PROCEEDINGS OF SCIENCE

Current Developments in TMVA - An Outlook to TMVA4

Jörg Stelzer[*] [†]
DESY, Germany
E-mail: Joerg.Stelzer@cern.ch

Andreas Höcker
CERN, Geneva, Switzerland
E-mail: Andreas.Hoecker@cern.ch

Peter Speckmayer
CERN, Geneva, Switzerland
E-mail: Peter.Speckmayer@cern.ch

Helge Voss
Max-Planck-Institut für Kernphysik, Heidelberg, Germany
E-mail: Helge.Voss@cern.ch

Multivariate classification methods based on machine learning techniques play a fundamental role in today's high-energy physics analyses. In a time of ever larger datasets with an ever smaller signal fraction it becomes increasingly important to use all the features of signal and background that are present. Not only the one dimensional variable projections of the signal must be evaluated against the background, in particular the information present in the variable correlations must be explored. The possible complexities of the data distributed in a high-dimensional space are difficult to disentangle manually and without the help of automata. TMVA is a toolkit which implements a large variety of multivariate classification algorithms. TMVA provides a framework to simultaneously train and evaluate any chosen set of classifiers, and giving means to compare the performance on any dataset for any given problem.

XII Advanced Computing and Analysis Techniques in Physics Research
November 3-7 2008
Erice, Italy

[*] Speaker.
[†] Contributors are listed in arXiv physics/0703039

PoS(ACAT08)063

1. Introduction

The Toolkit for Multivariate Analysis (TMVA) provides a variety of sophisticated multivariate classification techniques, together with a framework for simultaneous classifier training, evaluation, and performance comparison. TMVA is integrated in the ROOT framework [1], making use of its data storing architecture, its mathematical library, and its function evaluation and histogramming capabilities. A large number of classifiers, ranging from simple rectangular cut optimization and projective likelihood estimators, over linear and non-linear multi-dimensional discriminants to more recent developments like boosted decision trees, rule fitting and support vector machines is implemented in TMVA.

An overview about the TMVA package and the implemented classifiers was presented at the 2007 conference for advanced computing and analysis techniques in physics research [2]. There the following classifiers were shown

- rectangular cut optimization,
- projective likelihood estimation,
- multi-dimensional likelihood estimation (PDE range-search and k-NN),
- linear and nonlinear discriminant analysis (H-Matrix, Fisher, FDA),
- artificial neural networks, with three different implementations,
- support vector machine,
- boosted and bagged decision trees, and
- predictive learning via rule ensembles.

A more detailed description of all classifiers is available at our website [4] and also given in the TMVA Users Guide [3]. The users guide includes an introduction into the underlying principle of each classifier, the available tuning options, as well as the advantages and disadvantages with respect to the particular properties of the data samples.

After two years of development of TMVA and acquiring a large user community it became clear that a number of desired features were not possible with the existing framework. It was agreed upon to rewrite large parts of the framework as prerequisite. While maintaining all existing functionality of the TMVA package, the new version of TMVA will contain the following features:

- Data regression
- Multi-class categorization
- Automated classifier tuning and validation using cross-validation methods
- Boosting and bagging as a generic feature of all classifier
- Composite classifiers for parallel but independent training of different phase space regions

- Combined transformation of input data
- Multi threaded minimization and classifier training

In addition, a new likelihood estimator will be available, PDEFoam. As all other PDE classifiers, it provides a multi-dimensional probability density, but uses the novel TFoam method of ROOT for variable space partitioning.

2. Data Regression

Unlike data classification, where events are discretely categorized, data regression denotes the possibility of a classifier to predict the value of a variable. The predicted variable, denoted as the regression target in TMVA, can be a single variable or a vector of variables.

Classification	$\mathscr{R}^N \mapsto \mathscr{R}^{N_x} \mapsto \{1,2,\ldots,N_c\}$
Regression	$\mathscr{R}^N \mapsto \mathscr{R}^{N_r}$

Here N is the dimension of the input variable space, N_c the number of classes, and N_r the dimension of the regression target space. If the classifier is used to distinguish between signal and background (or a number of classes N_c), the result of the mapping $\mapsto \mathscr{R}^{N_x}$ needs to be further used as input into the final classification decision. In case of regression analysis, the output of the mapping $\mapsto \mathscr{R}^{N_r}$ provides directly the prediction for the target variable. In this case, the classifier tries to describe the functional dependence of the target variable on the set of input variables. An often given example in high energy physics is the prediction of the cluster energy correction based on particle type, energy, and detector region.

Similar to the supervised classification training, regression training requires the specification of a training target. This can be a single variable or a list of variables. For most classifiers, regression training happens in the same way as classification training. Event weights can also be applied in order to signify the event importance.

