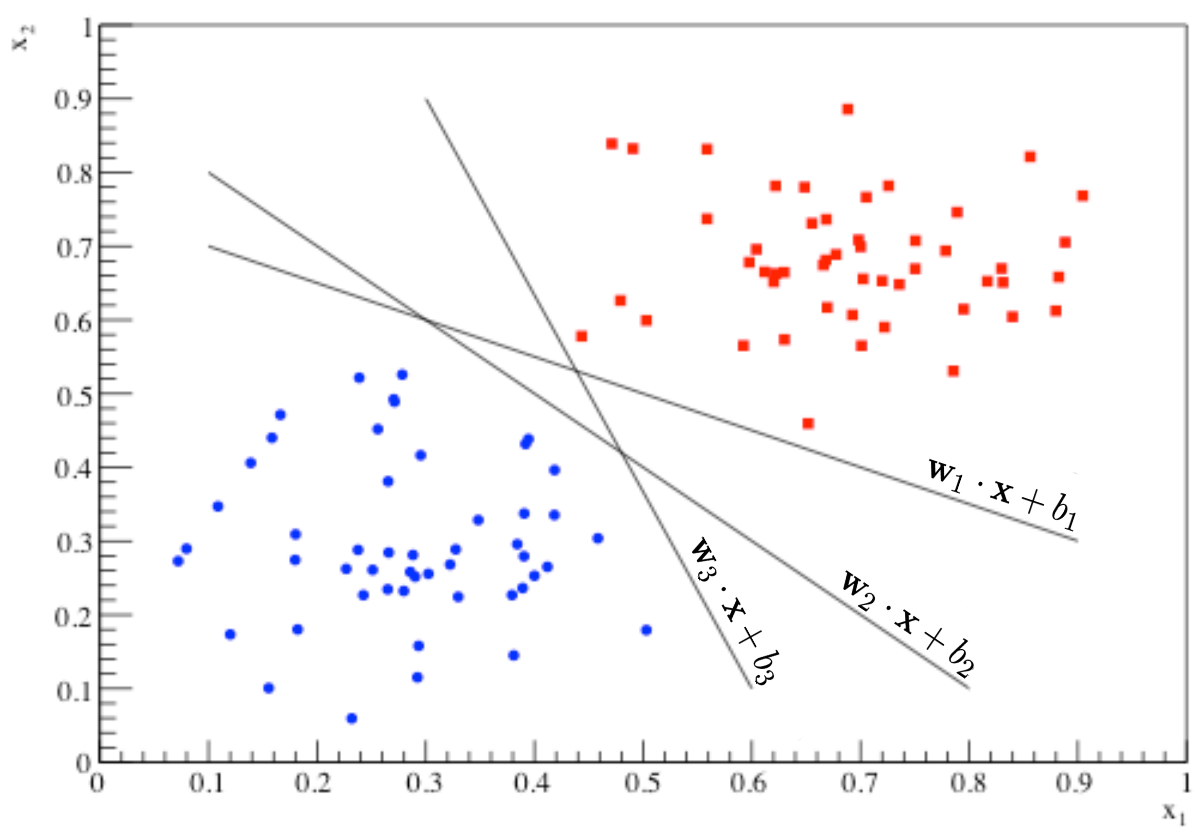
- Support Vector Machines:
  - Overview
  - Hard Margin SVM
  - Soft Margin SVM
  - Kernel Functions and Feature Spaces
  - Checkerboard example
- Generalisation:
  - Motivation and the issue
  - Hold-out method
  - Cross-validation to generalise MVA techniques
  - Checkerboard example again
- Summary

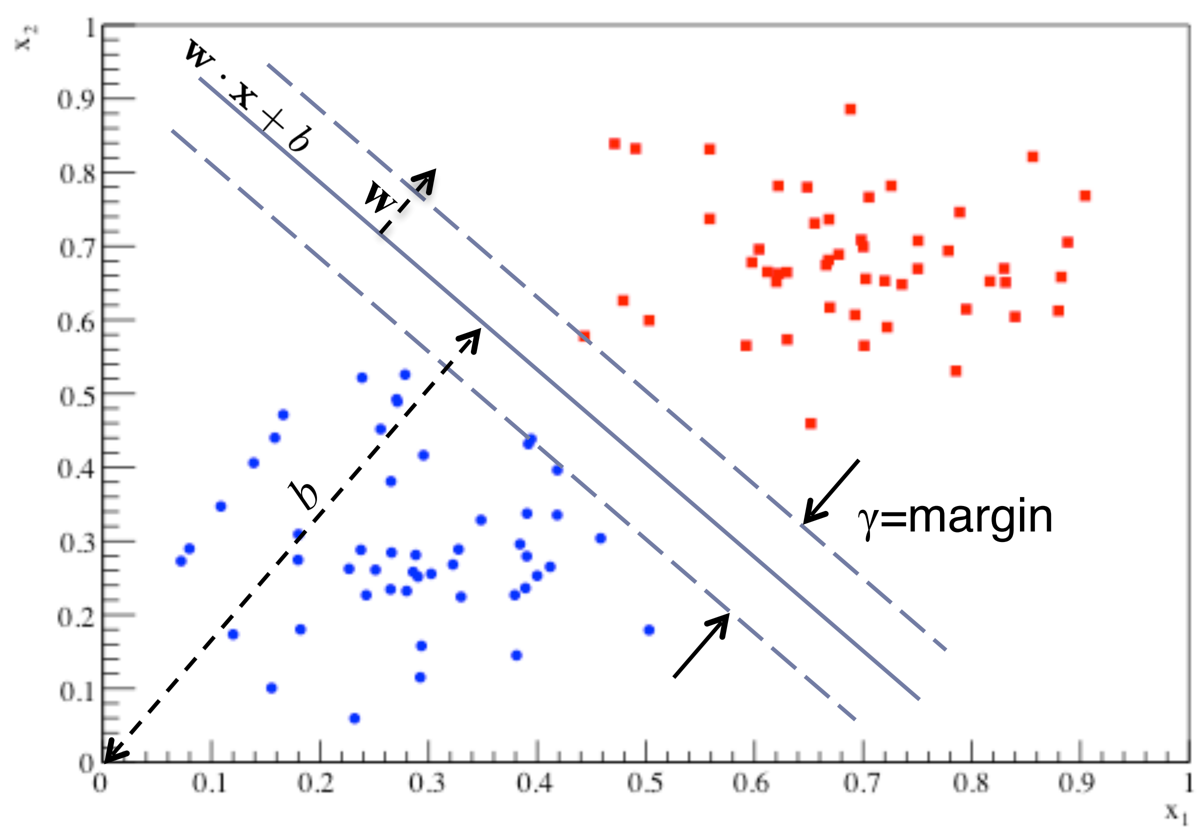
- Linearly separable problems are solved with hard margin SVM.
  - Optimal SVM provides absolute classification, i.e. no classification error.
- Soft margin SVM used for overlapping data samples.
  - Parameters, slack (ξ) and cost (C), introduced to provide penalty and regulate misclassification.
- Kernel functions, $\mathbf{K}$, provide mapping from the problem space ($\mathbf{X}$) to higher dimensional feature space ($\mathbf{F}$).
  - Problem may be separable in this dual space.
  - In practice kernel function is varied to test performance, rather than objectively understanding the mapping.
  - Referred to as Kernel Trick.

Queen Mary University of London

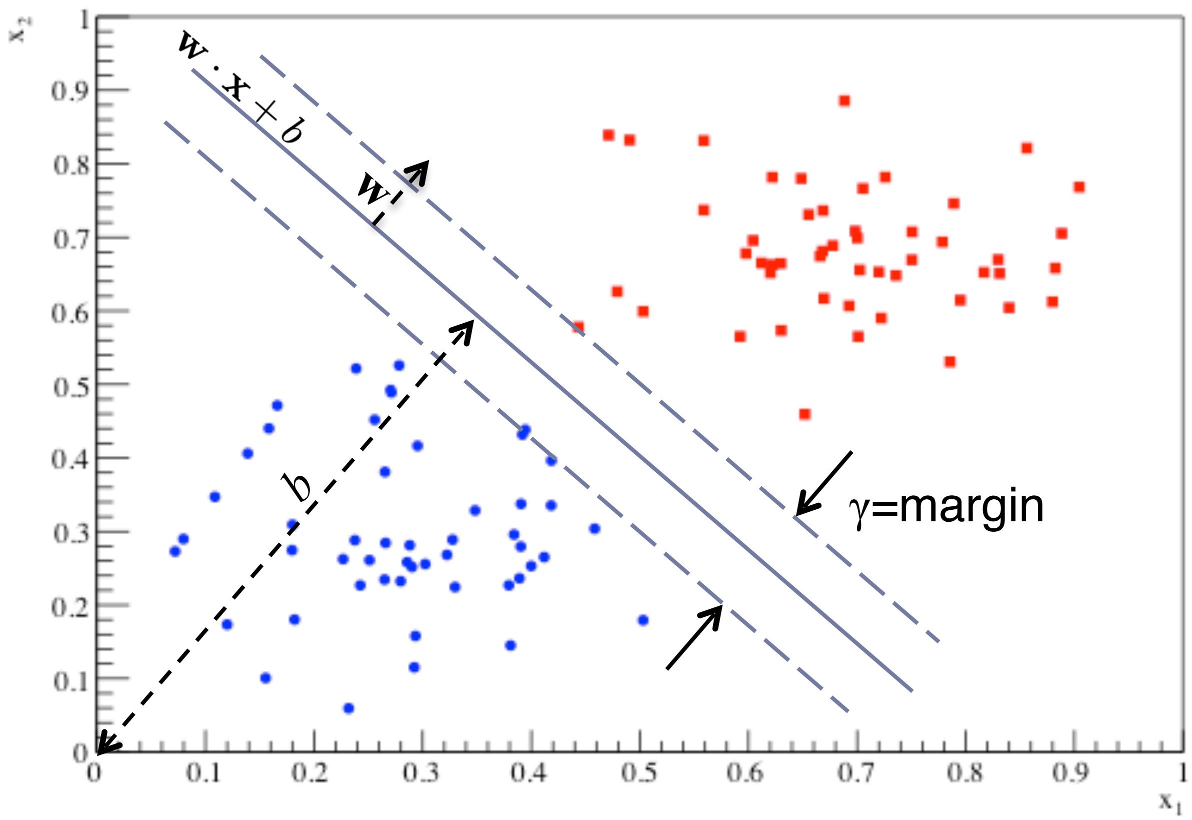
- Discussions of methods and examples using those implemented in TMVA.
  - TMVA is a multivariate analysis toolkit integrated within ROOT.
- Functionality detailed soon available to download as part of the ROOT release.
- Details of usage can be found in the backup slides.

- Object is to find maximally separating hyperplane.
- Achieved by maximising the margin, γ:
  - Distance between the hyperplane and the points closest to the decision boundary, known as the support vectors (SV).
- Simple example:
  - Clearly if this is possible with SVM, cutting on the data would remove the background.





- This problem is solved in the dual space by minimising the Lagrangian for the parameters α_i (Lagrange multipliers):



$$\widetilde{L}(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j)$$

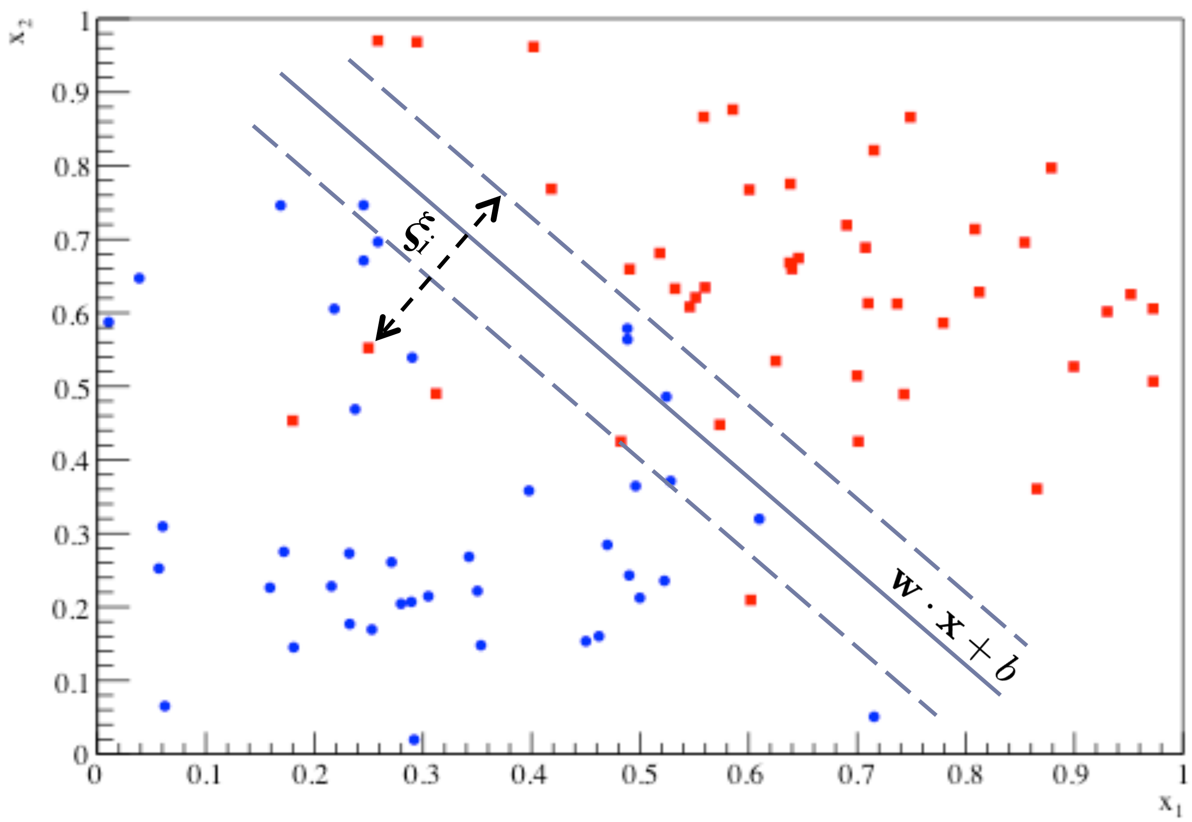
- Subject to: $\alpha_i \geq 0$

and

$$\sum_{i=1}^{n} \alpha_i y_i = 0$$

- Note:
  - α_i are non-zero for support vectors only.
  - The last sum provides a constraint equation for optimisation.

