Annales UMCS Informatica AI 6 (2007) 149-160

Annales UMCS
Informatica
Lublin-Polonia
Sectio AI

http://www.annales.umcs.lublin.pl/

Implementation of the SVM algorithm for high energy physics data analysis

Marcin Wolter[1*], Andrzej Zemła[1,2]

[1]*Institute of Nuclear Physics PAN, Radzikowskiego 152, 31-342 Kraków, Poland*
[2]*Jagiellonian University, Reymonta 4, 30-059 Kraków, Poland*

Abstract

Elementary particle physics experiments, searching for very rare processes require the efficient analysis and selection algorithms able to separate signal from the overwhelming background. In the last ten years a number of powerful kernel-based learning machines, like Support Vector Machines (SVM), have been developed. SVM approach to signal and background separation is based on building a separating hyperplane defined by the support vectors. The margin between them and the hyperplane is maximized. The extensions to a non-linear separation are performed by mapping the input vectors into a high dimensional space, in which data can be linearly separated. The use of kernel functions allows us to perform computations in a high dimension feature space without explicitly knowing a mapping function.

We have implemented an SVM algorithm and integrated it with the CERN ROOT package, which is currently a standard analysis tool used by elementary particle physicists. We also used the implemented SVM package to identify hadronic decays of τ leptons in the ATLAS experiment at LHC accelerator. The performance of the method is compared to the likelihood estimator, which does not take into account correlations between variables. The use of SVM significantly reduces the number of background events.

1. Introduction

In elementary particle physics the efficient analysis of a huge amount of collected data requires the use of sophisticated selection and analysis algorithms. Scientists taking part in all of the future LHC (Large Hadron Collider proton accelerator at CERN) experiments [1] are searching for new phenomena in physics (Higgs boson, new symmetries, additional dimensions), which are very rare processes, with a probability going down to even 10^{-13}. The efficient search for such rare phenomena requires multivariate analysis tools capable of high level background suppression.

[*]Corresponding author: *e-mail address*: Marcin.Wolter@ifj.edu.pl

The reason for application of statistical learning multivariate methods is, in most cases, simply the lack of knowledge about the mathematical dependence of the quantity of interest on the relevant measured variables. Either there is no mathematical model at all and an exhaustive search is the only possibility to find the correct dependence, or the known models are insufficient and statistical learning provides a better description of data. These methods require a set of training data, which is typically supplied by the Monte Carlo simulation.

2. Support Vector Machine

In the early 1960s the linear support vector method was developed to construct separating hyperplanes for pattern recognition problems [2,3]. It took 30 years until the method was generalized for constructing nonlinear separating functions [4,5] and for estimating real-valued functions (regression) [6]. At that moment it became a general purpose algorithm performing data classification and regression which can compete with neural networks. Typical applications of SVMs include text categorization, character recognition, bioinformatics and face detection.

The main idea of the SVM approach is to build a separating hyperplane which maximizes the margin. The position of the hyperplane is defined by the subset of all training vectors called support vectors. The extension into non-linear SVM is performed by mapping input vectors into a high dimensional feature space in which data can be separated by a linear procedure using the optimal separating hyperplane. The use of the kernel functions eliminates the explicit transformation to the feature space and simplifies the computations.

2.1. Linear Support Vector Machine

A detailed description of SVM formalism can be found for example in [7], here only a brief introduction is given. Consider a simple two-class classifier with oriented hyperplanes. If the training data is linearly separable, then such a set of $(\vec{w},b)$ pairs can be found that the following constraints are satisfied:

$$\forall_i y_i \left(\vec{x}_i \cdot \vec{w} + b\right) - 1 \geq 0 , \tag{1}$$

where x_i are the input vectors, y_i the desired outputs (y_i=±1) and $(\vec{w},b)$ define a hyperplane. The decision function of the classifier is $f(\vec{x}_i) = sign(\vec{x}_i \cdot \vec{w} + b)$, which is +1 for all points on one side of the hyperplane and –1 for the points on the other side. Intuitively, the classifier with the largest margin will give better generalization. The margin for this linear classifier is just $2/|\vec{w}|$. Hence, in order to maximize the margin, one needs to minimize the cost function W:

