EPJ Web of Conferences **116**, 07006 (2016)
DOI: 10.1051/epjconf/201611607006

Application of data mining techniques in atmospheric neutrino analyses with IceCube

T. Ruhe[a] for the IceCube collaboration[b]

Department of Physics, TU Dortmund, Germany

Abstract. The selection of event candidates by machine learning algorithms has become an important analysis tool. Data mining, however, goes beyond the simple training and application of a learning algorithm. It also incorporates finding a good representation of data in fewer dimensions without losing relevant information, as well as a thorough validation of the results throughout the entire analysis. A data mining-based event selection chain has been developed for the measurement of the atmospheric ν_μ spectrum with IceCube in the 59-string configuration. It yielded a high statistics and high purity sample ($99.59 \pm 0.37\%$) of ν_μ, while allowing only $1.0 \times 10^{-4}\%$ of the incoming background muons to pass. In this paper the setup of the analysis chain is presented and the results are discussed in the context of atmospheric ν_μ analyses.

1. Introduction

In the context of machine learning and data mining the techniques used by the IceCube neutrino telescope [1] are interesting for several reasons. IceCube is taking data on the order of 1 TB/day on a daily basis. These data have to be stored and processed, which poses a general computational challenge. Moreover, ν-induced μ are hidden in a dominant background of atmospheric muons, which makes the efficient rejection of background a key task in the IceCube analysis chain. From the machine learning point of view the rejection of background corresponds to a two-class seperation task, with ν-induced muons being one class and atmospheric muons being the other one. In atmospheric ν_μ analyses the signal to background ratio is roughly 10^{-3}. A challenge therefore arises from the highly skewed distribution of classes.

In general, data mining starts with a set of annotated events, which in neutrino astronomy are often derived from Monte Carlo simulation. These annotated events are then fed into a learning algorithm, which delivers a model that can be applied to unseen data and will return a label for these. For neutrino telescopes, often a relatively large number of input variables is available (several hundred). It is preferrable to reduce the number of input variables in order to reduce the dimensionality of the problem and to limit the required resources – CPU time and memory – to a reasonable amount. The selection of

[a]e-mail: tim.ruhe@tu-dortmund.de
[b] http://icecube.wisc.edu

the variables as well as the performance of the learning algorithm has to be carefully validated before the analysis chain is applied to real data. In total 120,000 simulated ν_μ events and 4.96×10^6 simulated background events were available for verifying the analysis chain. This corresponds to a detector lifetime of ≈ 15 days.

The analysis chain used in atmospheric ν_μ analyses for the IceCube detector in the 59-string configuration consists of three consecutive steps. The first one is the application of simple straight cuts, intended to reduce obvious background events and the required CPU resources. These cuts were applied at the zenith angle ($\theta_{\text{Zenith}} > 88^\circ$ and the estimated velocity of the lepton $v_{\text{Lepton}} > 0.19\,\text{c}$). Their application yielded a background rejection of 91.4% at a signal efficiency of 57.1%. The remaining background solely consists of falsely reconstructed atmospheric μ, which are removed by using machine learning algorithms.

In a second step the input variables for the learning algorithm are selected in a partially automated selection procedure. This selection procedure is carried out on a limited number of simulated events. Therefore, its stability with respect to statistical fluctuations in these sets needs to be ensured. In a third step a Random Forest [2] is trained and tested using the input parameters selected in step two. All machine learning algorithms were used in the data mining environment RapidMiner [3].

2. Variable selection

The selection of the input variables is carried out using the Maximum Relevance Minimum Redundancy (MRMR) algorithm [4]. MRMR is an iterative procedure, in which one variable is added to the subset of variables per iteration according to its relevance V_F, as well as its redundancy W_c. The redundancy is computed with respect to the variables selected in previous iterations [4, 5].

Compared to Forward Selection or Background Elimination, MRMR is independent of a specific learning algorithm. Also the resource requirements in terms of CPU time and memory are reduced by approximately a factor of 10.

Since the variable selection is carried out on a limited number of events, statistical fluctuations need to be taken into account by monitoring the stability of the variable selection as a function of the number of variables. Two indices, the Jaccard index and Kuncheva's index [6] were used to judge the stability of the variable selection.

The Jaccard index for two subsets of variables A and B is computed via:

$$J = \frac{|A \cap B|}{|A \cup B|} \tag{1}$$

whereas Kuncheva's index is calculated via [6]:

$$I_C(A, B) = \frac{rn - k^2}{k(n - k)}. \tag{2}$$

In Eq. (2) k represents the size of the subset, whereas $r = |A \cap B|$ represents the cardinality of the intersection. The total number of variables available is denoted by n. For a fixed number of variables n_{var} 10 subsets of attributes are selected using MRMR. The stability indices are then computed for every pair of subsets and divided by the number of possible combinations [5]. The stability of the MRMR variable selection is shown in Fig. 1. In this analysis 25 variables were selected as input to the learning algorithm. For $n_{\text{var}} = 25$ both indices were found to be larger than 0.7.