- Relax the hard margin constraint by introducing misclassification.
    - Described by:
        - Slack (ξ_i) - the distance from the hyperplane (defined by the margin) to the ith support vector.
        - Cost (C) - tuneable weight which penalises misclassified points. Multiplies sum of slack parameters.
- More useful for most problems.



- Note: Alternatively described by loss functions, which describe the error rate.

Queen Mary University of London

- The dual form of the Lagrangian simplifies to the same as for the hard margin case:

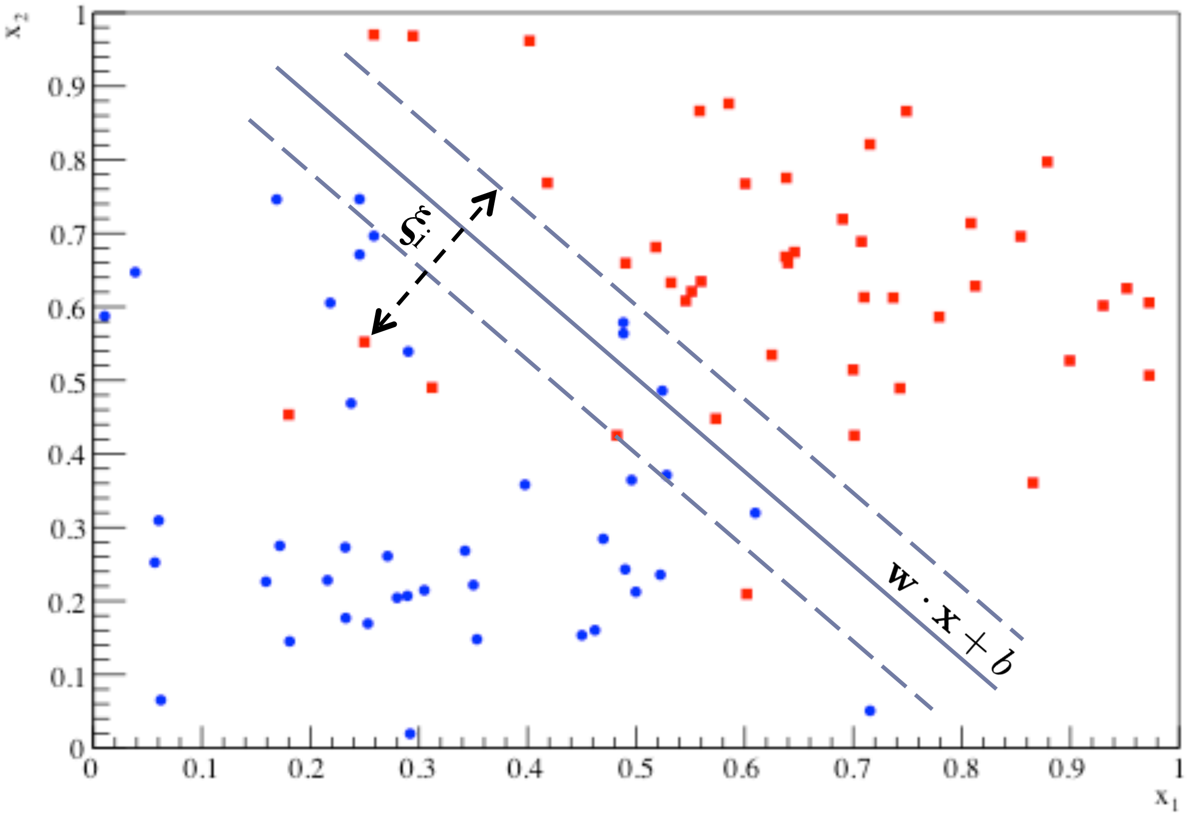
$$\widetilde{L}(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j)$$

where now

$$0 \leq \alpha_i \leq C$$

and as before, the optimisation constraint is given by

$$\sum_{i=1}^{n} \alpha_i y_i = 0$$



Queen Mary
University of London

- The kernel function $K(x,y)$ is used in place of the inner product.
- $K(x,y)$ maps the problem from the input space, X, to a potentially higher dimensional, implicit feature space, $F=\{\phi(x)|x \in X\}$ where the data may then be separable.

$$K(x, y) = \langle \phi(x) \cdot \phi(y) \rangle$$

- Feature map, $\phi(x)$, or the feature space do not need to be known, “kernel trick”.
- Some properties required to be a proper kernel function.
- Inner product defines the identity kernel.

- Polynomial:

$$K(x,y) = (\mathbf{x}\cdot\mathbf{y}+c)^d = \left(\sum_{i=1}^{\dim(X)} \mathbf{x}_i\mathbf{y}_i + c\right)^d \qquad (1)$$

- Radial Basis Function (RBF):
  - Distance between two support vectors, x and y, is computed and used as input to Gaussian KF:

$$K(x,y) = e^{-\Gamma\|x-y\|^2} \qquad (2)$$

- Multi-Gaussian:

$$K(x, y) = \prod_{i=1}^{\dim(X)} e^{-\Gamma_i (x_i - y_i)^2} \qquad (3)$$

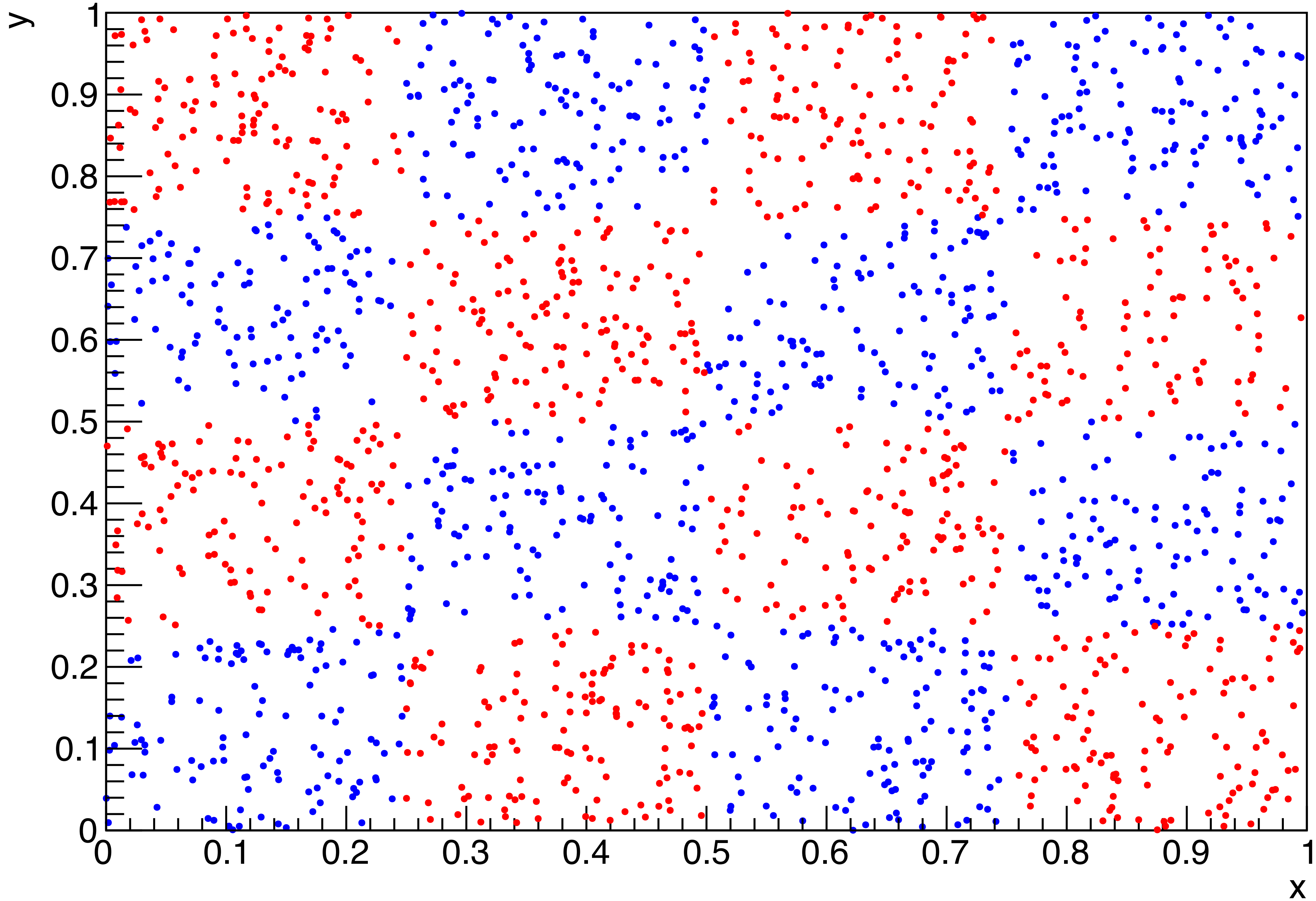
  - Neglects correlations between dimensions in the input data.
  - Further generalised as:

$$K(x, y) = e^{-(\mathbf{x}-\mathbf{y})^T \Sigma^{-1} (\mathbf{x}-\mathbf{y})}$$

  - Σ is the $n \times n$ covariance matrix, where $n = \dim(X)$.
  - Can be computationally expensive, so usually assumed diagonal.

Queen Mary University of London

- Example dataset:
  - 1000 signal (blue), 1000 background (red)
  - Checkerboard arrangement



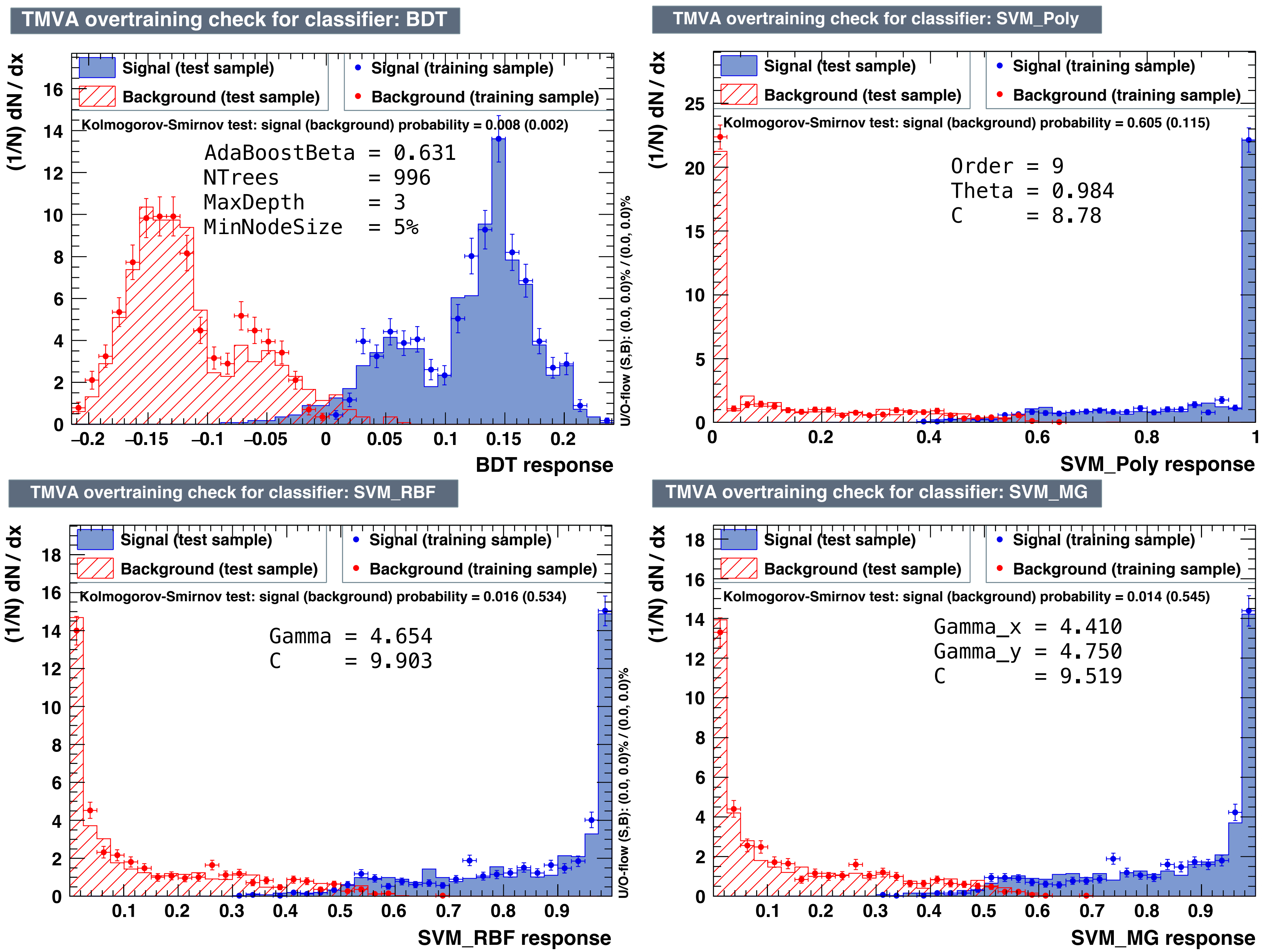
SVMs and Generalisation for HEP

- SVM - Checkerboard Example

  - MVAs optimised and trained using <u>TMVA</u> out-the-box
  - Parameters fully optimised using hold-out method
  - Boosted Decision Tree (BDT) for comparison