Neural network training is based on minimizing the error function that is the weighted sum of the deviation of the training target from the network output. One-dimensional and multi-dimensional probability density estimators predict the target value of an evaluation event based on the target values of the training events in the vicinity of the evaluation event. In that way they do not describe the signal probability for a given point in the phase space, but contain the most likely target value by averaging over the training sample in that phase space region.

In the current version, regression is implemented for the linear and functional discriminant, the support vector machine, the probability density estimators, and the neural networks. As an example, a two dimensional regression problem is presented here:

$$f_{\text{target}} = ax^2 + by^2$$

Figure 1 shows the performance evaluation plots for a number of classifiers for the given example.

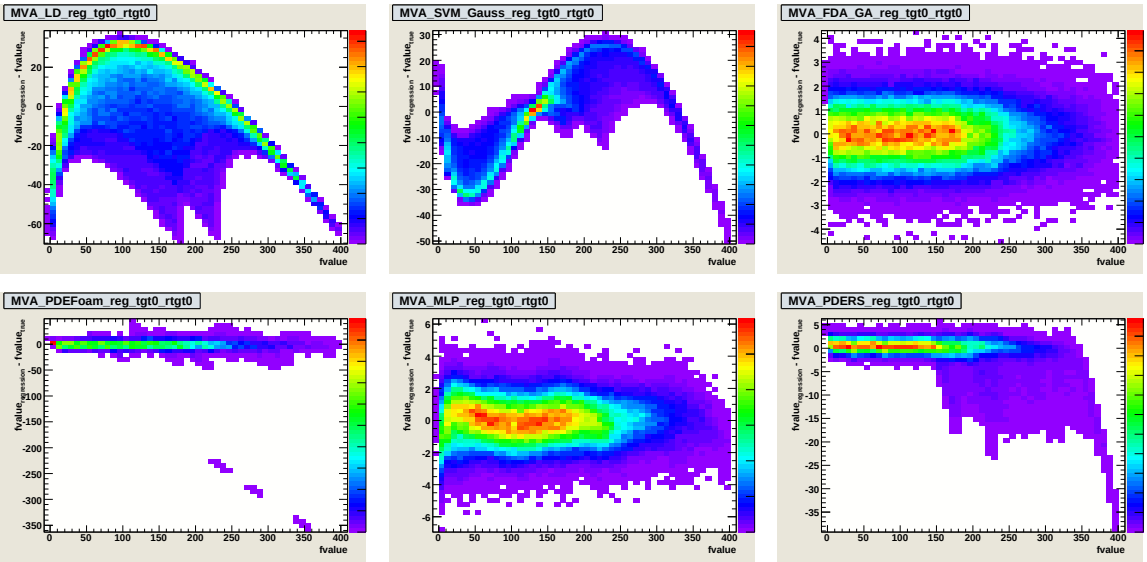

Figure 1: Performance evaluation plots for a number of classifiers on the given example $f_{\text{target}} = 7x^2 + 9y^2$. Shown is the deviation of the prediction from the target versus the target value. The following classifiers are shown: Linear Discriminant, Support Vector Machine with Gaussian Kernel, Functional Discriminant, Probability Density Estimator with Foam partitioning, Multi-layer Perceptron, Probability Density Estimator with range search.

3. Multi Class Categorization

In its current version, TMVA classifies binary datasets, where events are of either *signal* or *background* type. Given the fundamental idea behind TMVA and its design, it is a logical and feasible extension to provide multi-target classification. The framework has been extended such that the user can specify data samples for multiple classes. The classifiers are being adapted to then train on multiple classes and provide multi-class output. Some of the classifiers implemented in TMVA support this naturally, like the neural network and the likelihood based approaches.

4. Automated Classifier Tuning

Many classifiers have parameters which can be tuned in order to improve the classifier's performance. For some, this performance increase can be quite significant. For example, the boosted decision tree classifier is quite robust and in most cases does not require any tuning. Hence it is often the method of choice, although it is not necessarily the best performing or best suitable for a particular problem. One the other hand, the Support Vector Machine is proven to be well performing, but its behavior is quite sensitive to the choice of training parameters. Providing means to determine the best set of training parameters will help to optimize the classifier performance.

An example of classifier tuning is the protection against overtraining. Some classifiers, such as neural networks, boosted decision trees, and probability density estimator methods, potentially suffer from overtraining. Overtrained classifiers are too well adapted to the particular features of the training samples and hence loose there general applicability. The performance deteriorates, unless protective action is taken and the corresponding training parameters are properly adjusted.

For the boosted decision tree these parameters are the split criteria and pruning strength, for the density estimators, it is the search range or the partition size, and for the neural network, a special overtraining protection parameter can be set.

