87944 - STATISTICAL DATA ANALYSIS FOR NUCLEAR AND SUBNUCLEAR PHYSICS

Module 3 : Laboratory of Stat. Data Analysis for Nucl. and SubNucl. Physics
teacher: G. Sirri

Hands-on n. 4: TMVA

Submission Deadline: before the examination (Recommended within two weeks)

This assignment involves solving an exercise by writing a C++ macro for ROOT. Alternatively, solutions can be implemented using the RooFit libraries in a Python+ROOT script or a Jupyter notebook. However, the provided hints are tailored for C++.

Teamwork is allowed, but each student must submit their own individual work.

DOCUMENTATION:

- slides shown during the lecture.

- TMVA ROOT website: https://root.cern/manual/tmva/

- TMVA Tutorials: https://root.cern/doc/master/group__tutorial__tmva.html

- TMVA Manual: https://github.com/root-project/root/blob/master/documentation/tmva/UsersGuide/TMVAUsersGuide.pdf

- an example how to perform the calssification (with Jupyter) https://nbviewer.jupyter.org/url/root.cern/doc/master/notebooks/classification.C.nbconvert.ipynb

[Exercise TMVA] - ATLAS full-detector simulation, with "Higgs to tautau" events

Inspired by arXiv:1501.04943v3 [hep-ex] 27 Apr 2015
https://arxiv.org/abs/1501.04943

Please visit the “Dataset from the ATLAS Higgs Boson Machine Learning Challenge 2014” at CERN OPEN DATA http://opendata.cern.ch/record/328, read the dataset and attribute information (i.e., dataset semantics) and download the dataset.

The dataset is based on the official ATLAS full-detector simulation, with "Higgs to tautau" events mixed with various background processes. It contains 818,238 events (279,560 signal events and 539,768 background events). The dataset is stored in a CSV file.

> *Hint: you may use the **prepare_dataset.cpp** macro to read the CSV and create a TTree. This macro utilizes ROOT::RDataFrame, a high-level interface for analyzing data stored in TTree, CSV, and other formats.*

Note: Do not use the following variables as INPUT VARIABLES for the MVA analysis:

“EventId”, “KaggleSet”, “KaggleWeight”

“Weight”, as it represents the event weight and must be correctly assigned as the individual event weight.

“Label”, since it is used to distinguish the signal sample from the background sample. This variable would not be available in a real dataset.

--

For this exercise you shall perform a simple multivariate analysis using TMVA and other ROOT routines.

TMVA generates many auxiliary files, so it's highly recommended to create a new working directory to keep your analysis clean.

he easiest way to start with TMVA is to run an example macro:

```
$ROOTSYS/tutorials/tmva/TMVAClassification.C.
```

Hint: To avoid cluttering your ROOT installation, copy the macro into your working directory.

Modify the file TMVAClassification.C (or write your own **tmva_training.cpp** macro following the example provided in the lecture) in such a way that a data loader is created to perform the following actions:

- define the input variables to be used for the MVA training;
- read in the “signal” and the “background” datasets;

General Hint: The data loader is versatile and can handle input data from both ROOT files and text files. Refer to the TMVA User’s Guide and note that in the guide, methods like AddBackgroundTree(..), AddSignalTree(..), SetInputTrees(..) belongs to the factory, but in recent ROOT versions, these are methods of the data loader.

- *Code example 6: signal and background stored as TTrees from separate TFile sources;*
- *Code example 7: signal and background stored in the same TTree from the same TFile source;*
- *Code example 9; signal and background stored in different text files. If using text files, you can preprocess the input before reading data.*

Specific Hint: The prepare_dataset.cpp script creates two input files, one for the signal and one for the background. Therefore, Code Example 6 applies to this case.

- Set the individual event weight for signal and background

Book a list of classifiers, as for instance: Cuts, Fisher, MLPBNN, and BDT. The first is an automatic rectangular cut optimization, the second one is the Fisher discriminant, the third one is a neural network, and the last one is a standard boosted decision tree.

Now run training and testing. TMVA automatically splits the data internally into a training set and a test set.

~~*Hint: see the slides of the course. When you prepare the training and test tree consider the following options to split the sample in order to use half of the events for training, half for testing*~~

~~NTrain_Signal=0:NTrain_Background=0:NTest_Signal=0:NTest_Background=0~~

Hint: Refer to the course slides for more details. When preparing the training and test trees, consider the following options to split the sample so that 10k (or 20k) signal (background) events are used for training, and the remaining events are used for testing:

```
NTrain_Signal=10000:NTrain_Background=20000:NTest_Signal=0:NTest_Background=0
```

Hint: To book and configure the Multilayer Perceptron (MLP), use a line like this (see the TMVA manual for more details):

```
factory->BookMethod(TMVA::Types::kMLP, "MLP", "H:!V:HiddenLayers=N+5");
```

Hint: To book and configure a Boosted Decision Tree (BDT) with 200 boosting iterations, use a line like this (See the TMVA manual for more details):

```
factory->BookMethod(TMVA::Types::kBDT, "BDT", "NTrees=200:BoostType=AdaBoost");
```

FISHER: determine the coefficients of a Fisher discriminant.

When you run the program, the coefficients of the discriminating functions are written into a subdirectory weights as text files. Take a look at these files and identify the relevant coefficients.

MLP & BDT: Coefficients of the multilayer perceptron and BDT are stored in a file in the **weights** subdirectory.

Open the TMVAGUI:

Have a look at the input variable distributions.

Look how different the signal and the background are distributed.

Assume that the expected number of respectively signal and background events are 2000 and 6000.

At which values would you cut to separate the signal from the background?

submit the source code, relevant plots (not more than 5), a text file with your answer

For willing students:

the macro called by the "(5a) Classifier Cut Efficiencies" button of TMVAGUI compute a significance using an approximated formula Z = sqrt(S /(S+B)) (S – number of signal, B – number of background).

create a new macro **my_mvaeff.cpp** based on $ROOTSYS/include/tmva/mvaeff.h (or mvaeff.C in ROOT 5) to estimate the *expected discovery significance* with the following formula:

$$Z = \sqrt{2\left((s+b)\ln\left(1+\frac{s}{b}\right)-s\right)}$$

G. Cowan et al., "Asymptotic formulae for likelihood-based tests of new physics", ., Eur. Phys. J. C71, 1554 (2011)

Submit the plot created by the modified macro