BDT

SVM Polynomial Kernel (1)



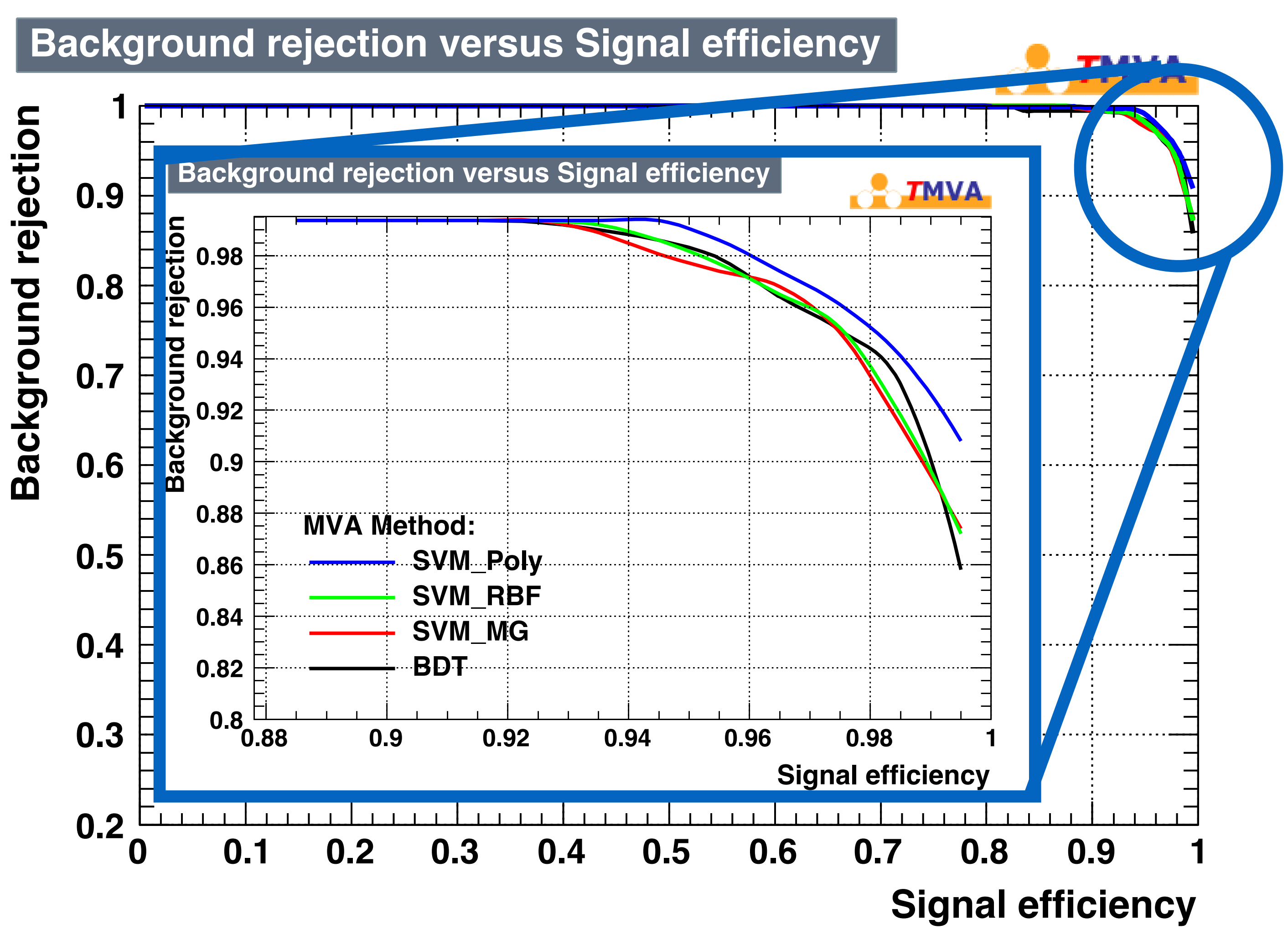
SVM RBF Kernel (2)

SVM Multi-Gaussian (3)

SVM - Checkerboard Example

Queen Mary University of London

- ROC curve shows similar performance for all MVAs.
- We cannot be sure that these solutions are fine tuned.
- Require a method to confirm this, i.e. generalisation.



- Note on measures for generalisation:
  - TMVA computation uses a binned KS test (as on previous slide) which is not uniformly distributed and therefore should not be taken “literally” as a quantified metric of if a classifier is overtrained.

Queen Mary University of London

- Need confidence that the trained MVA is robust and the performance on unseen samples can be accurately predicted, i.e. generalised.

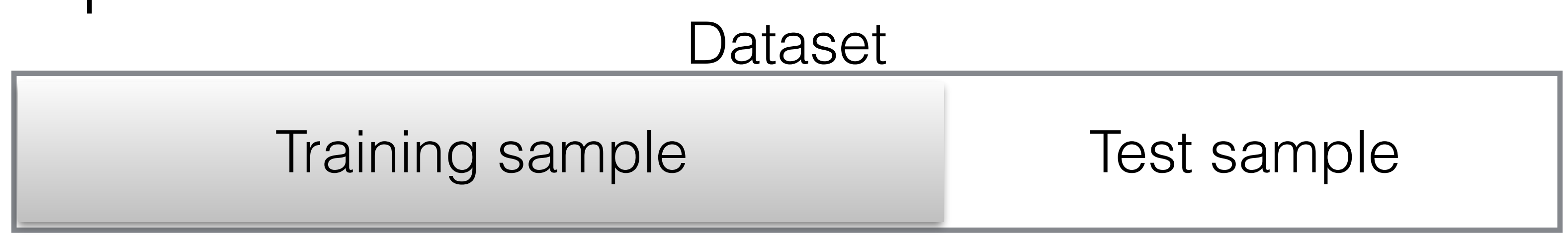
- This motivates validation techniques which are required for:
  - Model Selection:
    - Most methods have at least one free parameter e.g.
      - BDT - #trees, min node size, etc.
      - MLP - #neurons, #layers, weight vectors, etc.
      - SVM - kernel function, kernel parameters, cost, etc.
    - 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).

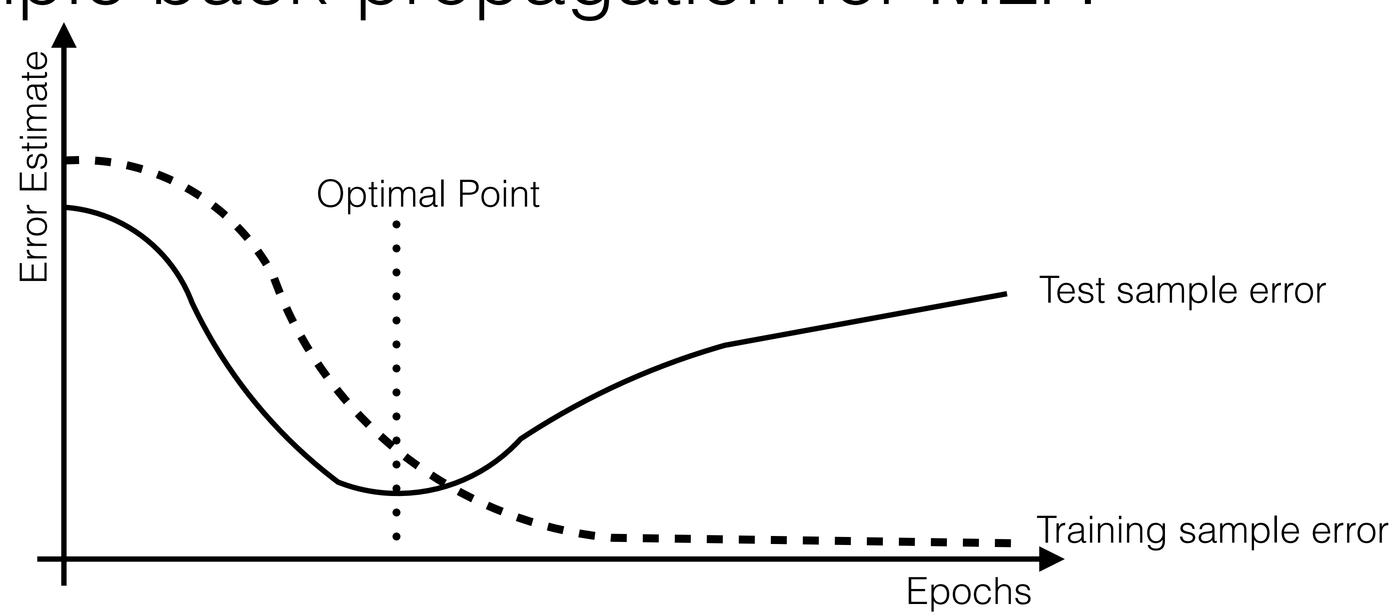
Generalisation - The Issue

- 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.
- See the recent review by S. Arlot and A. Celisse on "Cross-validation procedures for model selection" in Statistics Surveys Vol. 4 (2010) 4079, and references therein for a more detailed discussion on cross validation.

- Potential way to overcome these issues is use hold-out technique, splitting the dataset into training and test subsamples.



- Can use these datasets to select “optimal” parameters, for example back-propagation for MLP.



- Can give misleading error estimate depending on how the data is split.

Queen Mary
University of London

- May not be able to reserve a large portion of data for testing, so hold-out method may not be viable.

- Instead can use *k*-fold cross-validation:

Dataset

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

  - Split the 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:

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

- Generalisation - *k*-fold Cross-validation

Queen Mary University of London

- How many folds???

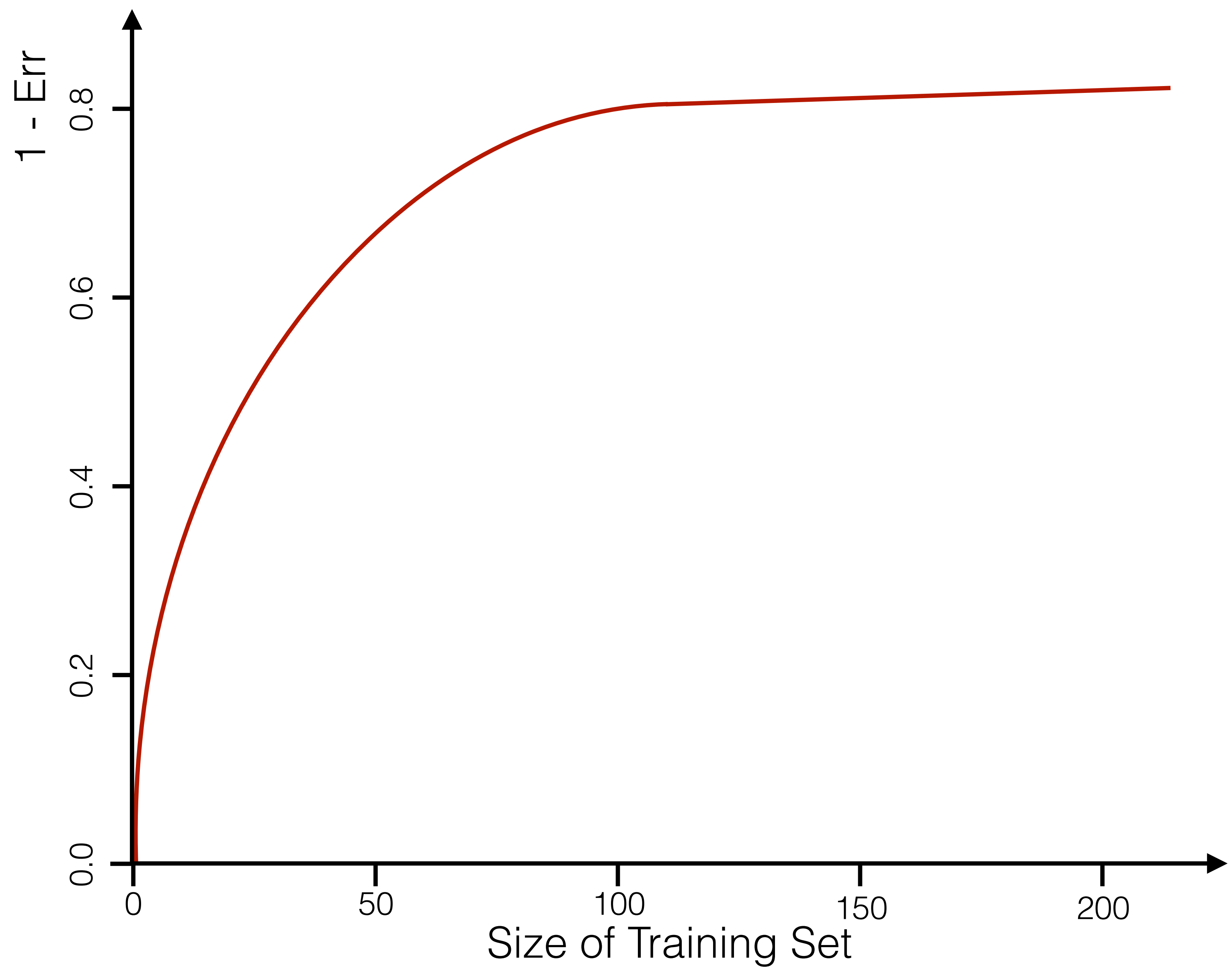
- Large number of folds:
  - Good estimate of average error rate (bias of the estimator is small).
  - Variance of the estimator is large.
  - Computational time is long.

