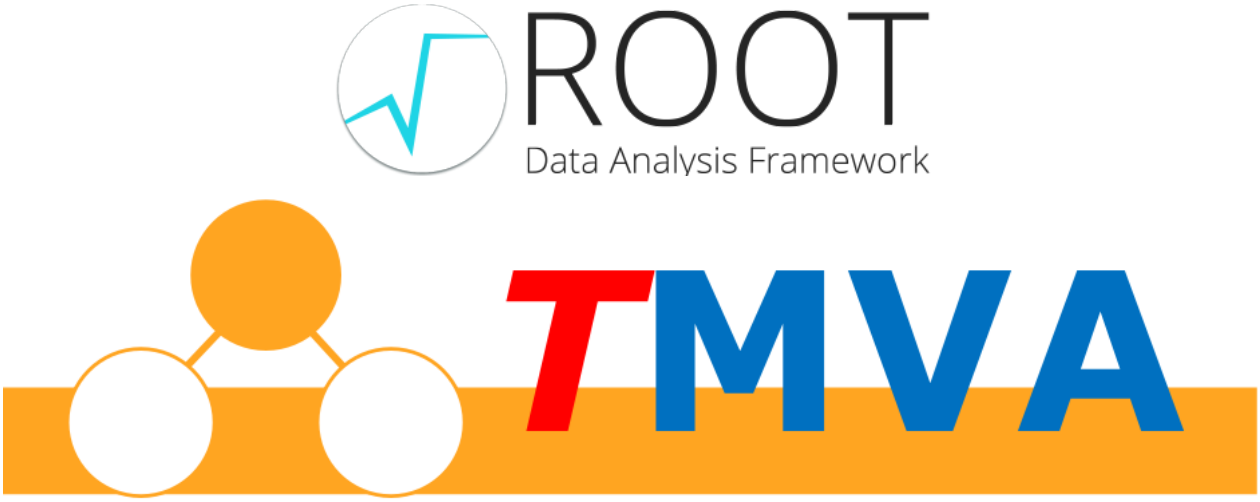


Exercise 4: TMVA

This exercise will get you started on using TMVA. TMVA is the Toolkit for Multivariate Data Analysis with ROOT (more information http://tmva.sourceforge.net/ (http://tmva.sourceforge.net/)) and it is a powerful tool to perform multivariate analysis.
This is a short introduction to the standard functionality with some objectives at the end.

In [1]:

```
from ROOT import TH1D,TH2D,TFile,TTree,TCanvas,TCut,TMVA
```

```
Welcome to ROOTaaS 6.06/04
```

Let's read in a ROOT file which contains a signal and background tree with some variables `var1, var2, var3, var4`.

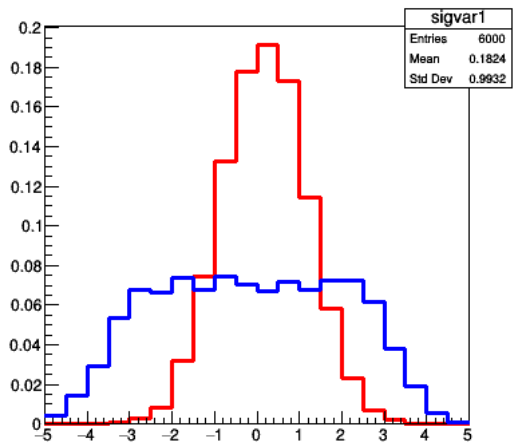
In [2]:

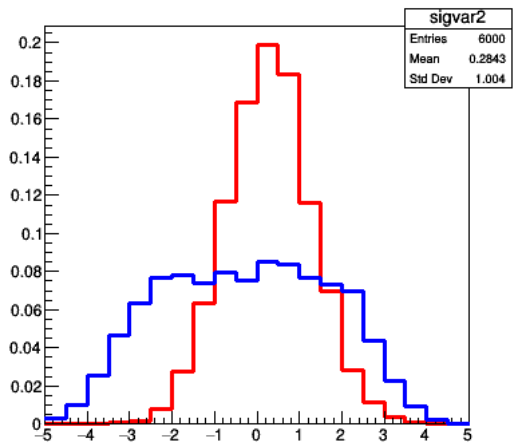
```
fin = TFile("toy_sigbkg.root")
signal     = fin.Get("TreeS")
background = fin.Get("TreeB")
```

We create a canvas and histograms to plot two of the variables for signal (red) and background (blue).