The basic principle of automated classifier tuning is the repeated execution of the training and evaluation cycle on the training data. The test data remains unused at this stage, since it is necessary for the independent performance validation of the final classifier. Since classifier training and evaluation should never be done on the same data, the training data sample is partitioned into subsamples. Out of these, each is retained once as evaluation sample, while training is performed on the other subsamples. This principle, called cross-validation or rotation estimation, has several implementations. For TMVA we implemented the K-fold cross-validation, which works as follows.

The training data sample is divided into K subsamples T_i, $(i = 1,\ldots,K)$. For a given set of training parameters α, the training is performed on $K-1$ subsamples T_i, $(i = 1,\ldots,k-1,k+1,\ldots,K)$, retaining the k-th subsample T_k for validation. This is repeated K times, $(k = 1,\ldots,K)$. For each of the K resulting classifiers C_k, the error E_k, $(k = 1,\ldots,K)$, is calculated on the test subsample T_k:

$$E_k(\alpha) = \mathbf{E}\left(C_k(\cdot|\alpha)|T_k\right) = \frac{1}{N_k}\sum_{j=1}^{N_k} w_j^k L(x_j^k, C_k(x_j^k|\alpha)),$$

where N_k is the size of the test sample T_k, x_j^k and w_j^k are the j-th event and its weight in that sample, and L is the loss function. The loss function is a measure for the deviation of the prediction from the truth and it presents the cost or the regret that is associated with the decision for a given event. Its exact form depends on the problem that needs to be solved. The goal of automated tuning is to find that set of training parameters for which the average error

$$\bar{E}(\alpha) = \frac{1}{K}\sum_{k=1}^{K} E_k(\alpha)$$

is minimal. For this the space of classifier tuning parameters is scanned. With the optimal set of classifier parameters, α_{opt}, - for these $\bar{E}(\alpha_{\text{opt}})$ is smallest - the classifier is retrained on the complete training data. The untouched test data is then used for validation and performance evaluation.

5. Generic Boosting

A decision tree is a series of splitting cuts that divide a data sample into ever smaller sets. After the splitting, leafs are assigned either signal or background status. Decision trees are easy to understand, but not very powerful classifiers. The performance can improved when, instead of using a single decision tree, a whole group of decision trees is combined. This group is built up by adding new trees based on the performance of the already existing ones. Each new tree is created from a data sample that emphasizes the classification mistakes made by previous trees. This method is deployed by the boosted decision tree classifier.

In general, the *boosting* method produces a series of discriminators, which, combined, results in a more powerful classifier. This principle, instead of being applied only to decision trees, can be applied as a meta-algorithm around other learning algorithms as well. The advantages are

- an improved performance in phase space regions where classes overlap

- classifier that are desensitized to the particularities of the training data, and an increase robustness in case of small training samples.

A number of variants of the boosting principle exist, of which TMVA implements the method of AdaBoost [5]. AdaBoost trains a classifier F rounds, each time re-weighting events to increase the importance of those that were wrongly classified in the previous round. Finally, one can then either use the last classifier retained in the series, or combine all F classifiers, as it is done for TMVA's boosted decision tree classifier

$$y(x) = \sum_{i=1}^{F} \log\left(\frac{1-f^i_{\text{err}}}{f^i_{\text{err}}}\right) C^i(x), \quad \text{with} \quad f^i_{\text{err}} = \frac{N}{N_{\text{wrong}}} - 1\,.$$

With the new framework TMVA provides the possibility to boost any classifier via AdaBoost, and compare the performance with the not boosted version. A comparative check of the boosted decision tree classifier and a decision tree classifier trained with the boosting framework has been done. First tests have been performed with the boosting of the neural net classifier. Likelihood based classifiers and discriminators like Fisher can not be boosted. For likelihood estimators re-weighting only distorts the probability density distribution.

6. Composite Classifiers

It is quite often desired to simultaneously train different classifiers on different phase space regions of the same data sample, or to sequentially train different classifiers on the combined output of the previous step(s). A few examples from high energy physics are:

- Training of a neural network, *e.g.* for muon identification, differently in different regions of the detector. A detector region would be identified by rapidity η and angle ϕ. These variable can of course not be used for the training itself, since an imbalance in the signal and background distribution in the training sample would cause a bias in the network prediction. The regions need to be trained separately, the composite classifier is designed to help with the setup of this.

- A multi-class problem learned by a neural network can show an improvement, if a probability density estimator is applied on the output. This also provides a likelihood that can be used as input to further treatment.

- Multi-particle event classification. It is conceivable to have a two step decision making process. In a first step, classification input can be based on one or more particles, for instance to identify leptons, Z, or Higgs boson candidates. In a second step, an event decision is formed based on the output of the first step and on additional event data. In this way, particle identification and searches for particle decay chains can be combined.