- Small number of folds:
  - Poor estimate of average error rate (bias of the estimator is large).
  - Variance of the estimator is small.
  - Computational time is relatively short.

- In reality choice is motivated by the size of the dataset, i.e. sparse dataset need extreme of leave-one-out method to train on as much data as possible.

Queen Mary
University of London

- Hypothetical example:
  - For sample size of 200, 5 fold CV will estimate the error with similar performance on training set of 160 to that of the full sample.
  - However for sample of 50, 5 fold CV will give a larger error than not using CV.

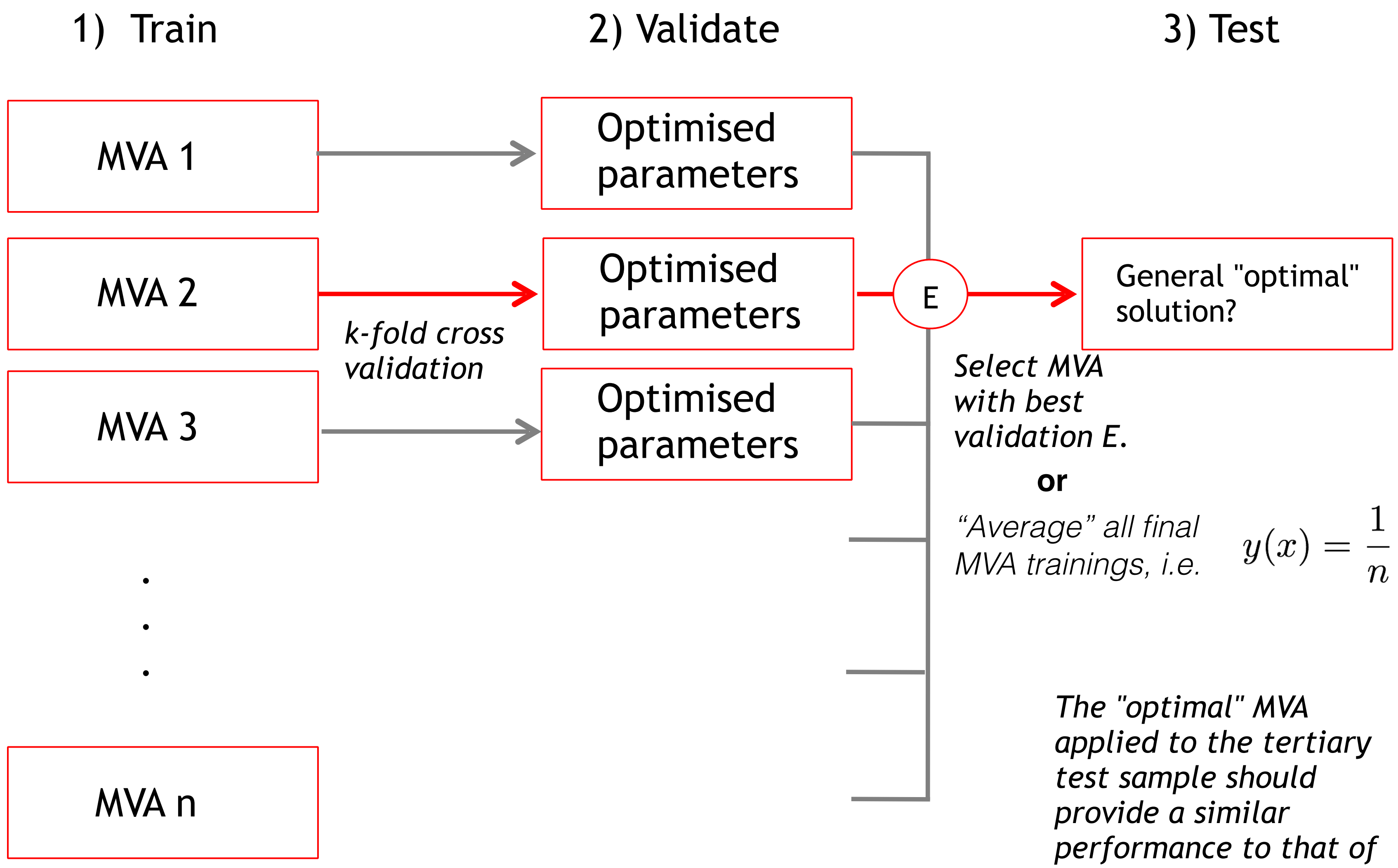


- Common choices are between 5 & 10 folds, however *k* **should** be determined for the given problem.

- Ideally 3 statistically independent dataset; training, validation and testing.
- Training and validation sets used to choose classifier/ model and tune parameters.
- Test set used to assess performance of final fully trained classifier.
- Avoids bias from using the same sample for model selection and parameter tuning.

- Taking the "best" performing MVA doesn't necessarily give the desired output.
  - e.g. some pathologies in distributions.
- Also involves throwing away a large number of trainings.
- Take the aggregated output of all the final trained MVAs on the test sample in some form of average.

Generalisation - *k*-fold Cross-validation



Select MVA with best validation E.

or

"Average" all final MVA trainings, i.e. $y(x) = \frac{1}{n}\sum_{i=1}^{n} y_i(x)$

MVA algorithms with different generic architectures, and weight parameters to be determined.

Each optimised MVA has a corresponding validation error E. This FOM can be used to determine the "optimal" MVA to use with the test sample.

The "optimal" MVA applied to the tertiary test sample should provide a similar performance to that of the training and validation sample if the MVA is to be considered general.

If not; start the process again.

Courtesy of Adrian Bevan

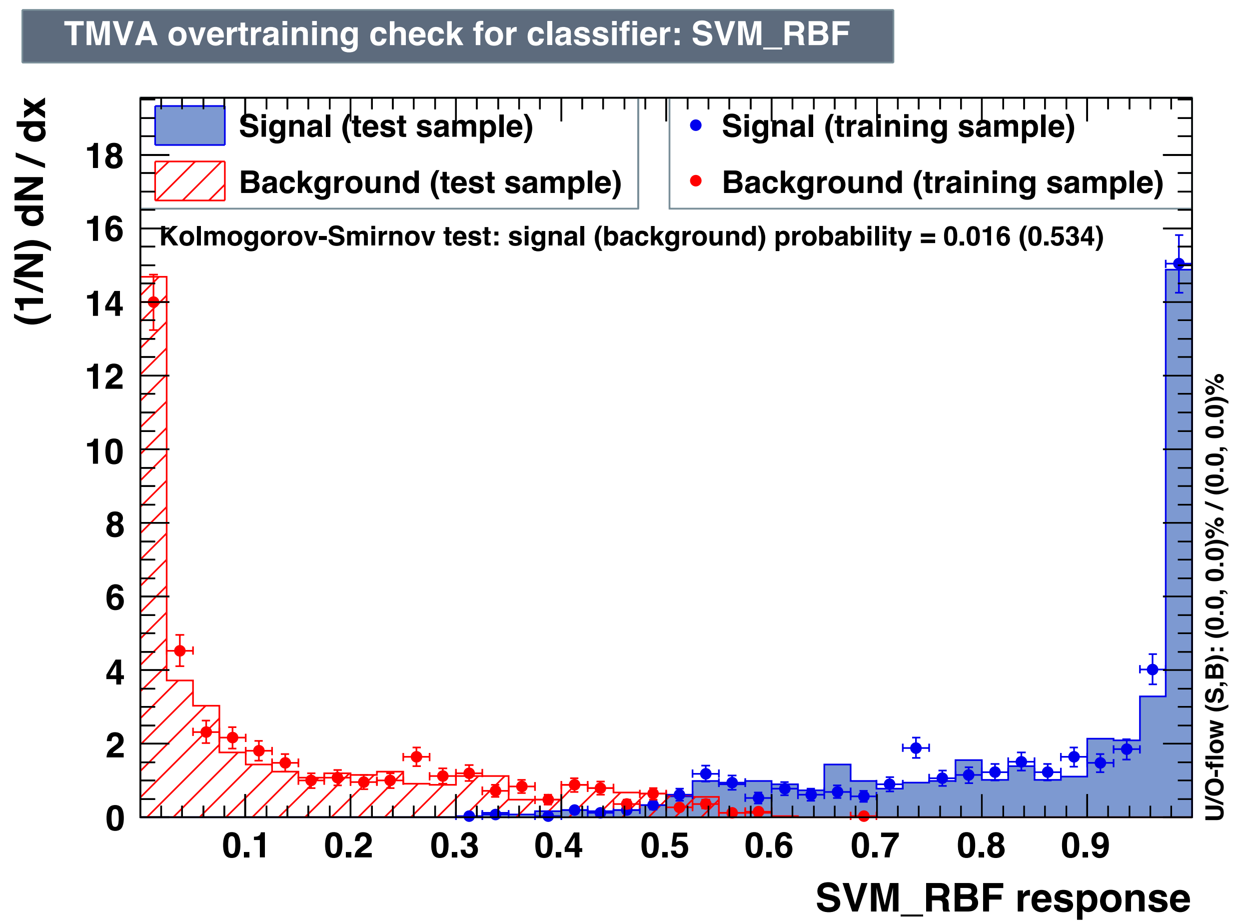
SVMs and Generalisation for HEP

- Generalisation - Checkerboard Example

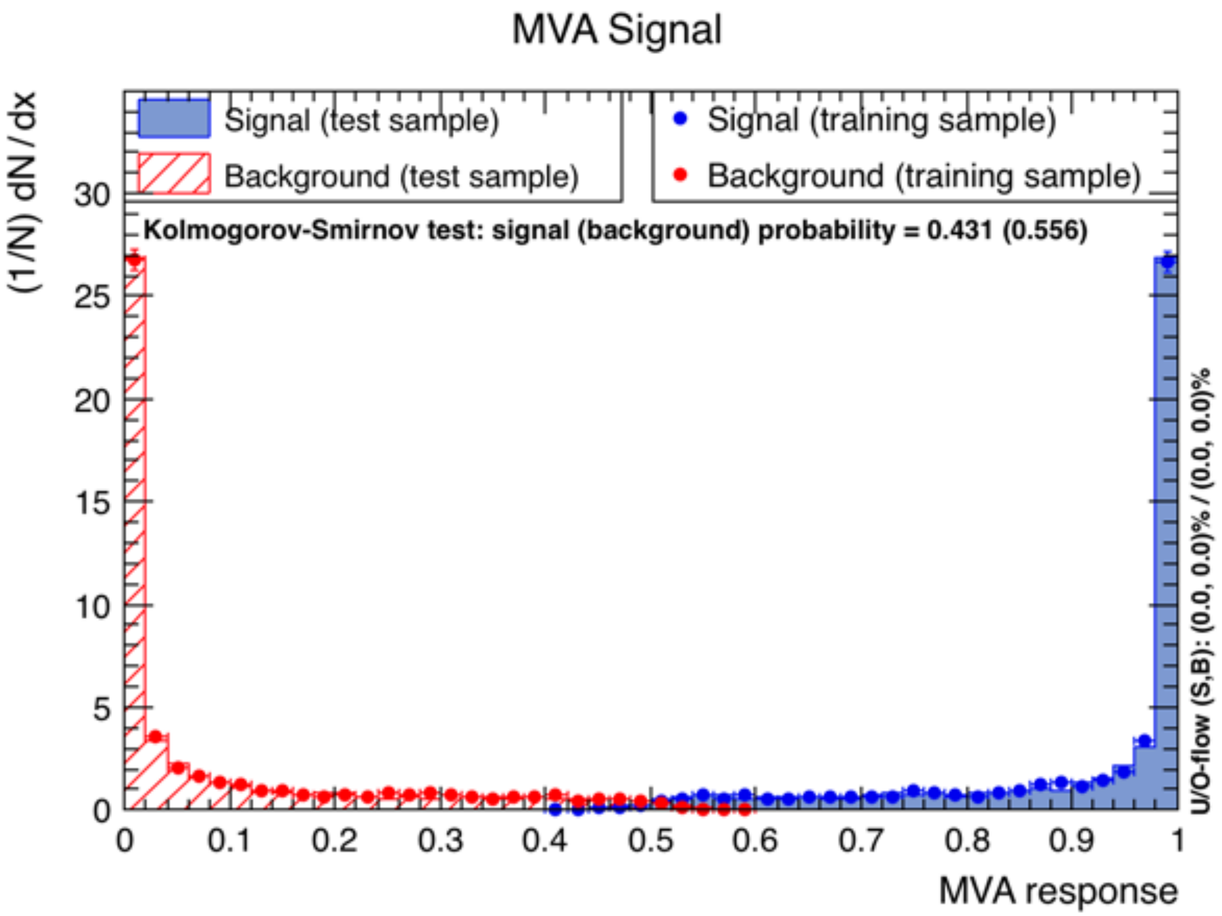
Queen Mary University of London

- Following procedure outline, using macro for TMVA
- 4-fold cross validation on checkerboard - SVM RBF