$$W = (1/2)|\vec{w}|^2 , \tag{2}$$

with the constraints from Eqn. 1. At this point it would be beneficial to consider the significance of different input vectors x_i. The training data points lying on the margins, which are called the support vectors (SV), are the data that contribute to defining the decision boundary (see Fig. 1). If the other data are removed and the classifier is retrained on the remaining data, the training will result in the same decision boundary. To solve this constrained quadratic optimization problem, we first reformulate it in terms of a Lagrangian:

$$L(\vec{w},b,\vec{\alpha}) = 1/2|\vec{w}|^2 - \sum_i \alpha_i \left(y_i \left((\vec{x}_i \cdot \vec{w}) + b \right) - 1 \right), \tag{3}$$

where $\alpha_i \geq 0$ and the condition from Eqn. 1 must be fulfilled.

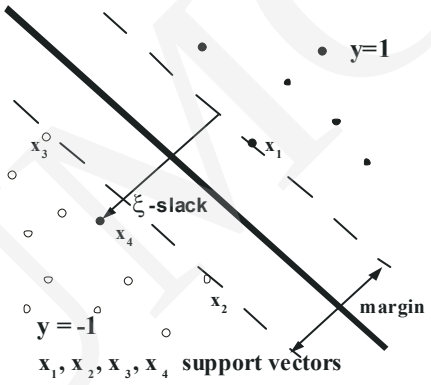

Fig. 1. Hyperplane classifier in two dimensions. Points x_1, x_2 and x_3 define the margin, i.e. they are the support vectors

Lagrangian L should be minimized with respect to $|\vec{w}|$ and b and maximized with respect to $\vec{\alpha}$. The solution has an expansion in terms of a subset of input vectors for which $\alpha_i \neq 0$ (these are the support vectors):

$$\vec{w} = \sum_i \alpha_i y_i \vec{x}_i \,, \tag{3}$$

since at extremum $\partial L / \partial b = 0$ and $\partial L / \partial \vec{w} = 0$. The optimization problem becomes the one of finding the $\vec{\alpha}$ which maximizes:

$$L(\vec{\alpha}) = \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \left(\vec{x}_i \cdot \vec{x}_j \right). \tag{4}$$

2.2. Non-separable data

The above algorithm can be extended to non-separable data. The correct classification constraints in Eqn. 1 are modified by adding a slack variable ξ_i to it ($\xi_i = 0$ if the vector is properly classified, otherwise ξ_i is a distance to the decision hyperplane).

$$\forall_i y_i\left(\vec{x}\cdot\vec{w}+b\right)-1+\xi_i\geq 0 \qquad \xi_i\geq 0\,. \tag{6}$$

This allows some points to be misclassified. The training algorithm needs to minimize the cost function, i.e. a trade-off between the maximum margin and the classification error:

$$W=\left(1/2\right)\left|\vec{w}\right|^2+C\sum_i\xi_i\,. \tag{7}$$

The selection of C parameter defines how much a misclassification increases the cost function.

2.3. Nonlinear Support Vector Machine

The formulation of SVM presented above can be further extended to build a nonlinear SVM, which can classify nonlinearly separable data. Consider a function Φ which maps the training data from $\Re^n$ to some higher dimensional space $\Re^n$. In this high dimensional space, the data can be linearly separable, hence the linear SVM formulation above can be applied to these data (see Fig. 2).