In [3]:

```
c = TCanvas("c", "c", 400, 400)
sigvar1 = TH1D("sigvar1","",20,-5,5)
bkgvar1 = TH1D("bkgvar1","",20,-5,5)

signal.Draw("var1>>sigvar1")
background.Draw("var1>>bkgvar1","","SAME")

# draw the signal events in red
sigvar1.SetLineColor(2)
sigvar1.Scale(1/sigvar1.Integral())
# draw the background events in blue
bkgvar1.SetLineColor(4)
bkgvar1.Scale(1/bkgvar1.Integral())

c.Draw()
sigvar2 = TH1D("sigvar2","",20,-5,5)
bkgvar2 = TH1D("bkgvar2","",20,-5,5)

signal.Draw("var2>>sigvar2")
background.Draw("var2>>bkgvar2","","SAME")

# draw the signal events in red and normalize
sigvar2.SetLineColor(2)
sigvar2.Scale(1/sigvar2.Integral())
# draw the background events in blue
bkgvar2.SetLineColor(4)
bkgvar2.Scale(1/bkgvar2.Integral())

c.Draw()
```

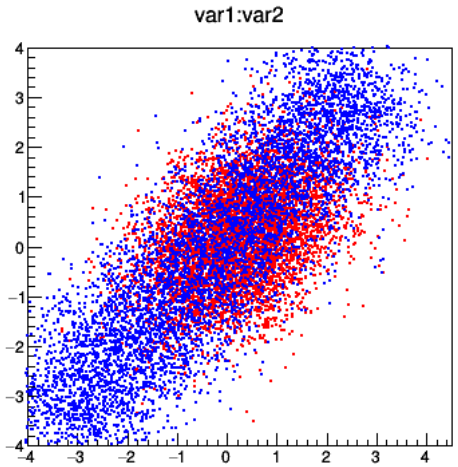




There is some clear separation between signal and background, but it is not easy to cut on any value.
And that is not all, the variables can be also correlated, lets plot them also in a 2D plot.

In [4]:

```
# draw an empty 2D histogram for the axes
hsig = TH2D("histosig","",1,-5,5,1,-5,5)
hbkg = TH2D("histobkg","",1,-5,5,1,-5,5)

sigCut = TCut("")
bgCut = TCut("")

signal.SetMarkerColor(2)
signal.Draw("var1:var2>>hsig") #,sigCut,"same")
background.SetMarkerColor(4)
background.Draw("var1:var2>>hbkg","","SAME") #,bgCut,"same")

c.Draw()
```



There is clearly a stron correlation for the background events.
You can see, that just by looking at the two variables, it is hard to separate signal from background, there is no clear cut to apply. So maybe we should use a multivariate technique to combine them in order to find a better cut value.

We will use TMVA for this, first we need to create the TMVA factory, the basic component of TMVA for the training of multivariate methods.

In [5]:

```
TMVA.Tools.Instance()

# note that it seems to be mandatory to have an
# output file, just passing None to TMVA::Factory(..)
# does not work. Make sure you don't overwrite an
# existing file.
fout = TFile("TMVAResults.root","RECREATE")
factory = TMVA.Factory("TMVAClassification", fout,
                           ":".join([
                               "!V",
                               "!Silent",
                               "!Color",
                               "!DrawProgressBar",
                               "Transformations=I;D;P;G,D",
                               "AnalysisType=Classification"]
                                    ))
```

```
--- Factory                  : You are running ROOT Version: 6.06/04, May 3, 2016
--- Factory                  :
--- Factory                  : _/_/_/_/_/ _|       _|  _|       _|    _|_|
--- Factory                  :    _/       _|_|  _|_|  _|       _|  _|    _|
--- Factory                  :   _/        _|  _|  _|  _|       _|  _|_|_|_|
--- Factory                  :  _/         _|      _|    _|  _|    _|    _|
--- Factory                  : _/          _|      _|      _|      _|    _|
--- Factory                  :
--- Factory                  : ___________TMVA Version 4.2.1, Feb 5, 2015
--- Factory                  :
```

Add the variables and trees to the factory. TMVA will handle them completely automatically.

In [6]:

```
factory.AddVariable("var1","F")
factory.AddVariable("var2","F")
factory.AddVariable("var3","F")
factory.AddVariable("var4","F")
factory.AddSignalTree(signal)
factory.AddBackgroundTree(background)
```

```
--- DataSetInfo              : Added class "Signal"      with internal class number 0
--- Factory                  : Add Tree TreeS of type Signal with 6000 events
--- DataSetInfo              : Added class "Background"  with internal class number 1
--- Factory                  : Add Tree TreeB of type Background with 6000 events
```

A preparation function defines how many events should be used for training and how the sample is split, what cuts should be applied and the normalization method.