Hold-out



4-fold Average



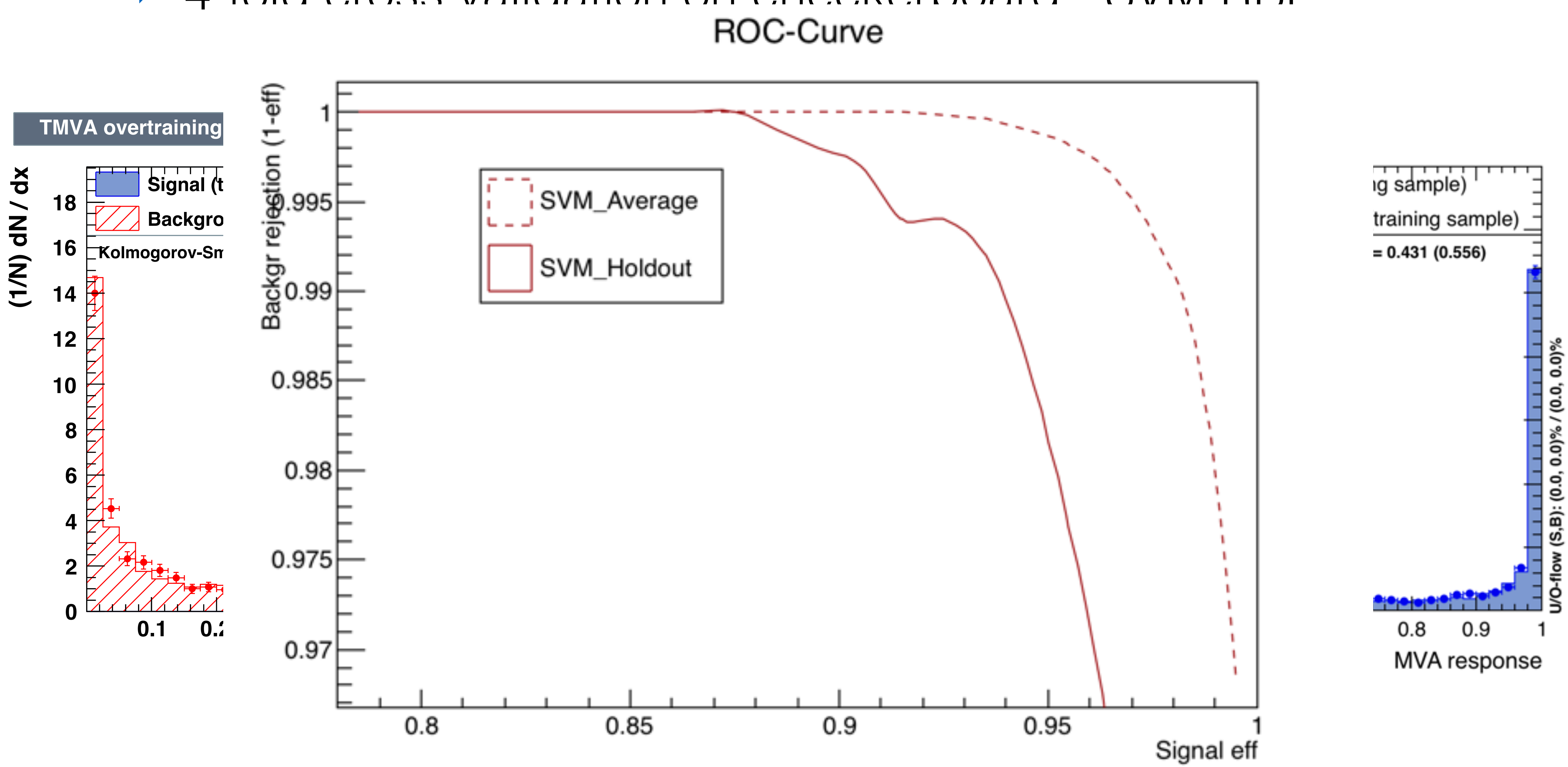
- Further *k*-fold cross validated trainings can be found in backup.

SVMs and Generalisation for HEP

- Generalisation - Checkerboard Example

Queen Mary University of London

- Following procedure outline, using macro for TMVA
- 4-fold cross validation on checkerboard - SVM RBF



- Further *k*-fold cross validated trainings can be found in backup.

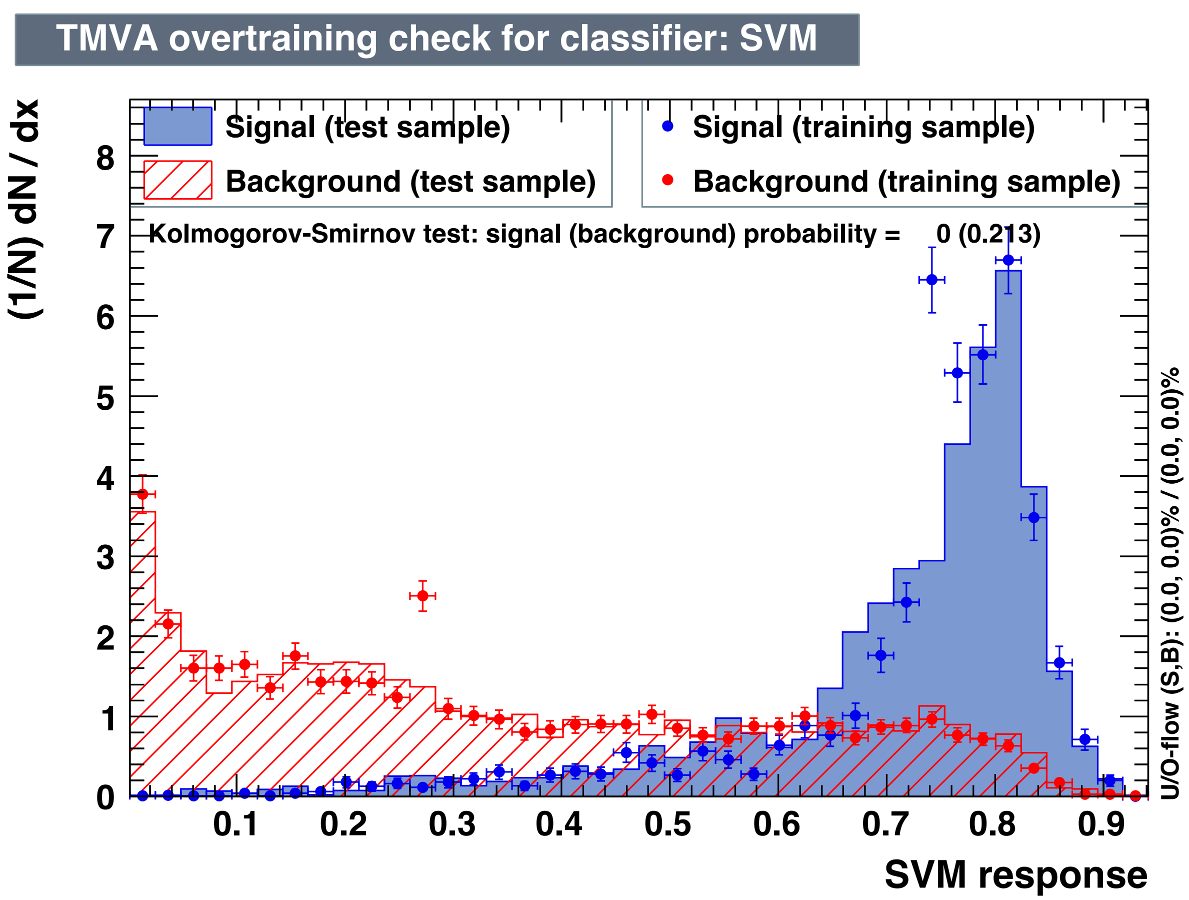
- Support Vector Machines:
  - Modern supervised learning method.
  - Kernel functions set is expanded and soon included in TMVA release.
    - Several minor modifications in mapping input variables to ensure the algorithm more user friendly.

- Generalisation:
  - HEP generally uses hold-out CV.
  - *k*-fold CV used in the wider ML community.
  - A multistage training/validation/testing process have been detailed.
  - Example macro to perform k-fold CV with TMVA soon available in ROOT release.

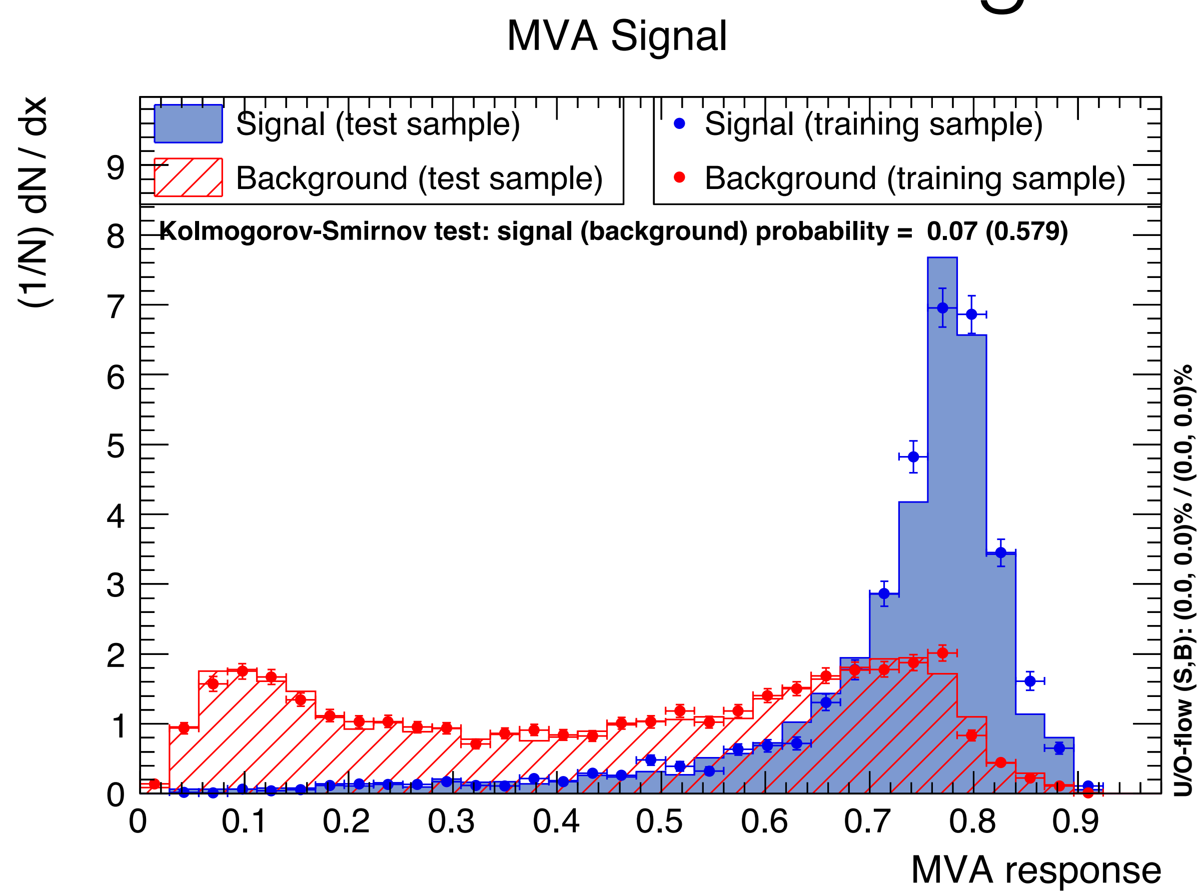
SVMs and Generalisation for HEP

- Ongoing work
  - Integrating *k*-fold CV into TMVA.
  - Investigating real physics examples:
    - $H \to \tau\tau$ Higgs machine learning challenge dataset.
    - Main Physics Analyses.
    - Benchmarking against BDT etc.
- $H \to \tau\tau$ example using first 10 variables:

Hold-out



10-fold CV average



Backup

- SVM - Soft Margin

Queen Mary University of London

- Options table for SVM in TMVA.

Option	Default	Predefined Values	Description
Kernel	RBF	RBF, MultiGauss, Polynomial, Prod, Sum	Choice of kernel function. RBF, Multi-Gaussian and Polynomial are standard kernels. Prod or Sum require **Multi-Kernels** option to be specified with a list of kernels.
MultiKernels	—	—	Option for specifying products or sums of kernels. String with de-limiter */+ for product/sum, e.g. `MultiKernels=RBF*Polynomial`.
Gamma	1	—	RBF kernel parameter. Related to the width, $\Gamma = 1/2\sigma^2$.
Theta	0	—	Polynomial kernel parameter.
Order	1	—	Polynomial kernel parameter.
Gammas	—	—	Multi-Gaussian kernel parameters. Re-quires same number of gammas as in-put variables. Separated by , delimiter e.g. `Gammas=0.8,0.7,0.4` for problem with 3 input variables.
Tune	All	—	Choice of kernel parameters and range to optimise. String separated by com-mas for parameters and range within square brackets separated by semi-colon e.g. `Tune=Gamma[0.1;0.9;8]` will optimise Γ between 0.1 and 0.9 in 8 steps if using Scan to optimise.
C	1	—	Cost parameter.
Tol	0.01	—	Tolerance parameter.
MaxIter	1000	—	Maximum number of training loops.