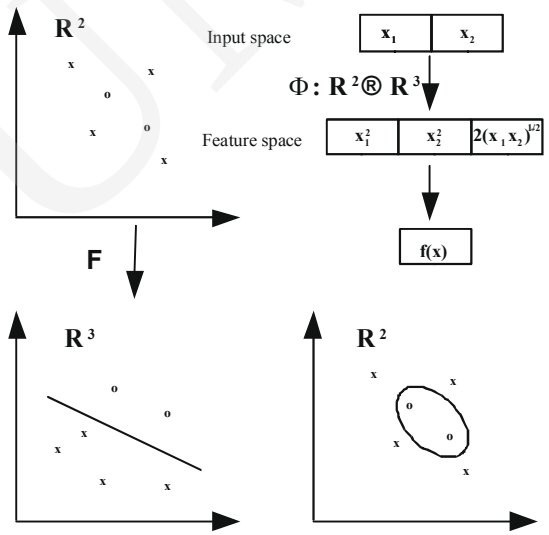

Fig. 2. Example of data separable by an elliptic curve in $\Re^2$, but linearly separable in the feature space $\Re^3$. The mapping to transforms $(x_1,x_2)\rightarrow(x_1^2,x_2^2,\sqrt{2}x_1x_2)$.

In the SVM formulation data appear only in the form of dot products $(\vec{x}_i\cdot\vec{x}_j)$ (see Eqn. 5). The dot product $\Phi(\vec{x}_i)\cdot\Phi(\vec{x}_j)$ appears in the high dimension feature space. It can be replaced by a kernel function:

$$K\left(\vec{x}_i,\vec{x}_j\right)=\Phi\left(\vec{x}_i\right)\cdot\Phi\left(\vec{x}_j\right). \tag{8}$$

By computing the dot product directly using a kernel function, one avoids the mapping $\Phi(\vec{x})$. This is desirable, because $\Phi(\vec{x})$ can be tricky or impossible to compute. Using a kernel function, one does not need to know explicitly what the mapping is. The most frequently used kernel functions are:

$$\begin{aligned} K(\vec{x}_i,\vec{x}_j) &= \left((\vec{x}_i \cdot \vec{x}_j) = c\right)^d && \textit{polynominal} \\ K(\vec{x}_i,\vec{x}_j) &= \tanh\left(\kappa(\vec{x}_i \cdot \vec{x}_j) + \theta\right) && \textit{sigmoid} \\ K(\vec{x}_i,\vec{x}_j) &= \exp\left(\frac{-1}{2\sigma^2}\left\|\vec{x}_i - \vec{x}_j\right\|^2\right) && \textit{Gaussian} \end{aligned} \tag{9}$$

A question arises whether there is any constraint on the type of kernel function suitable for this task. It was shown that a function must fulfill the Mercer's condition to form a suitable kernel:

$$\int K(\vec{x},\vec{y})g(\vec{x})g(\vec{y})d\vec{x}d\vec{y} \geq 0 \tag{10}$$

for any function *g* such that $\int g(\vec{x})^2 d\vec{x}$ is finite.

To extend the methodology described for a linear case to nonlinear problems, one substitutes $\vec{x}_i \cdot \vec{x}_j$ for $K(\vec{x}_i,\vec{x}_j)$ in Eqn. 5. Due to Mercer's conditions on the kernel, the corresponding optimization problem is a well defined convex quadratic programming problem, which assures us that there exists a global minimum. This is an advantage of SVMs compared to neural networks, which may find one of the local minima.

2.4. Regression SVM

A version of a SVM which can perform regression (called SVR) was proposed in 1997 [8]. The model produced by support vector classification (as described above) depends only on a subset of training data, because the cost function for building the model does not care about training points that lie beyond the margin and are properly classified. Analogously, the model produced by SVR depends only on a subset of the training data, because the cost function for building the model ignores any training data that is close (within a threshold) to the model prediction:

$$\begin{aligned} \left|\vec{y} - f(\vec{x})\right|_\varepsilon &:= \max\left\{0, \left|\vec{y} - f(\vec{x}) - \varepsilon\right|\right\} \\ W &= (1/2)\left|\vec{w}\right|^2 + C/m\sum_{i=1}^{m}\left|\vec{y}_i - f(\vec{x}_i)\right|_\varepsilon. \end{aligned} \tag{11}$$

The ε-insensitive cost function is more robust to small changes in data and in model and also less sensitive to outliers when compared to least squares cost function used by neural networks.