In [7]:

```
factory.PrepareTrainingAndTestTree(sigCut,bgCut,
                        "nTrain_Signal=1000:nTrain_Background=1000:SplitMode=Random:NormMode=NumEvents:!V" )
```

```
--- Factory                  : Preparing trees for training and testing...
```

Now we book the multivariate method to train. In this case we are training a BDT and we specify the settings:

- NTrees: number of trees
- nEventsMin: minimum events per tree
- MaxDepth: maximum depth of the decision tree allowed
- BoostType: boosting algorithm used. In this case it is the adaptive boost
- AdaBoostBeta: specific parameter of the adaptive boost algorithm
- SeparationType: separation criterion for node splitting
- nCuts: number of grid points in variable range used in finding optimal cut in node splitting
- PruneMethod: method used to prune the tree. In this case no pruning is applied.

In [8]:

```
methodBDT = factory.BookMethod(TMVA.Types.kBDT, "BDT",
                   ":".join([
                       "!H",
                       "!V",
                       "NTrees=850",
                       "nEventsMin=150",
                       "MaxDepth=3",
                       "BoostType=AdaBoost",
                       "AdaBoostBeta=0.5",
                       "SeparationType=GiniIndex",
                       "nCuts=20",
                       "PruneMethod=NoPruning",
                       ]))
```

```
--- Factory                  : Booking method: BDT
--- DataSetFactory           : Splitmode is: "RANDOM" the mixmode is: "SAMEASSPLITMODE"
--- DataSetFactory           : Create training and testing trees -- looping over class "Signal" ...
--- DataSetFactory           : Weight expression for class 'Signal': ""
--- DataSetFactory           : Create training and testing trees -- looping over class "Background" ...
--- DataSetFactory           : Weight expression for class 'Background': ""
--- DataSetFactory           : Number of events in input trees (after possible flattening of arrays):
--- DataSetFactory           :     Signal          -- number of events       : 6000   / sum of weights: 6000
--- DataSetFactory           :     Background      -- number of events       : 6000   / sum of weights: 6000
--- DataSetFactory           :     Signal     tree -- total number of entries: 6000
--- DataSetFactory           :     Background tree -- total number of entries: 6000
--- DataSetFactory           : Preselection: (will NOT affect number of requested training and testing events)
--- DataSetFactory           :     No preselection cuts applied on event classes
--- DataSetFactory           : Weight renormalisation mode: "NumEvents": renormalises all event classes
--- DataSetFactory           :  such that the effective (weighted) number of events in each class equals the respective
--- DataSetFactory           :  number of events (entries) that you demanded in PrepareTrainingAndTestTree("","nTrain_Signa
l=.. )
--- DataSetFactory           :  ... i.e. such that Sum[i=1..N_j]{w_i} = N_j, j=0,1,2...
--- DataSetFactory           :  ... (note that N_j is the sum of TRAINING events (nTrain_j...with j=Signal,Background..
--- DataSetFactory           :  ..... Testing events are not renormalised nor included in the renormalisation factor! )
--- DataSetFactory           : --> Rescale Signal     event weights by factor: 1
--- DataSetFactory           : --> Rescale Background event weights by factor: 1
--- DataSetFactory           : Number of training and testing events after rescaling:
--- DataSetFactory           : ------------------------------------------------------
--- DataSetFactory           : Signal     -- training events            : 1000 (sum of weights: 1000) - requested were 1000
events
--- DataSetFactory           : Signal     -- testing events             : 5000 (sum of weights: 5000) - requested were 0 ev
ents
--- DataSetFactory           : Signal     -- training and testing events: 6000 (sum of weights: 6000)
--- DataSetFactory           : Background -- training events            : 1000 (sum of weights: 1000) - requested were 1000
events
--- DataSetFactory           : Background -- testing events             : 5000 (sum of weights: 5000) - requested were 0 ev
ents
--- DataSetFactory           : Background -- training and testing events: 6000 (sum of weights: 6000)
--- DataSetFactory           : Create internal training tree
--- DataSetFactory           : Create internal testing tree
--- DataSetInfo              : Correlation matrix (Signal):
--- DataSetInfo              : ----------------------------------------
--- DataSetInfo              :             var1    var2    var3    var4
--- DataSetInfo              :    var1:  +1.000  +0.386  +0.597  +0.808
--- DataSetInfo              :    var2:  +0.386  +1.000  +0.696  +0.743
--- DataSetInfo              :    var3:  +0.597  +0.696  +1.000  +0.860
--- DataSetInfo              :    var4:  +0.808  +0.743  +0.860  +1.000
--- DataSetInfo              : ----------------------------------------
--- DataSetInfo              : Correlation matrix (Background):
--- DataSetInfo              : ----------------------------------------
--- DataSetInfo              :             var1    var2    var3    var4
--- DataSetInfo              :    var1:  +1.000  +0.856  +0.914  +0.964
--- DataSetInfo              :    var2:  +0.856  +1.000  +0.927  +0.937
--- DataSetInfo              :    var3:  +0.914  +0.927  +1.000  +0.971
--- DataSetInfo              :    var4:  +0.964  +0.937  +0.971  +1.000
--- DataSetInfo              : ----------------------------------------
--- DataSetFactory           :
--- <WARNING> BDT                      : You have explicitly set ** nEventsMin = 150 ** the min ablsolut number
--- <WARNING> BDT                      : of events in a leaf node. This is DEPRECATED, please use the option
--- <WARNING> BDT                      : *MinNodeSize* giving the relative number as percentage of training
--- <WARNING> BDT                      : events instead.
--- <WARNING> BDT                      : nEventsMin=150--> MinNodeSize=7.5%
--- <WARNING> BDT                      : Note also that explicitly setting *nEventsMin* so far OVERWRITES the option recome
nded
--- <WARNING> BDT                      :  *MinNodeSize* = 5% option !!
```