3. Training and testing of a Random Forest

A Random Forest [2] uses an ensemble of n_{tree} simple decision trees to obtain a classification of events. A given event j is assigned a label s_{ij} by tree i. This label is $s_{ij} = 1$ in case the event is recognized as

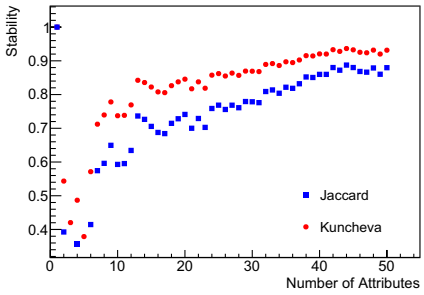


Figure 1. Stability of the Minimum Redundancy Maximum Relevance (MRMR) variable selection as a function of the number of variables selected.

signal and $s_{ij} = 0$ in case the event is recognized as background. The output returned by the forest s_j is an average over the output of the individual trees:

$$s_j = \frac{1}{n_{\text{trees}}} \sum_{i=1}^{n_{\text{trees}}} s_{ij}. \tag{3}$$

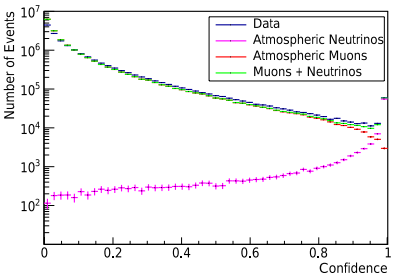
For the IC59 atmospheric ν_μ analysis, the Random Forest was trained setting the number of trees to $n_{\text{trees}} = 500$. The performance of the classification was validated in a five-fold cross validation using 750,000 simulated μ events and 70,000 simulated ν_μ events. Following the best practices for cross validating results, the events were divided in 5 statistically independent subsets. The training of the forest is then iterated 5 times using 4 of the 5 subsets, while holding out the 5th for testing. This leaves 650,000 simulated μ events and 56,000 simulated ν_μ events available for training per iteration. Studies showed that better results in terms of signal efficiency and background rejection were achieved using 27,000 simulated events of each class for training. Thus, in every iteration of the cross validation 27,000 events per class were sampled from the total number of events available for training. The use of a cross validation enables the analyzer to compute uncertainties on the performance of the learning algorithm, from which faulty behavior, such as overtraining, can be detected.

Background events were simulated with CORSIKA [7] according to the poly-gonato model [8]. Signal events were generated with NuGen an IceCube event generator based on ANIS [9], according to an E^{-2} spectrum. Although the spectrum of atmospheric ν_μ is roughly proportional to $E^{-3.7}$, events generated with a different spectral index can be used in the learning process. Tree-based algorithms classify events on an event-by-event basis, rather than on a statistical one. Furthermore, using events generated with flatter spectrum provides the learner with more events at high energies. The distribution of the random forest output variable s is shown in Fig. 2.

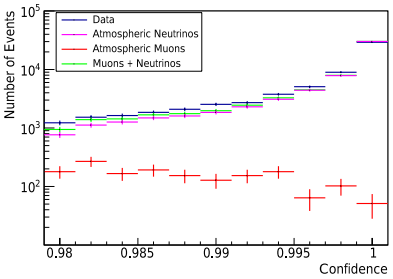
We find that due to the large number of background muons an additional cut needs to be placed on s. This cut was placed at $s = 1.0$, thus requiring that all trees in the forest classified an event as signal. The cut is rather strict due the overall analysis goal – an unfolding of an atmospheric ν_μ energy spectrum. In order to prevent a contamination of high energy bins with background muons, an event sample with a purity $\geq 99.5\%$ is required.

4. Overall performance of the event selection chain

The application of the event selection chain on data taken with IceCube-59 yielded 27,771 ν_μ candidates in a livetime of 346 days – roughly 80 ν_μ candidates per day. The number of remaining μ events was estimated to be 114 ± 103 for the entire lifetime. Comparing the number of detected events to 29,884 ν_μ



(a) Distribution of the Random Forest output s over the entire range.



(b) Distribution of the Random Forest output in the region from $s = 0.98$ and $s = 1.0$.

Figure 2. Distribution of the Random Forest output for experimental data (blue), simulated atm. ν_μ (magenta), simulated atm. μ (red) and the sum of simulated μ and simulated ν_μ (green).

candidates expected from simulation, we find that these numbers disagree by approximately 7%. This deficit, however, is found to be well within the systematic uncertainty of the event selection.

Thus, a purity of $(99.59 \pm 0.37)\%$ was achieved while retaining $1,0 \times 10^{-4}$ of the incident background muons, computed with respect to the starting level of the analysis (before the application of straight cuts). Comparing these results to the event selection chain of IceCube in the 40-string configuration we find that the event rate is increased by $\approx 8\%$ at a comparable purity. The event selection chain presented was found to be robust and reliable and has been applied to other detector configurations with only minor changes.

References

[1] A. Achterberg, et al., Astroparticle Physics **26**, 155 (2006), `astro-ph/0604450`
[2] L. Breiman, Machine Learning **45**, 5 (2001)
[3] S. Fischer, et al., Tech. Rep. CI-136/02, Collaborative Research Center 531, University of Dortmund, Dortmund, Germany (2002)
[4] C. Ding, H. Peng, J. of Bioinformatics and Computational Biology **3** (2005)
[5] B. Schowe, *Feature Selection for high-dimensional data in RapidMiner*, in *Proceedings of the 2nd RapidMiner Community Meeting And Conference (RCOMM 2011)*, edited by S. Fischer, I. Mierswa (Shaker Verlag, Aachen, 2011)
[6] L. Kuncheva, *A Stability Index for Feature Selection*, in *Artificial intelligence and applications* (2007), pp. 421–427
[7] D. Heck, et al., *CORSIKA: A Monte Carlo code to simulate extensive air showers*, Vol. 6019 (FZKA, 1998)
[8] J.R. Hoerandel, N.N. Kalmykov, A.I. Pavlov, *The Knee in the Energy Spectrum of Cosmic Rays in the Framework of the Poly-Gonato and Diffusion Models*, in *Proceedings of International Cosmic Ray Conference 2003* (2003), Vol. 1, p. 243
[9] A. Gazizov, M. Kowalski, Computer Physics Communications **172**, 203 (2005), `astro-ph/0406439`