2.5. Comparison of SVM and Neural Networks

For Support Vector Machines, in contrast to neural networks, the capacity is independent of dimensionality of the data. The algorithm can get bounds on the error, is statistically well motivated, and it uses the theory of structural risk minimization (theory which characterizes generalization abilities of learning machines). Finding the weights is a quadratic programming problem guaranteed to find a minimum of the error surface. Thus the algorithm is efficient and SVMs generate almost optimal classification and obtain good generalization performance due to a high dimensionality of the feature space. It has also very few free parameters. In the case of classification these are: the type of the kernel function, kernel parameters (frequently one parameter only, like in the case of Gaussian kernel) and the cost parameter C. Therefore it is generally possible to perform a grid search to identify an optimal set of parameters. By contrast, in the case of neural network the entire network architecture has to be optimized.

On the other hand, the training of SVM is definitely slower due to computationally intensive solution of minimization problem, especially for large amounts of training data. SVM generates complex solutions (frequently more than 60% of training points are used as support vectors) especially for large amounts of training data.

3. Packages ROOT and TMVA

In the last years the ROOT framework [9] has become the standard analysis tool used by particle physicists all over the world. The system, developed at CERN laboratory, provides a set of object oriented frameworks with all the functionality needed to handle and analyze large amounts of data in a very efficient way. Having the data defined as a set of objects, specialized storage methods are used to get direct access to the separate attributes of the selected objects, without having to touch the bulk of the data. Included are histogramming methods in 1, 2 and 3 dimensions, curve fitting, function evaluation, minimization, graphics and visualization classes to allow an easy setup of an analysis system that can query and process the data interactively or in batch mode.

ROOT is an open system that can be dynamically extended by linking external libraries. This makes ROOT a premier platform on which data acquisition, simulation and data analysis systems are built.

The Toolkit for Multivariate Analysis (TMVA) [10] provides a ROOT-integrated environment for the parallel processing and evaluation of MVA (i.e. machine-learning) techniques to discriminate signal from background. At present it includes:

- Rectangular cut optimization
- Likelihood estimator (PDE approach)
- Multi-dimensional likelihood estimator (PDE - range-search approach)
- H-Matrix (chi-squared) estimator
- Fisher discriminant
- Artificial Neural Network
- Boosted/Bagged Decision Trees
- RuleFit

The TMVA package includes implementations for each of these discrimination techniques, their training and testing (performance evaluation). In addition, all these methods can be tested in parallel, and hence their performance on a particular data set may be easily compared.The training, testing and evaluation phases are performed in parallel for various methods. The evaluation accommodates several numerical performance estimators as well as various plots, like the efficiency vs. background rejection curves, correlation matrices etc.

4. Implementation of SVM in the ROOT/TMVA framework

We have implemented the SVM algorithm in the ROOT/TMVA framework. This implementation uses a Sequential Minimal Optimization (SMO) [11] to solve the quadratic problem. Further modifications proposed by Keerthi [12] speed up the algorithm. To speed up the minimization most of the algorithms divide a set of vectors into smaller subsets. The SMO method puts the subset selection to the extreme by selecting subsets of two vectors.

Let us give a brief description of the SMO algorithm, the details can be found in [11] and [12]. The pairs of vectors are chosen, using heuristic rules, to make the largest possible minimization step. Because the working set is of the size of two it is straightforward to write down the analytic solution. The minimization procedure is repeated recursively until the minimum is found. The SMO algorithm has proven to be significantly faster than the other methods like chunking [13] or SVMlight [14], and has become the most common minimization method used in the SVM implementations.

The implemented SVM algorithm performs the classification tasks using the linear or Gaussian kernel function, the other kernels will be added in the next release. The Gaussian kernel allows us to apply discriminant shape in the input space. An exemplary problem is shown in Fig. 3, where a set of vectors forming a ring is chosen as a signal against the flat background. The SVM algorithm finds a set of support vectors on the borders of the ring and selects the signal points out of the background.

5. Application to the identification of τ particle in the ATLAS experiment

The SVM algorithm implemented within the TMVA frame has been used for identification of τ leptons in the ATLAS experiment, one of the four great experimental setups constructed at the LHC accelerator at CERN. Identification of hadronic τ decays will be the key to the possible Higgs boson and supersymmetry discovery in the ATLAS experiment [15] and it has been studied for several years [16].