Everything is setup, now train, test and evaluate all methods (the BDT in this case).

Train Methods

In [9]:

```
factory.TrainAllMethods()
```

```
--- Factory                  :
--- Factory                  : current transformation string: 'I'
--- Factory                  : Create Transformation "I" with events from all classes.
--- Id                       : Transformation, Variable selection :
--- Id                       : Input : variable 'var1' (index=0).   <---> Output : variable 'var1' (index=0).
--- Id                       : Input : variable 'var2' (index=1).   <---> Output : variable 'var2' (index=1).
--- Id                       : Input : variable 'var3' (index=2).   <---> Output : variable 'var3' (index=2).
--- Id                       : Input : variable 'var4' (index=3).   <---> Output : variable 'var4' (index=3).
--- Factory                  :
--- Factory                  : current transformation string: 'D'
--- Factory                  : Create Transformation "D" with events from all classes.
--- Deco                     : Transformation, Variable selection :
--- Deco                     : Input : variable 'var1' (index=0).   <---> Output : variable 'var1' (index=0).
--- Deco                     : Input : variable 'var2' (index=1).   <---> Output : variable 'var2' (index=1).
--- Deco                     : Input : variable 'var3' (index=2).   <---> Output : variable 'var3' (index=2).
--- Deco                     : Input : variable 'var4' (index=3).   <---> Output : variable 'var4' (index=3).
--- Factory                  :
--- Factory                  : current transformation string: 'P'
--- Factory                  : Create Transformation "P" with events from all classes.
--- PCA                      : Transformation, Variable selection :
--- PCA                      : Input : variable 'var1' (index=0).   <---> Output : variable 'var1' (index=0).
--- PCA                      : Input : variable 'var2' (index=1).   <---> Output : variable 'var2' (index=1).
--- PCA                      : Input : variable 'var3' (index=2).   <---> Output : variable 'var3' (index=2).
--- PCA                      : Input : variable 'var4' (index=3).   <---> Output : variable 'var4' (index=3).
--- Factory                  :
--- Factory                  : current transformation string: 'G,D'
--- Factory                  : Create Transformation "G" with events from all classes.
--- Gauss                    : Transformation, Variable selection :
--- Gauss                    : Input : variable 'var1' (index=0).   <---> Output : variable 'var1' (index=0).
--- Gauss                    : Input : variable 'var2' (index=1).   <---> Output : variable 'var2' (index=1).
--- Gauss                    : Input : variable 'var3' (index=2).   <---> Output : variable 'var3' (index=2).
--- Gauss                    : Input : variable 'var4' (index=3).   <---> Output : variable 'var4' (index=3).
--- Factory                  : Create Transformation "D" with events from all classes.
--- Deco                     : Transformation, Variable selection :
--- Deco                     : Input : variable 'var1' (index=0).   <---> Output : variable 'var1' (index=0).
--- Deco                     : Input : variable 'var2' (index=1).   <---> Output : variable 'var2' (index=1).
--- Deco                     : Input : variable 'var3' (index=2).   <---> Output : variable 'var3' (index=2).
--- Deco                     : Input : variable 'var4' (index=3).   <---> Output : variable 'var4' (index=3).
--- Id                       : Preparing the Identity transformation...
--- TFHandler_Factory        : -------------------------------------------------------------
--- TFHandler_Factory        : Variable        Mean        RMS   [        Min        Max ]
--- TFHandler_Factory        : -------------------------------------------------------------
--- TFHandler_Factory        :     var1:   0.088993     1.6612   [    -4.9583     4.3390 ]
--- TFHandler_Factory        :     var2:   0.095657     1.5731   [    -4.6474     3.9513 ]
--- TFHandler_Factory        :     var3:    0.10551     1.7363   [    -5.0373     4.2785 ]
--- TFHandler_Factory        :     var4:    0.27815     2.
```