Queen Mary University of London

- Primal form of problem:
  - Optimise the parameters of the maximal margin hyperplane:

$$\arg\min_{\mathbf{w},b} \frac{1}{2}\langle \mathbf{w} \cdot \mathbf{w} \rangle$$

  - Subject to:

$$y_i \left(\langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b \right) \geq 1$$

- Expressed as a minimisation problem in Lagrangian form as:

$$\arg\min_{\mathbf{w},b} \max_{\alpha \geq 0} \left[\frac{1}{2}\langle \mathbf{w} \cdot \mathbf{w} \rangle - \sum_{i=1}^{n} \alpha_i \left[y_i \left(\langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b \right) - 1 \right] \right]$$

where $\mathbf{w} = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i$ and $b = \frac{1}{N_{SV}} \sum_{i=1}^{n} \left(\langle \mathbf{w} \cdot \mathbf{x}_i \rangle - y_i \right)$

- Soft margin primal Lagrangian

$$\mathcal{L}(\mathbf{w}, b, \xi, \boldsymbol{\alpha}, \mathbf{r}) = \frac{1}{2}\langle \mathbf{w} \cdot \mathbf{w} \rangle + C \sum_{i=1}^{n} \xi_i - \sum_{i=1}^{n} \alpha_i \left[y_i \left(\langle \mathbf{x}_i \cdot \mathbf{w} \rangle + b \right) - 1 + \xi_i \right] - \sum_{i=1}^{n} \mathbf{r}_i \xi_i$$

- Minimisation gives:

$$\frac{\partial \mathcal{L}(\mathbf{w}, b, \xi, \boldsymbol{\alpha}, \mathbf{r})}{\partial \mathbf{w}} = \mathbf{0} \implies \mathbf{w} = \sum_{i=1}^{n} y_i \alpha_i \mathbf{x}_i$$

$$\frac{\partial \mathcal{L}(\mathbf{w}, b, \xi, \boldsymbol{\alpha}, \mathbf{r})}{\partial \xi} = 0 \implies C - \alpha_i - \mathbf{r} = 0$$

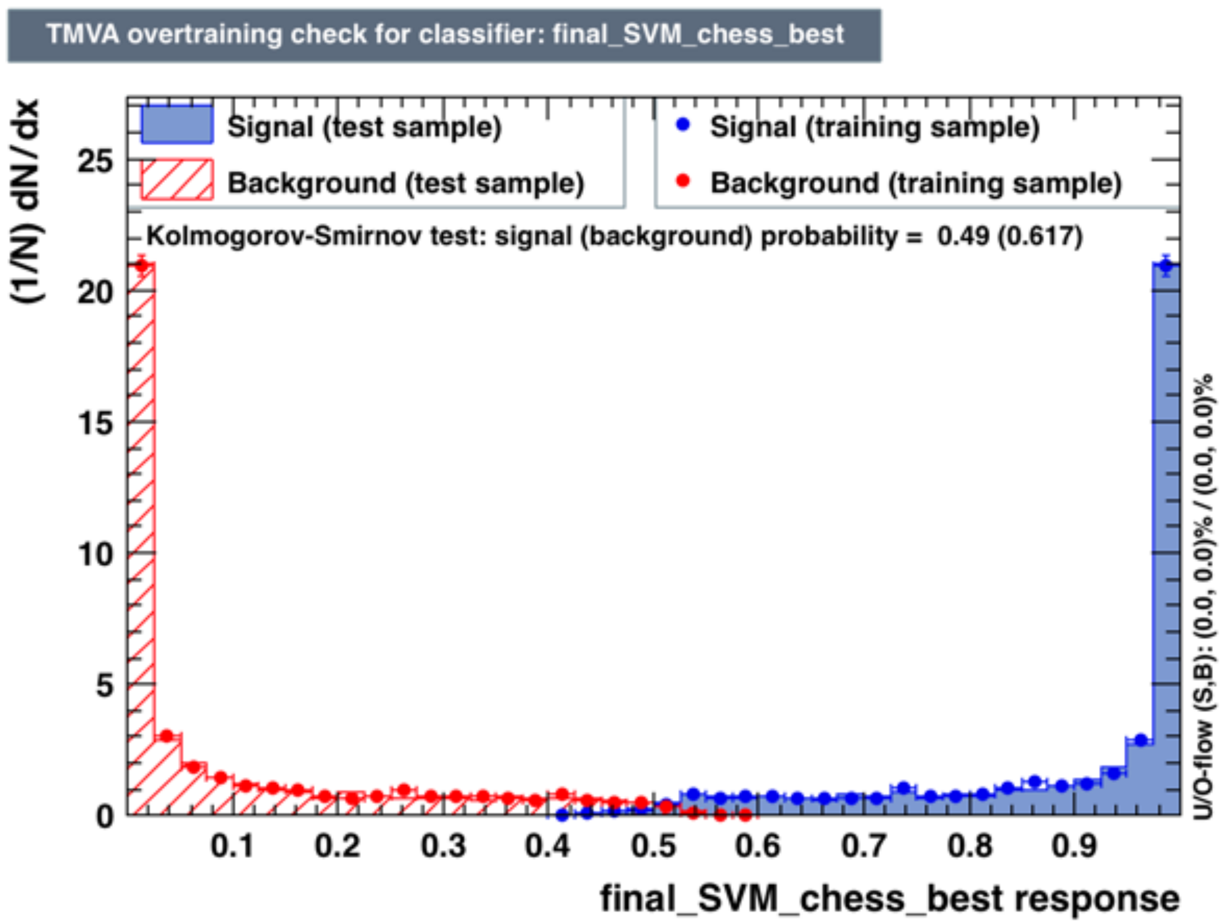
$$\frac{\partial \mathcal{L}(\mathbf{w}, b, \xi, \boldsymbol{\alpha}, \mathbf{r})}{\partial b} = 0 \implies \sum_{i=1}^{n} y_i \alpha_i = 0$$

- Generalisation - Checkerboard Example

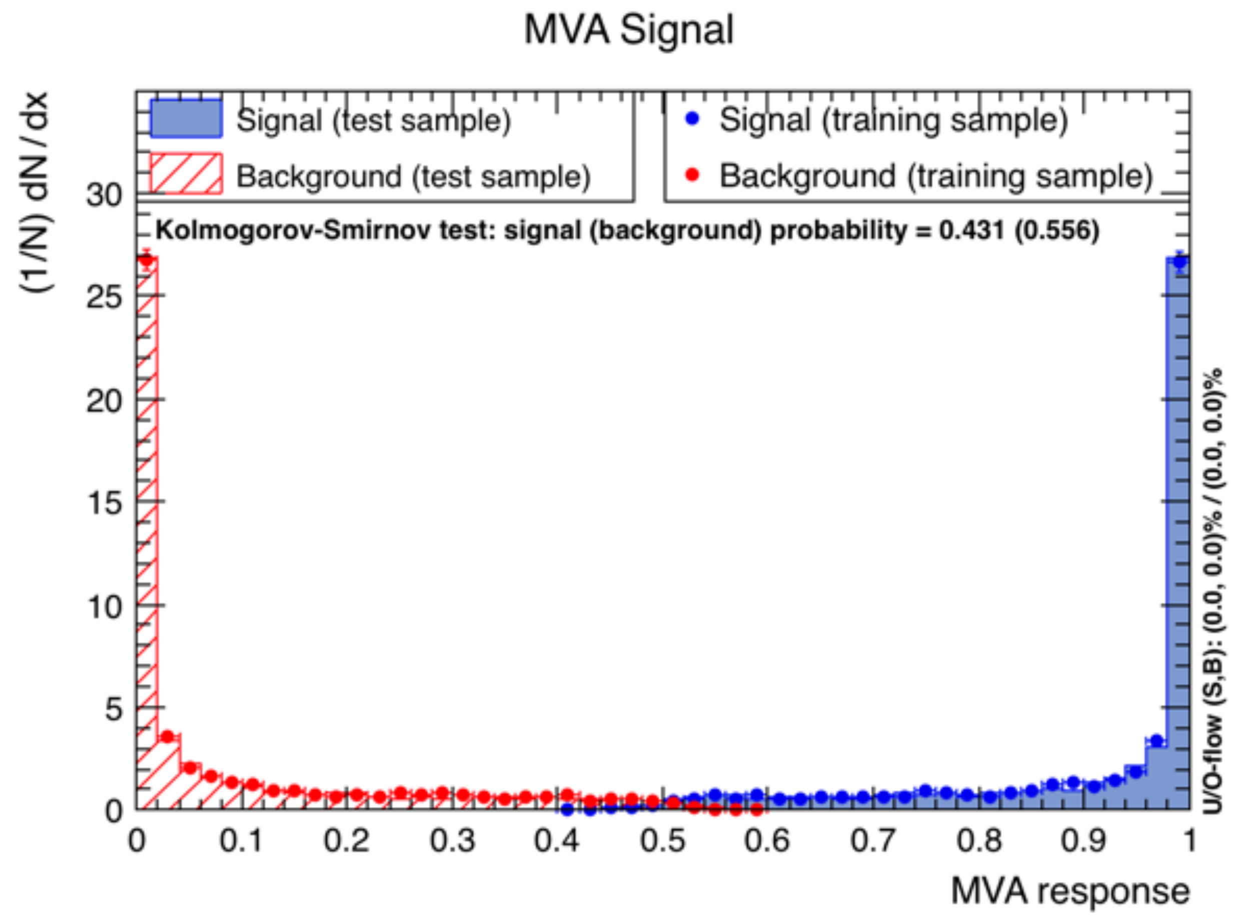
Queen Mary University of London

- Following procedure outline
- 4-fold cross validation on checkerboard - SVM RBF