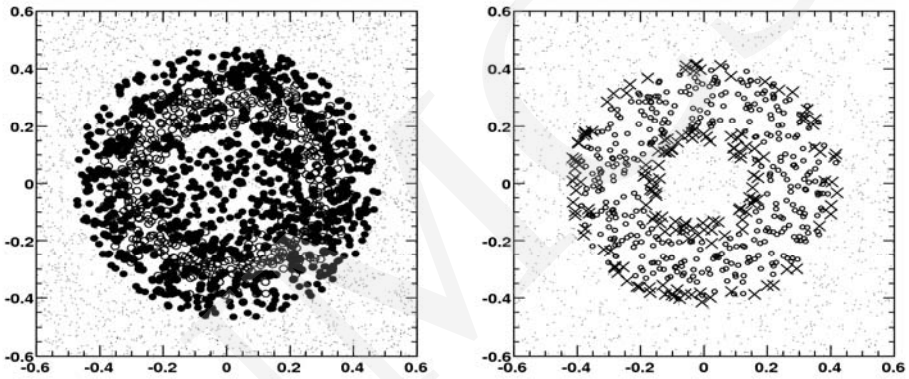

Fig. 3. A set of signal points (empty circles) forming a ring and background points (black dots) are used to train an SVM. The filled black circles represent the support vectors (left plot). In the right plot the empty circles and black points represent the correctly classified signal and background vectors, the crosses stay for wrongly classified vectors

Lepton τ decays predominantly into a small number of charged and neutral pions, that form a well collimated jet characterized by one (1-prong jet) or three (3-prong jet) tracks in the inner part of the detector. The discriminating variables are not independent and no single variable provides a really good signal and background separation (see Fig. 4) [17,18]. Signal efficiency is defined as a ratio of accepted and all signal events $\epsilon_s = N_{accepted}/N_{all}$ and background rejection as a ratio of rejected and all background events $R = 1 - \epsilon_b = N_{accepted}/N_{all}$.

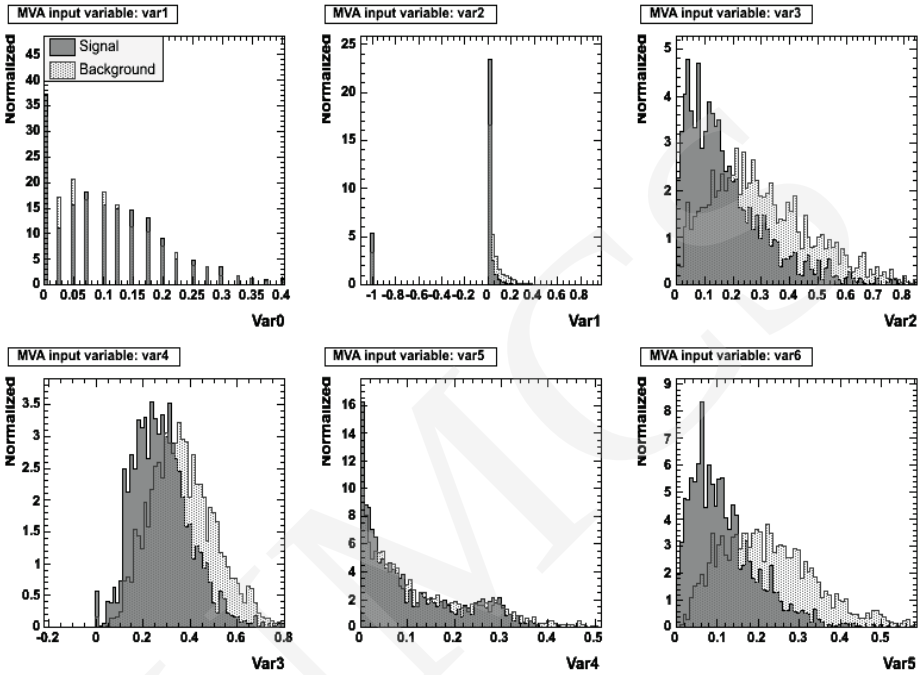

Fig. 4. Distributions of discriminating variables for 1-prong tau candidates. None of the variables provides a good signal and background discrimination

The SVM algorithm tested on simulated ATLAS events and used to identify τ leptons using 1-prong data. The Gaussian kernel function was chosen. The data was divided into two subsamples: one for training and one for verification and calculating the background rejection. The performance of the SVM with radial kernel depends on two parameters: the width of the Gaussian kernel and the cost parameter *C*. A grid search in the space of these two parameters was performed to maximize the background rejection for 80% signal detection efficiency.