Test Methods

In [10]:

```
factory.TestAllMethods()
```

```
1526   [    -5.9505     4.6404 ]
--- TFHandler_Factory        : ---------------------------------------------------------
--- TFHandler_Factory        : Plot event variables for Id
--- TFHandler_Factory        : Create scatter and profile plots in target-file directory:
--- TFHandler_Factory        : TMVAResults.root:/InputVariables_Id/CorrelationPlots
--- Deco                     : Preparing the Decorrelation transformation...
--- TFHandler_Factory        : ---------------------------------------------------------
--- TFHandler_Factory        : Variable        Mean        RMS   [        Min        Max ]
--- TFHandler_Factory        : ---------------------------------------------------------
--- TFHandler_Factory        :     var1:   -0.11061     1.0000   [    -3.0333     3.1504 ]
--- TFHandler_Factory        :     var2:  -0.055650     1.0000   [    -3.6232     3.3971 ]
--- TFHandler_Factory        :     var3:  -0.094061     1.0000   [    -3.2704     3.3920 ]
--- TFHandler_Factory        :     var4:    0.33403     1.0000   [    -3.4036     2.8754 ]
--- TFHandler_Factory        : ---------------------------------------------------------
--- TFHandler_Factory        : Plot event variables for Deco
--- TFHandler_Factory        : Create scatter and profile plots in target-file directory:
--- TFHandler_Factory        : TMVAResults.root:/InputVariables_Deco/CorrelationPlots
--- PCA                      : Preparing the Principle Component (PCA) transformation...
--- TFHandler_Factory        : ---------------------------------------------------------
--- TFHandler_Factory        : Variable        Mean        RMS   [        Min        Max ]
--- TFHandler_Factory        : ---------------------------------------------------------
--- TFHandler_Factory        :     var1:-3.0541e-09     3.4471   [    -9.5264     7.4519 ]
--- TFHandler_Factory        :     var2: 4.7089e-10    0.79667   [    -2.7149     2.8657 ]
--- TFHandler_Factory        :     var3: 4.8645e-10    0.51237   [    -1.8293     1.7177 ]
--- TFHandler_Factory        :     var4:-1.4414e-11    0.32022   [   -0.93958    0.93507 ]
--- TFHandler_Factory        : ---------------------------------------------------------
--- TFHandler_Factory        : Plot event variables for PCA
--- TFHandler_Factory        : Create scatter and profile plots in target-file directory:
--- TFHandler_Factory        : TMVAResults.root:/InputVariables_PCA/CorrelationPlots
--- Gauss                    : Preparing the Gaussian transformation...
--- Deco                     : Preparing the Decorrelation transformation...
--- TFHandler_Factory        : ---------------------------------------------------------
--- TFHandler_Factory        : Variable        Mean        RMS   [        Min        Max ]
--- TFHandler_Factory        : ---------------------------------------------------------
--- TFHandler_Factory        :     var1:   0.017693     1.0000   [    -4.9795     9.4170 ]
--- TFHandler_Factory        :     var2:   0.015693     1.0000   [    -2.8195     8.3312 ]
--- TFHandler_Factory        :     var3:   0.018793     1.0000   [    -2.7796     10.641 ]
--- TFHandler_Factory        :     var4:   0.010608     1.0000   [    -2.9338     13.805 ]
--- TFHandler_Factory        : ---------------------------------------------------------
--- TFHandler_Factory        : Plot event variables for Gauss_Deco
--- TFHandler_Factory        : Create scatter and profile plots in target-file directory:
--- TFHandler_Factory        : TMVAResults.root:/InputVariables_Gauss_Deco/CorrelationPlots
--- TFHandler_Factory        :
--- TFHandler_Factory        : Ranking input variables (method unspecific)...
--- IdTransformation         : Ranking result (top variable is best ranked)
--- IdTransformation         : ------------------------------
--- IdTransformation         : Rank : Variable  : Separation
--- IdTransformation         : ------------------------------
--- IdTransformation         :    1 : var4      : 3.585e-01
--- IdTransformation         :    2
```