4-fold Best



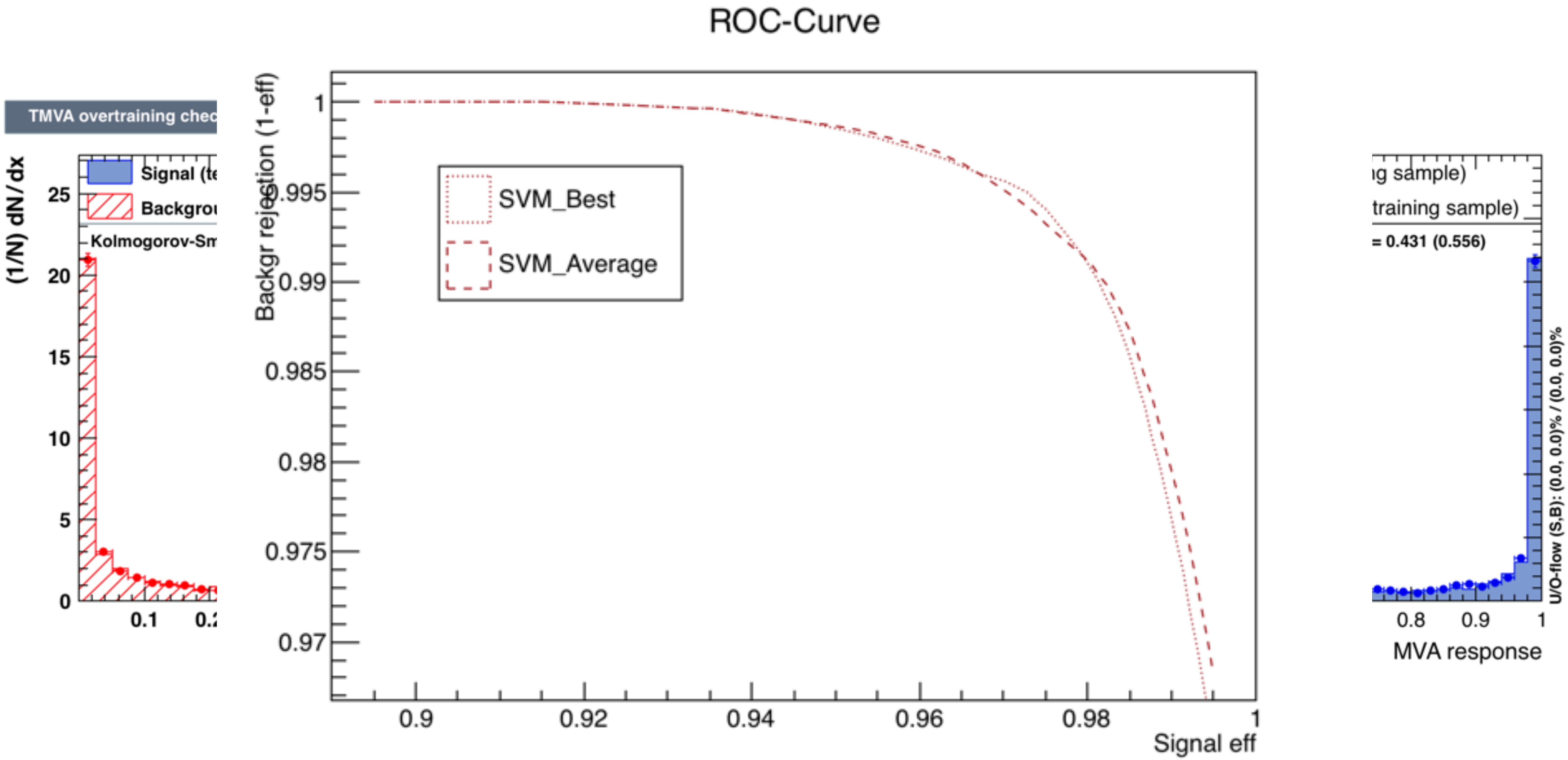
4-fold Average



SVMs and Generalisation for HEP

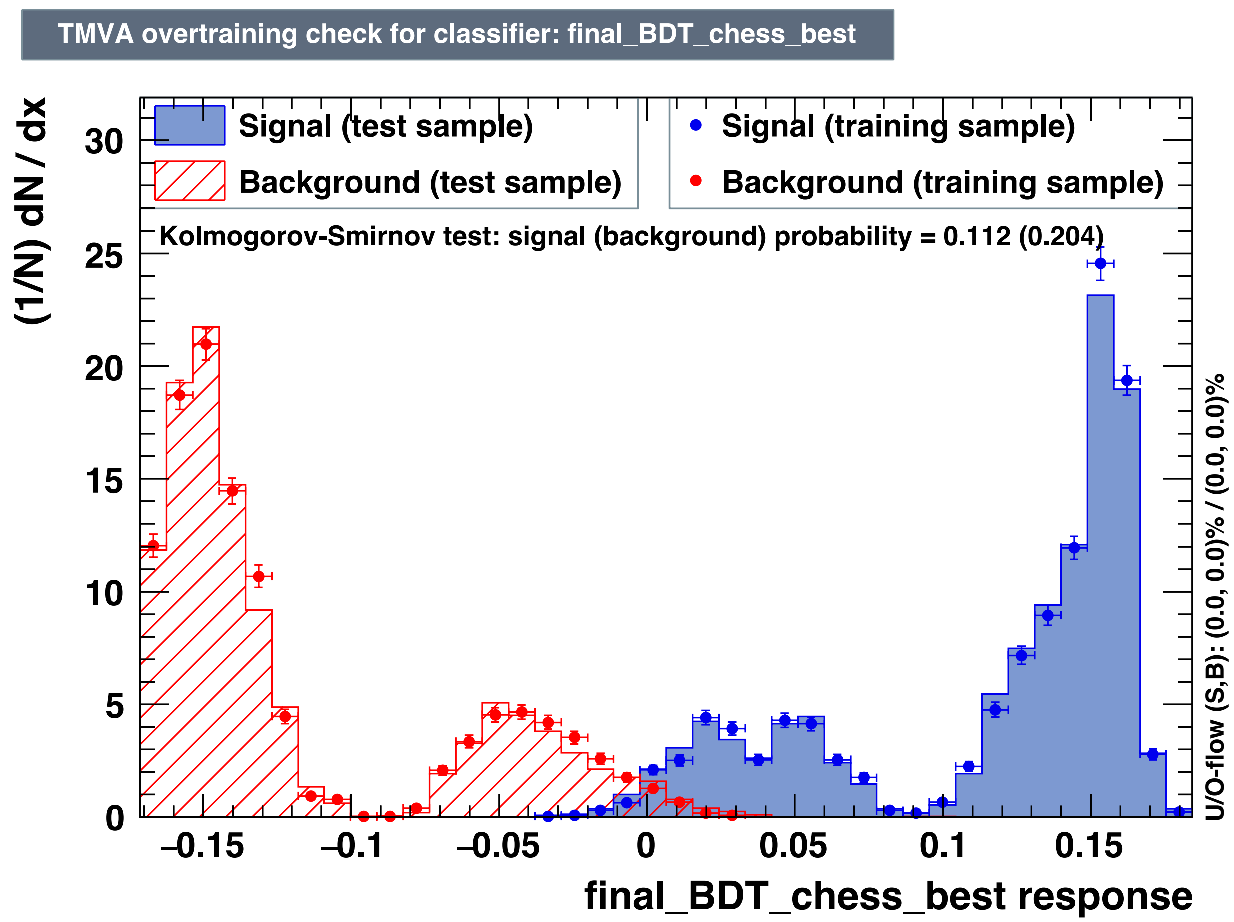
- Generalisation - Checkerboard Example

- Following procedure outline
- 4-fold cross validation on checkerboard - SVM RBF

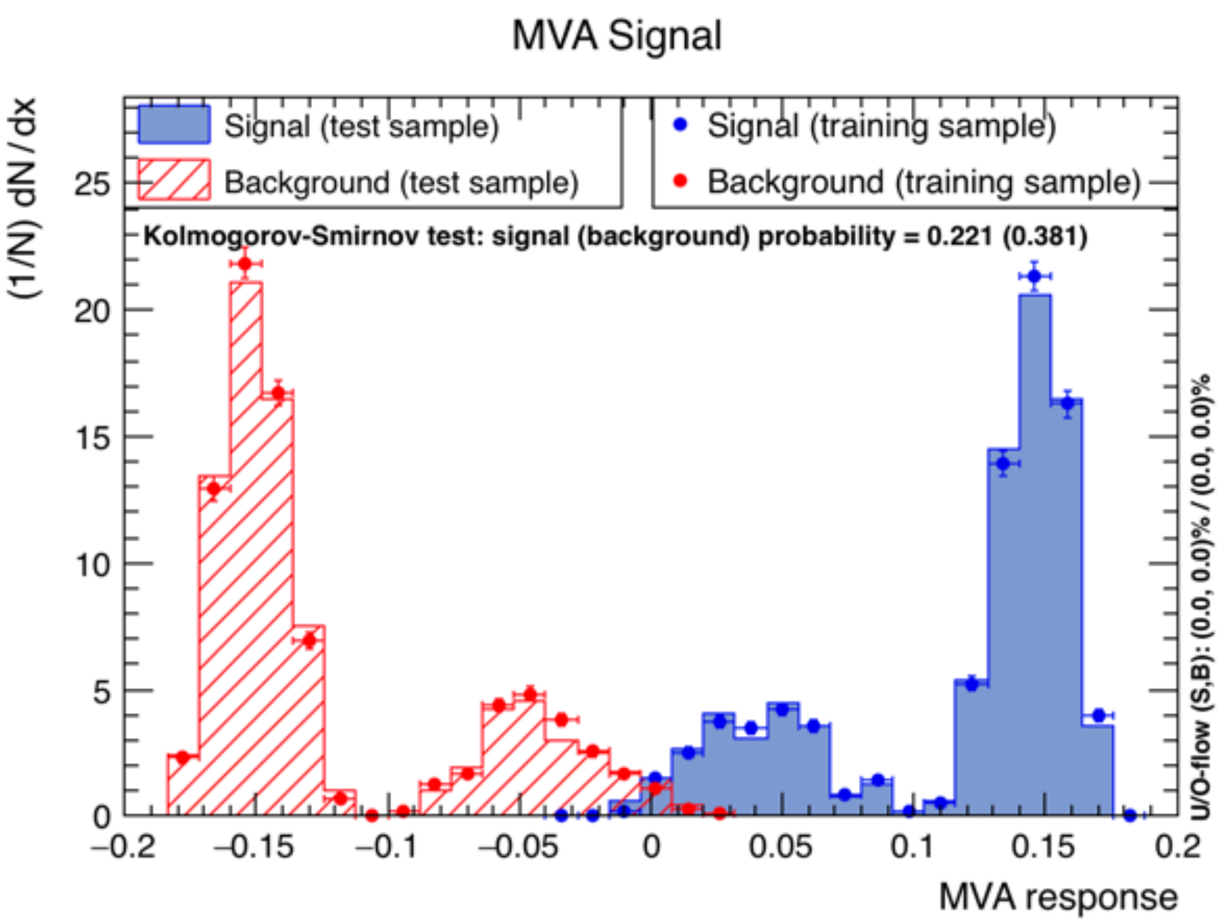


‣ 4-fold cross validation on checkerboard - BDT

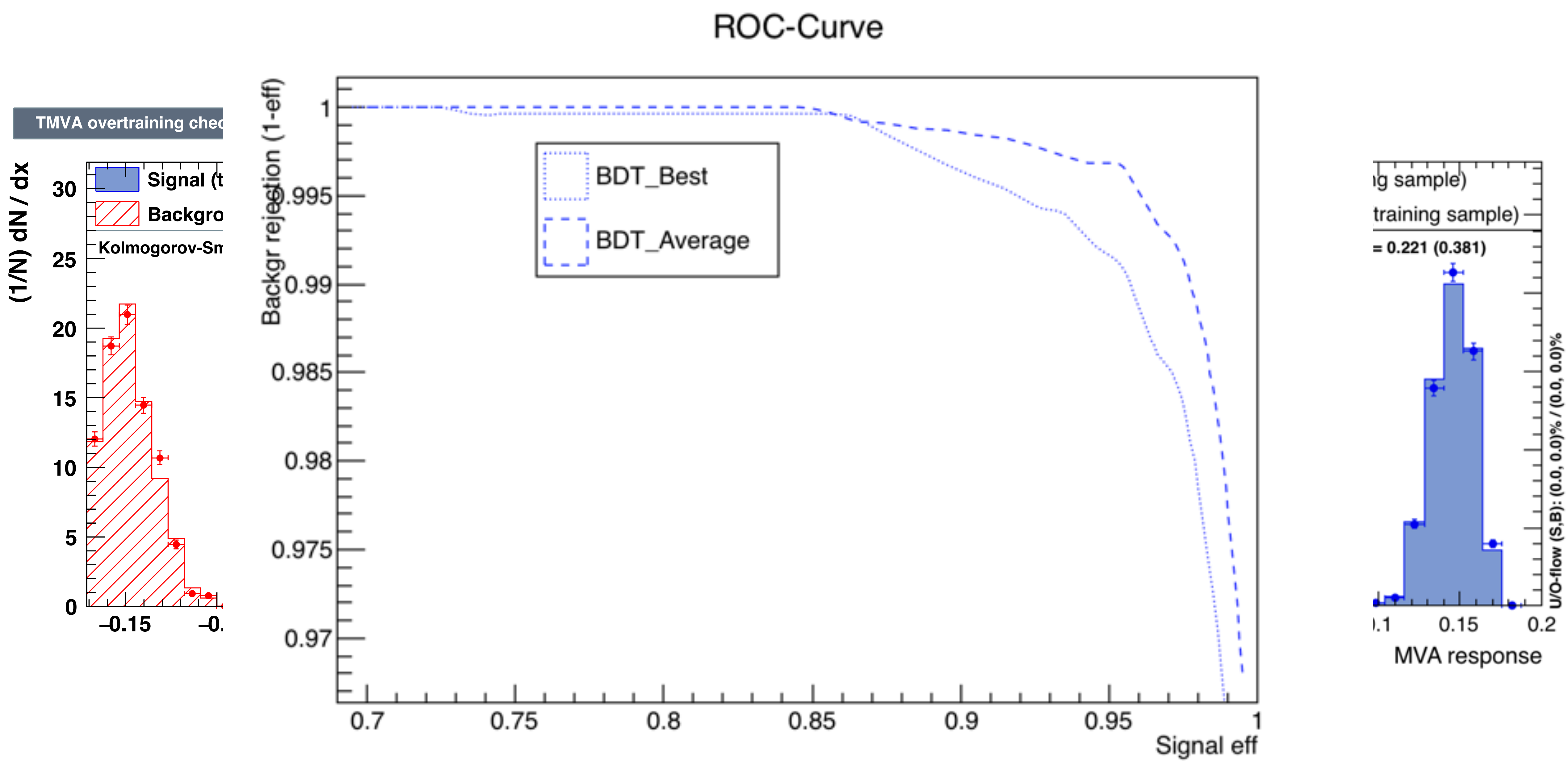
4-fold Best



4-fold Average



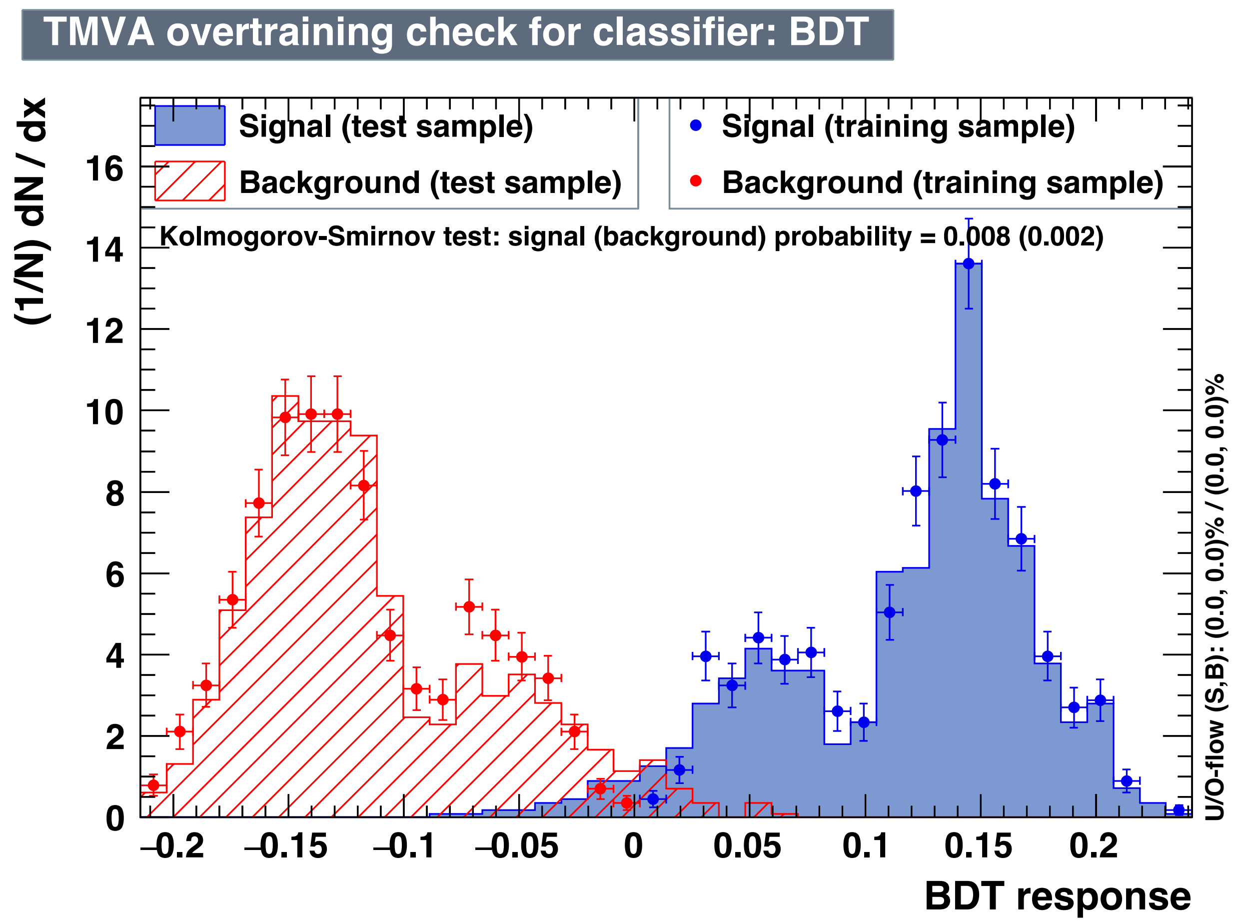
- 4-fold cross validation on checkerboard - BDT