It should be noted that all these points can already be achieved, but need to be set up manually. Composite classifiers are primarily designed for convenience. They are also less prone to user errors. Inside the composite classifier, the training will still happen step-wise. It is not planned to train a set of chained classifiers in a single process.

7. Further Developments

With the PDEFoam classifier, a new probability density estimator has been introduced into TMVA. It unites the advantages of a multi-dimensional likelihood, as stated in the Neyman-Pearson-Lemma [6], with the speed advantages that a clever partitioning algorithm provides. The Foam partitioning algorithm [7] combines training data events that lie in the same phase space region and that show only a small variance in class type or regression target into a single cell. This cell is described by the class or target mean, the average event position, and the total (weighted) number of contained events. The variance is a measure of the uniformity of the cell content. When calculating the local probability density, extensive lookup and calculations are avoided, since the representative means of the contributing cells are used. TMVA uses the Foam partitioning algorithm that is implemented in the ROOT framework. A detailed description of the mechanism and first performance studies are also presented in these conference proceedings [8].

For the efforts in physics analysis that are intensified at the dawn of LHC data taking, large samples of data will be available for study using TMVA. The new features of TMVA, such as automated classifier tuning and generic boosting, will only add to the amount of needed computing power. In these days, most desktop machines are equipped with multi-core CPU's. Efforts in TMVA are underway to provide multi-thread support for a number of the most extensive computing tasks, such as the Monte Carlo sampling and genetic algorithm minimization methods, the building of decision trees, and the training of neural networks and support vector machines.

8. Conclusion

Over the years, TMVA has required a large user base. Over this time several new features have been thought of and suggested by the developer and user community. With the last version of TMVA 3 being stable and released with ROOT, an effort was started to redesign the framework and implement a number of new features and concepts. Other improvements were also implemented. Just to name a few, the Broyden-Fletcher-Goldfarb-Shanno method was added as MLP learning algorithm alternative to back-propagation, the support vector machine and the *k*-nearest neighbor density estimator were equipped with Gaussian kernels, and, for better release compatibility, xml was chosen as the format for the classifier weight files. It is planned to release the new TMVA 4 version before summer 2009, with the new framework, regression, multi-class decision, generic boosting, and PDEFoam fully functional.

Acknowledgements

The fast growth of TMVA would not have been possible without the contribution and feedback from many developers mentioned as co-authors of the users guide [3]) as well as René Brun and the ROOT team. We thank in particular the CERN Summer students Matt Jachowski (Stanford U.) for the implementation of TMVA's MLP neural network, Yair Mahalalel (Tel Aviv U.) for a significant improvement of PDERS, and Alexander Voigt (Dresden University) for the implementation of the PDEFoam classifier. The Support Vector Machine has been contributed to TMVA by Andrzej Zemla and Marcin Wolter (IFJ PAN Krakow), and the k-NN classifier has been written by Rustem Ospanov (Texas U.).

References

[1] R. Brun and F. Rademakers, "*ROOT - An Object Oriented Data Analysis Framework*", Nucl. Inst. Meth. in Phys. Res. A 389, 81 (1997).

[2] H. Voss, "*TMVA, the Toolkit for Multivariate Data Analysis with ROOT*", in proceedings of "XI International Workshop on Advanced Computing and Analysis Techniques in Physics Research" PoS(ACAT)040 (2007)

[3] A.Höcker, P. Speckmayer, J. Stelzer, F. Tegenfeldt, H. Voss, K. Voss, A. Christov, S. Henrot-Versille, M. Jachowski, A. Krasznahorkay Jr., Y. Mahalalel, R. Ospanov, X. Prudent, M. Wolter, A. Zemla, "*TMVA - Toolkit for Multivariate Data Analysis*", arXiv:physics/0703039v4

[4] http://tmva.sourceforge.net

[5] Y. Freund and R.E. Schapire,"*Experiments with a new boosting algorithm*", Machine Learning: Proceedings of the Thirteenth International Conference, pages 148-156, 1996.

[6] The best criteria to distinguish two hyptheses $H_0 : \theta = \theta_0$ and $H_1 : \theta = \theta_1$ ist the likelihood ratio test, which rejects H_0 in favor of H_1, when $L(\theta_0|x)/L(\theta_1|x) = \Lambda(x) \leq \eta$, where $\mathscr{P}(\Lambda(x) \leq \eta|H_0) = \alpha$, is the most powerfull test of Type-I error α for threshold η.

[7] S. Jadach, "*Foam: A general purpose cellular Monte Carlo event generator*", Comput. Phys. Commun. **152**, 55 (2003) [arXiv:physics/0203033].

[8] D. Dannheim, "*PDE-FOAM - a probability-density estimation method based on self-adapting phase-space binning*", in proceedings of "XII International Workshop on Advanced Computing and Analysis Techniques in Physics Research", PoS(ACAT08)064 (2008)