Evaluate Methods

In [11]:

```
factory.EvaluateAllMethods()
```

```
 : var1      : 2.898e-01
--- IdTransformation         :    3 : var3      : 2.828e-01
--- IdTransformation         :    4 : var2      : 2.032e-01
--- IdTransformation         : ------------------------------
--- Factory                  :
--- Factory                  : Train all methods for Classification ...
--- Factory                  : Train method: BDT for Classification
--- BDT                      : Begin training
--- BDT                      :  found and suggest the following possible pre-selection cuts
--- BDT                      : as option DoPreselection was not used, these cuts however will not be performed, but the tra
ining will see the full sample
--- BDT                      :  found cut: Bkg if var 0 < -2.99038
--- BDT                      :  found cut: Bkg if var 2 < -2.88493
--- BDT                      :  found cut: Bkg if var 3 < -2.54088
--- BDT                      : <InitEventSample> For classification trees,
--- BDT                      :  the effective number of backgrounds is scaled to match
--- BDT                      :  the signal. Othersise the first boosting step would do 'just that'!
--- BDT                      : re-normlise events such that Sig and Bkg have respective sum of weights = 1
--- BDT                      :   sig->sig*1ev. bkg->bkg*1ev.
--- BDT                      : #events: (reweighted) sig: 1000 bkg: 1000
--- BDT                      : #events: (unweighted) sig: 1000 bkg: 1000
--- BDT                      : Training 850 Decision Trees ... patience please
--- BinaryTree               : The minimal node size MinNodeSize=7.5 fMinNodeSize=7.5% is translated to an actual number of
events = 150 for the training sample size of 2000
--- BinaryTree               : Note: This number will be taken as absolute minimum in the node,
--- BinaryTree               :       in terms of 'weighted events' and unweighted ones !!
--- BDT                      : <Train> elapsed time: 0.921 sec
--- BDT                      : <Train> average number of nodes (w/o pruning) : 9
--- BDT                      : End of training
--- BDT                      : Elapsed time for training with 2000 events: 0.924 sec
--- BDT                      : Create MVA output for classification on training sample
--- BDT                      : Evaluation of BDT on training sample (2000 events)
--- BDT                      : Elapsed time for evaluation of 2000 events: 0.215 sec
--- BDT                      : Creating weight file in xml format: weights/TMVAClassification_BDT.weights.xml
--- BDT                      : Creating standalone response class: weights/TMVAClassification_BDT.class.C
--- BDT                      : Write monitoring histograms to file: TMVAResults.root:/Method_BDT/BDT
--- Factory                  : Training finished
--- Factory                  :
--- Factory                  : Ranking input variables (method specific)...
--- BDT                      : Ranking result (top variable is best ranked)
--- BDT                      : ------------------------------------
--- BDT                      : Rank : Variable  : Variable Importance
--- BDT                      : ------------------------------------
--- BDT                      :    1 : var4      : 2.782e-01
--- BDT                      :    2 : var1      : 2.694e-01
--- BDT                      :    3 : var2      : 2.374e-01
--- BDT                      :    4 : var3      : 2.150e-01
--- BDT                      : ------------------------------------
--- Factory                  :
--- Factory                  : === Destroy and recreate all methods via weight files for testing ===
--- Factory                  :
--- MethodBase               : Reading weight file: weights/TMVAClassification_BDT.weights.xml
--- BDT                      : Read method "BDT" of type "BDT"
--- BDT                      : MVA method was trained with TMVA Version: 4.2.1
--- BDT                      : MVA method was trained with ROOT Version: 6.06/04
--- Factory                  : Test all methods...
--- Factory                  : Test method: BDT for Classification performance
--- BDT                      : Evaluation of BDT on testing sample (10000 events)
--- BDT                      : Elapsed time for evaluation of 10000 events: 0.73 sec
--- Factory                  : Evaluate all methods...
--- Factory                  : Evaluate classifier: BDT
--- BDT                      : Loop over test events and fill histograms with classifier response...
--- Factory                  : Write evaluation histograms to file
--- TFHandler_BDT            : Plot event variables for BDT
--- TFHandler_BDT            : ------------------------------------------------------------
--- TFHandler_BDT            : Variable        Mean        RMS   [        Min        Max ]
--- TFHandler_BDT            : ------------------------------------------------------------
--- TFHandler_BDT            :     var1: 0.00077102     1.6695   [    -5.8991     4.7639 ]
--- TFHandler_BDT            :     var2: -0.0063164     1.5765   [    -5.2454     4.8300 ]
--- TFHandler_BDT            :     var3:  -0.010870     1.7365   [    -5.3563     4.6430 ]
--- TFHandler_BDT            :     var4:    0.14557     2.1608   [    -6.9675     5.0307 ]
--- TFHandler_BDT            : ------------------------------------------------------------
--- TFHandler_BDT            : Create scatter and profile plots in target-file directory:
--- TFHandler_BDT            : TMVAResults.root:/Method_BDT/BDT/CorrelationPlots
--- Factory                  : Correlations between input variables and MVA response (signal):
--- Factory                  : ----------------
--- Factory                  :              BDT
--- Factory                  :    var1:  -0.248
--- Factory                  :    var2:  -0.175
--- Factory                  :    var3:  -0.137
--- Factory                  :    var4:  -0.023
--- Factory                  : ----------------
--- Factory                  :
--- Factory                  : Correlations between input variables and MVA response (background):
--- Factory                  : ----------------
--- Factory                  :              BDT
--- Factory                  :    var1:  +0.308
--- Factory                  :    var2:  +0.328
--- Factory                  :    var3:  +0.357
--- Factory                  :    var4:  +0.395
--- Factory                  : ----------------
```

```
--- Factory                  :
--- Factory                  : The following "overlap" matrices contain the fraction of events for which
--- Factory                  : the MVAs 'i' and 'j' have returned conform answers about "signal-likeness"
--- Factory                  : An event is signal-like, if its MVA output exceeds the following value:
--- Factory                  : --------------------
--- Factory                  :  Method:  Cut value:
--- Factory                  : --------------------
--- Factory                  :     BDT:     +0.005
--- Factory                  : --------------------
--- Factory                  : which correspond to the working point: eff(signal) = 1 - eff(background)
--- Factory                  :
--- Factory                  : Evaluation results ranked by best signal efficiency and purity (area)
--- Factory                  : -------------------------------------------------------------------------------
--- Factory                  : MVA              Signal efficiency at bkg eff.(error):       | Sepa-    Signifi-
--- Factory                  : Method:          @B=0.01    @B=0.10    @B=0.30    ROC-integ. | ration:  cance:
--- Factory                  : -------------------------------------------------------------------------------
--- Factory                  : BDT            : 0.289(06)  0.721(06)  0.932(03)    0.911    | 0.528    1.386
--- Factory                  : -------------------------------------------------------------------------------
--- Factory                  :
--- Factory                  : Testing efficiency compared to training efficiency (overtraining check)
--- Factory                  : -------------------------------------------------------------------------------
--- Factory                  : MVA              Signal efficiency: from test sample (from training sample)
--- Factory                  : Method:          @B=0.01             @B=0.10            @B=0.30
--- Factory                  : -------------------------------------------------------------------------------
--- Factory                  : BDT            : 0.289 (0.675)       0.721 (0.836)      0.932 (0.947)
--- Factory                  : -------------------------------------------------------------------------------
--- Factory                  :
--- Dataset:Default          : Created tree 'TestTree' with 10000 events
--- Dataset:Default          : Created tree 'TrainTree' with 2000 events
--- Factory                  :
--- Factory                  : Thank you for using TMVA!
--- Factory                  : For citation information, please visit: http://tmva.sf.net/citeTMVA.html
```