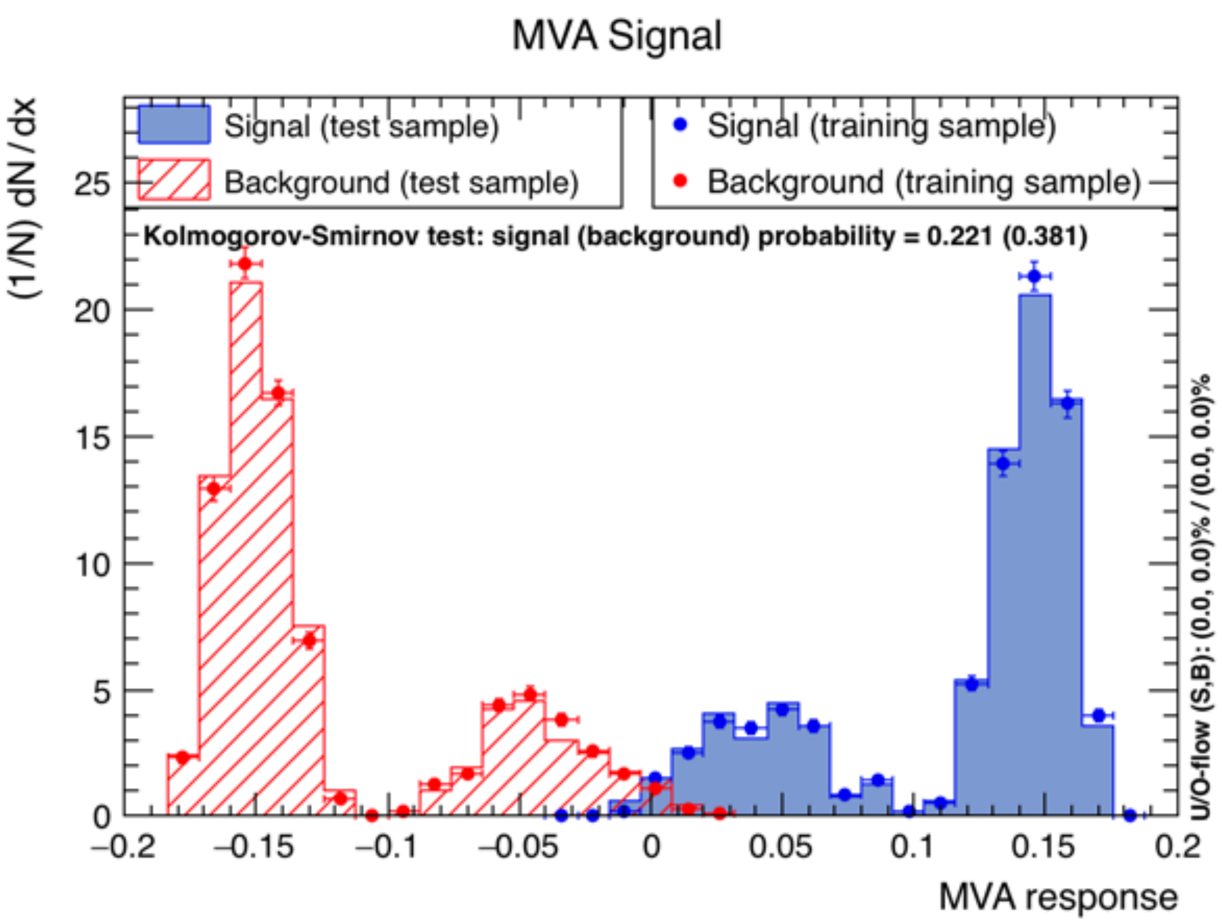
Queen Mary University of London

‣ 4-fold cross validation on checkerboard - BDT

Hold-out

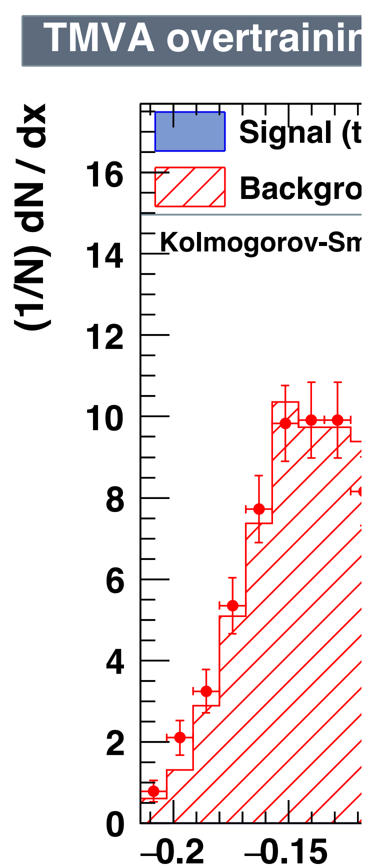


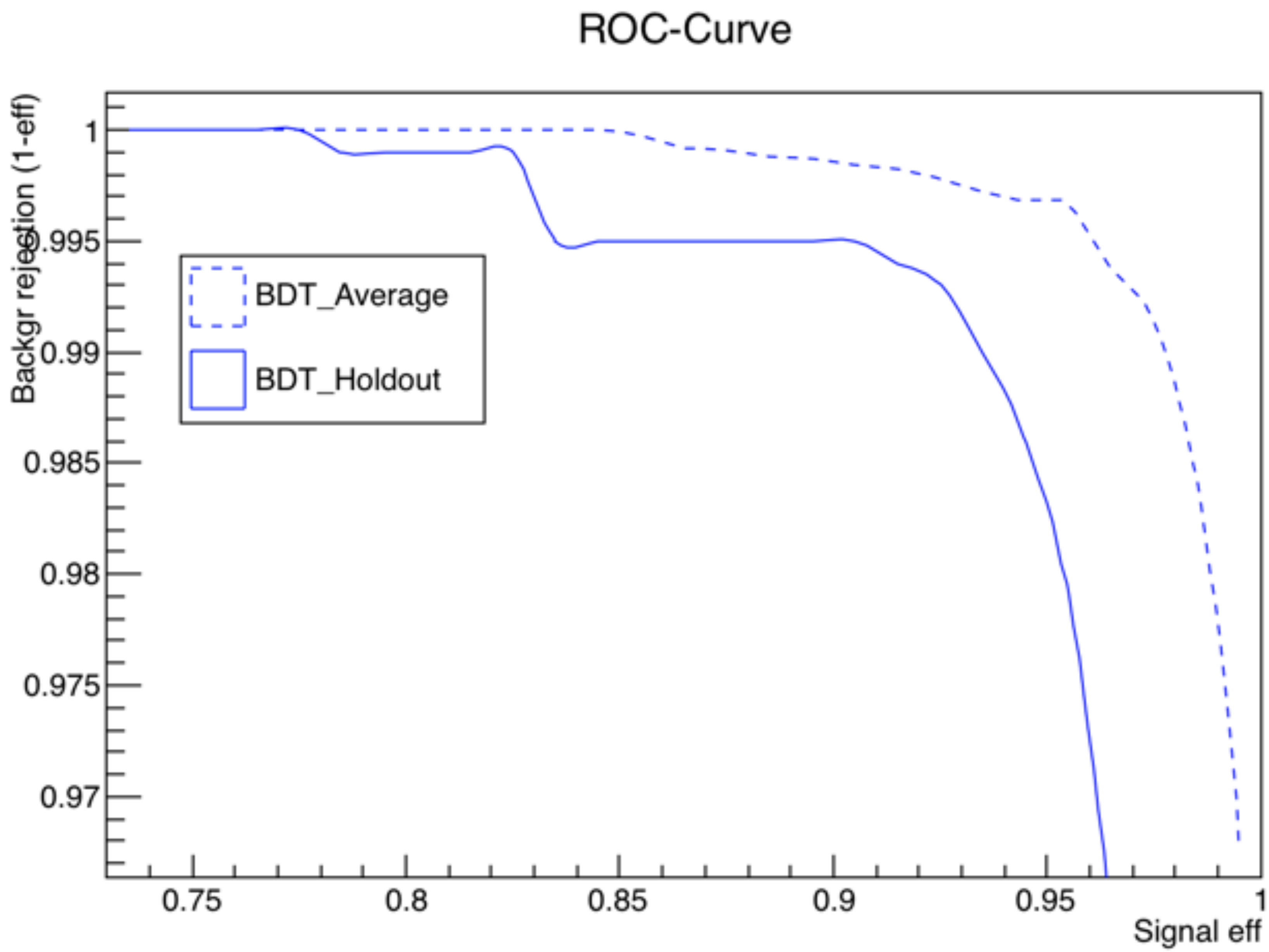
4-fold Average

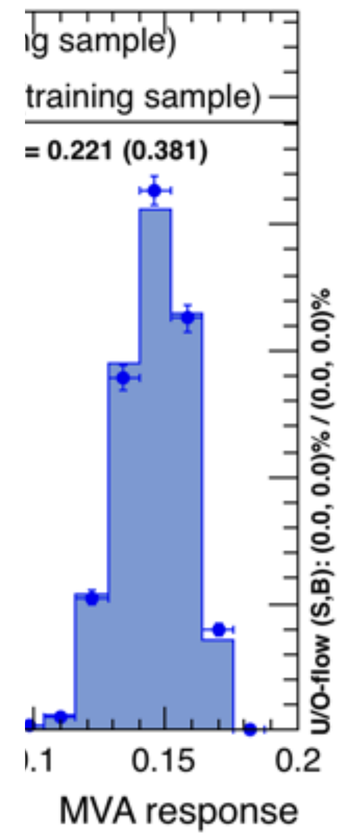


Queen Mary
University of London

‣ 4-fold cross validation on checkerboard - BDT

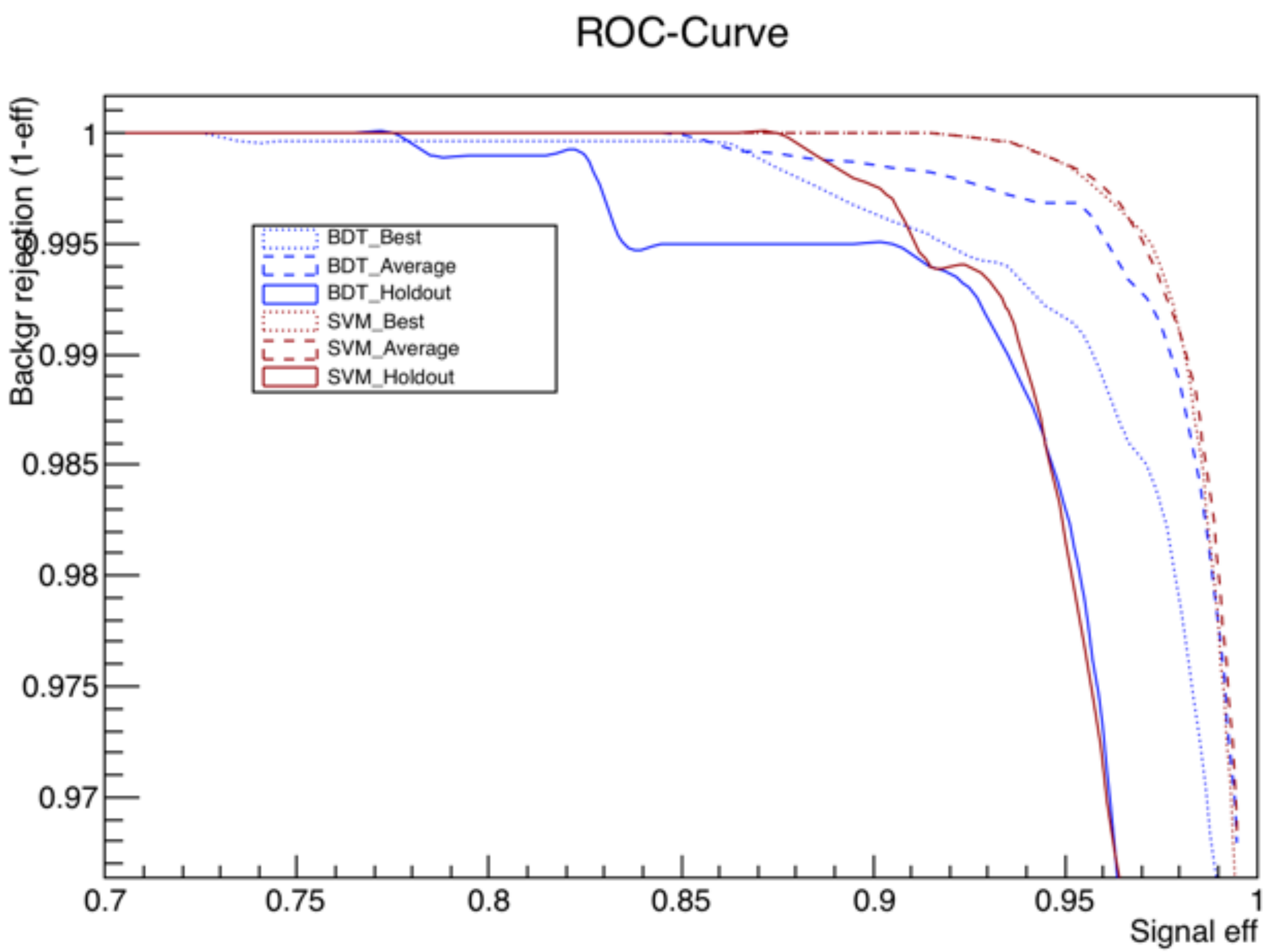






- Generalisation - Checkerboard Example

- ROC curves for all trainings
  - Cross-validated BDTs in backup slides



SVMs anc

- Ongoing v

een Mary
ty of London