The results for both training and verification samples are the same (Fig. 5), which ensures that there is no overtraining. The performance of the SVM is compared to the performance of the likelihood estimator, which does not take into account the correlations between variables (Fig. 4). A significant background reduction is observed: the background rejection for 80% signal efficiency increases from 70% to 75% therefore the use of SVM reduces the number of background events by about 17%.

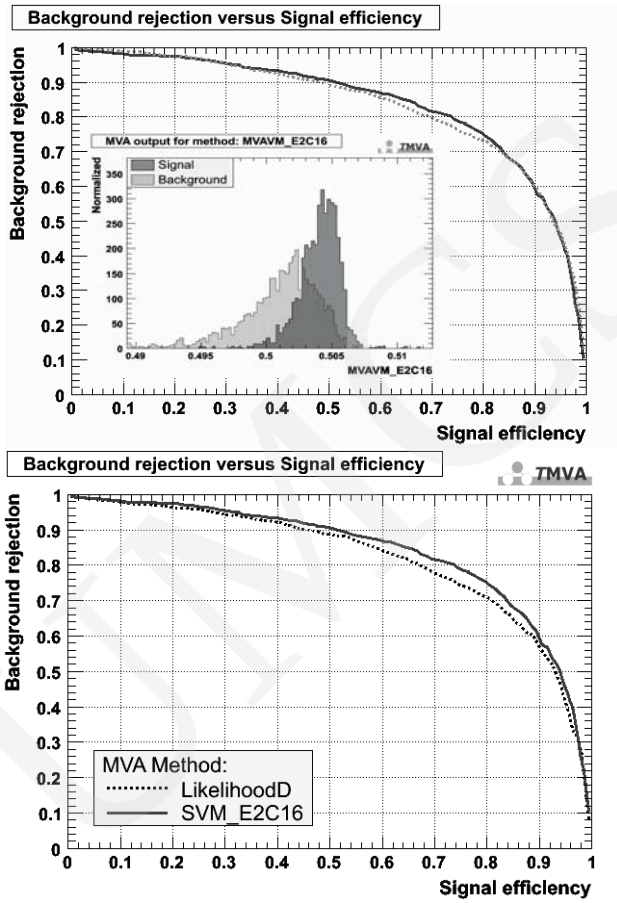

Fig. 5. The background rejection as a function of signal efficiency for the training (dashed line) and test (solid line) for the simulated ATLAS τ data while using SVM (left plot). Also the distributions of the discriminants for signal and background are shown. The SVM method (solid line) is compared with the likelihood estimator (dotted line) (right plot)

6. Conclusions and plans

The further extension of the presented SVM implementation is foreseen. We plan to add the automatic optimization procedure designed to find the optimal kernel parameters for a given dataset. Also additional kernel functions will be implemented (sigmoid, polynomial) and the package will become more functional by adding the Least Square SVM (LSSVM). LSSVM replaces the linear slack variable with a sum of squares. In this case the solution follows from solving a set of linear equations, instead of quadratic programming for classical SVM's [19].

The other planned extension is an implementation of the *ν*-SVM (*ν*-Soft Margin Support Vector Classifiers) [20]. In this algorithm an extra ν parameter allows us to control the number of support vectors. It enables us to eliminate the regularization constant *C* in the classification case and the accuracy parameter *ε* in the regression case.

We have shown that Support Vector Machine can be successfully used to analyze high energy physics data. The implementation described above will be included in the ROOT package, therefore it will be easily available to the entire particle physics community. This implementation extends the range of multivariate analysis tools available within the ROOT framework.