Great we have fully trained, tested and evaluated a BDT, with just a few lines of code. We got already a lot of useful and interesting output and TMVA has a GUI to look at the results. However you can also manually read and plot everything.
Let's plot the results of the training, we need a new instance and the TMVA Reader for this:

In [12]:

```
from ROOT import TH1D,TH2D,TFile,TTree,TCanvas,TCut,TMVA

TMVA.Tools.Instance()

reader = TMVA.Reader()
import array
var1= array.array('f',[0])
var2= array.array('f',[0])
var3= array.array('f',[0])
var4= array.array('f',[0])
reader.AddVariable("var1",var1)
reader.AddVariable("var2",var2)
reader.AddVariable("var3",var3)
reader.AddVariable("var4",var4)
reader.BookMVA("BDT","weights/TMVAClassification_BDT.weights.xml")
```

Out[12]:

```
<ROOT.TMVA::MethodBDT object ("BDT") at 0x5f14500>
```

```
--- Reader                   : Booking "BDT" of type "BDT" from weights/TMVAClassification_BDT.weights.xml.
--- MethodBase               : Reading weight file: weights/TMVAClassification_BDT.weights.xml
--- BDT                      : Read method "BDT" of type "BDT"
--- BDT                      : MVA method was trained with TMVA Version: 4.2.1
--- BDT                      : MVA method was trained with ROOT Version: 6.06/04
--- DataSetInfo              : Added class "Signal"      with internal class number 0
--- DataSetInfo              : Added class "Background"  with internal class number 1
--- Reader                   : Booked classifier "BDT" of type: "BDT"
```

First, we want to plot the output distributions for signal and backround, so lets create two histograms and fill them.

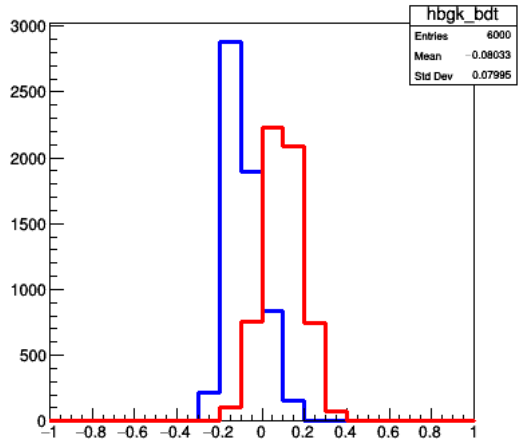
In [13]:

```
hsig_bdt = TH1D("hsig_bdt","",20,-1,1)
hbkg_bdt = TH1D("hbgk_bdt","",20,-1,1)
for i in signal:
    var1[0] = signal.var1
    var2[0] = signal.var2
    var3[0] = signal.var3
    var4[0] = signal.var4
    hsig_bdt.Fill(reader.EvaluateMVA("BDT"))

for i in background:
    var1[0] = background.var1
    var2[0] = background.var2
    var3[0] = background.var3
    var4[0] = background.var4
    hbkg_bdt.Fill(reader.EvaluateMVA("BDT"))
```

Give them color and draw:

In [14]:

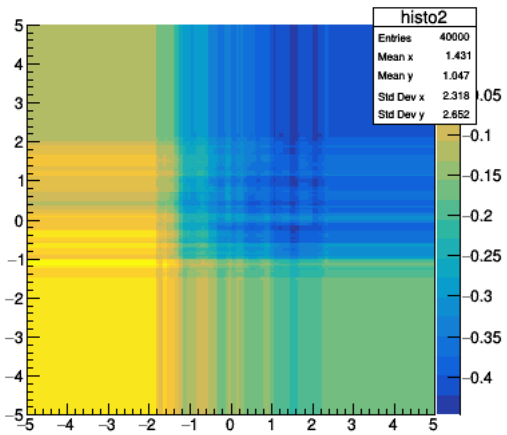
```
hbkg_bdt.SetLineColor(4)
hsig_bdt.SetLineColor(2)
hbkg_bdt.Draw()
hsig_bdt.Draw("SAME")
c.Draw()
```



Ok, this way we can much better separate signal from background and define a cut value! To better understand what is going and the correlations of the two variables, we can plot the BDT output dependent on the two variables in a 2D plot

In [15]:

```
# create a new 2D histogram with fine binning
histo2 = TH2D("histo2","",200,-5,5,200,-5,5)
# loop over the bins of a 2D histogram
for i in range(1,histo2.GetNbinsX() + 1):
    for j in range(1,histo2.GetNbinsY() + 1):

        # find the bin center coordinates
        var1[0] = histo2.GetXaxis().GetBinCenter(i)
        var2[0] = histo2.GetYaxis().GetBinCenter(j)

        # calculate the value of the classifier
        # function at the given coordinate
        bdtOutput = reader.EvaluateMVA("BDT")

        # set the bin content equal to the classifier output
        histo2.SetBinContent(i,j,bdtOutput)

c2 = TCanvas("c2", "c2", 800, 400)
c2.Divide(2,1)
c2.cd(1)
histo2.Draw("colz")
c2.Draw()
```



Ok, now you should know enough to be able to play with TMVA and understand what you are doing.

Instructions:

- **Setting everything up**
  - Copy the TMVA tutorial `cp -r $ROOTSYS/tutorials/tmva mytmva`
  - Go to mytmva directory and open the file `TMVAClassification.C`
  - You should be familiar now with the general procedure in the file, here we have switches `Use[XXX]` for the different MVA methods.
  - Switch on only `Likelihood`, `MLP` and `BDT`.
  - Which variables are used in the MVA?
- **Running the MVA**
  - Close the file and run `root TMVAClassification.C`
  - Wait a moment until the training and evaluation is finished.
  - A GUI should pop up with many buttons to plot the output, click on it and understand the plots. They are saved once you click on the button.
  - Using `TMVA::TMVAGui("TMVA.root")` should open the GUI with the file again, however this depends a bit on the ROOT version.
- **Questions**
  - Do you get the same correlation between `var1` and `var2` as in Exercise3?
  - Which method gives the best performance?
  - Play with the number of training events, how does it effect the training?
  - Why do we only use part of the statistic for training?
  - Finally, switch on other methods (not all), which one is the best?