Acknowledgments

The work was supported by the European Reintegration Grant MERG-CT-2005-030760 and by the Polish Government grant PBS 132/CER/2006/03.

References

[1] *Large Hadron Collider. The Experimental Programme*. Report CERN (1993)
ALICE Technical Proposal, http://consult.cern.ch/alice/documents/1995/01/abstract
ATLAS Technical Proposal. Report CERN/LHCC/94-43.1994,
http://atlas.web.cern.ch/atlas/tp/tp.html
CMS Technical Proposal. Report CERN/LHC 94-38. 1994;
LHCb technical proposal, http://lhcb-tp.web.cern.ch/lhcb-tp/.

[2] Vapnik V., Chervonenkis A., *A note on one class of perceptrons*. Automation and Remote Control, 25 (1964).

[3] Vapnik V. and Lerner A. *Pattern recognition using generalized portrait method*. Automation and Remote Control, 24 (1963).

[4] Boser B., Guyon I., Vapnik V., *A training algorithm for optimal margin classifiers*. Proceedings of the 5th Annual ACM Workshop on Computational Leraning Theory, ACM Press, (1992) 144.

[5] Cortes C., Vapnik V., *Support vector networks*. Machine Learning, 20 (1995) 273.

[6] Vapnik V., *The Nature of Statistical Learning Theory*. Springer Verlag, (1995).

[7] Burges C., *A tutorial on support vector machines for pattern recognition*. Data Mining and Knowledge Discovery, 2(2) (1998) 1.

[8] Vapnik V., Golowich S., Smola A., *Support vector method for function approximation, regression estimation, and signal processing*. In M. Mozer, M. Jordan, and T. Petsche, editors, Advances in Neural Information Processing Systems 9, Cambridge, MA, MIT Press, (1997) 281, 287.

[9] Brun R., Rademakers F., *ROOT - An Object Oriented Data Analysis Framework*. Proceedings AIHENP'96 Workshop, Lausanne, Sep. 1996, Nucl. Inst. & Meth. in Phys. Res. A 389 (1997) 81.
See also http://root.cern.ch/

[10] Höcker A., Voss H., Voss K., Stelzer J., *TMVA (Toolkit for MultiVariate Analysis)*. http://tmva.sourceforge.net

[11] Platt J., *Fast training of support vector machines using sequential minimal optimization*. In B. Scholkopf, C. Burges & A. Smola, eds, Advances in Kernel Methods-Support Vector Learning. MIT Press, (1999).

[12] Keerthi S., Shevade S., Bhattacharyya C., Murthy K., *Improvements to Platt's SMO algorithm for SVM classifier design*. Tech Report, Dept. of CSA, Banglore, India, (1999).
[13] Vapnik V., *Estimation of Dependences Based on Empirical Data*. Springer-Verlag, (1982).
[14] Joachims T., *Making large-scale support vector machine learning practical*. In B. Scholkopf, C. Burges, A. Smola. Advances in Kernel Methods: Support Vector Machines, MIT Press, Cambridge, MA, December (1998).
[15] ATLAS Collaboration. *Atlas performance and physics technical design report*. ATLAS TDR 15, CERN/LHCC/99-15, May (1999).
[16] Heldmann M., Cavalli D., *An improved tau-identification for the atlas experiment*. ATLAS Physics Note ATL-PHYS-PUB-2006-008, December (2005).
[17] Richter-Was E., Szymocha T., *Hadronic τ indentification with track based approach: the $z\rightarrow\tau\tau$,$w\rightarrow\tau\nu$ and di-jet events from dc1 samples*. ATLAS Note ATL-PHYS-PUB-2005-005.
[18] Richter-Was E., Przysiezniak H., and Tarrade, F. *Exploring hadronic τ identification with dc1 data samples: a track based approach*. Atlas Note ATL-PHYS-2004-030.
[19] Suykens J.A.K., Vandewalle J., *Least squares support vector machine classifiers*. Neural Processing Letters, 9(3) (1999) 293.
[20] Scholkopf B., Smola A., Williamson R. C., Bartlett P.L., *New support vector algorithms*. Neural Computation, 12 (2000) 1083.