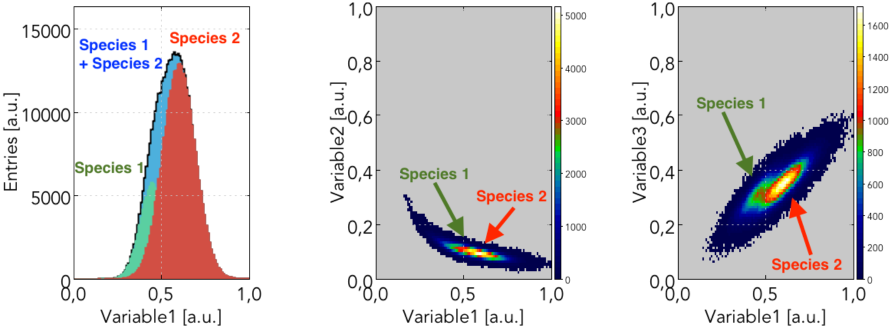
Machine Learning and CLAS12 PID: Tools and Concepts for solving **CLAS**sification Problems

Daniel Lersch / Michael C. Kunkel

Juelich Research Center

13.06.2017

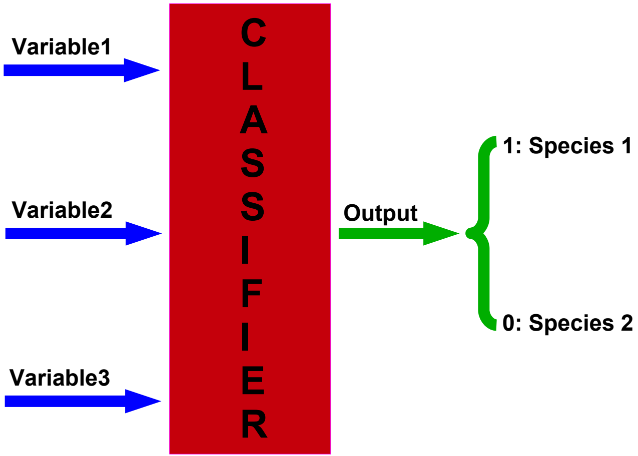
Introduction and Motivation



- Assume a **Dataset** with: **Species 1** and **Species 2**, where: $R \equiv \frac{N(\text{Species 1})}{N(\text{Species 2})} = \frac{1}{3}$
- The features of each species are charatcerised by variable 1,2, and 3 (e.g. momentum, energy, time etc.)
- Interested in **Species 1** (could be a particle type or a certain decay) for further analysis, but:
  - **Species 1** is the minority (e.g. electrons vs. pions)
  - All variables are correlated (i.e. rectangular cut is not sufficient)

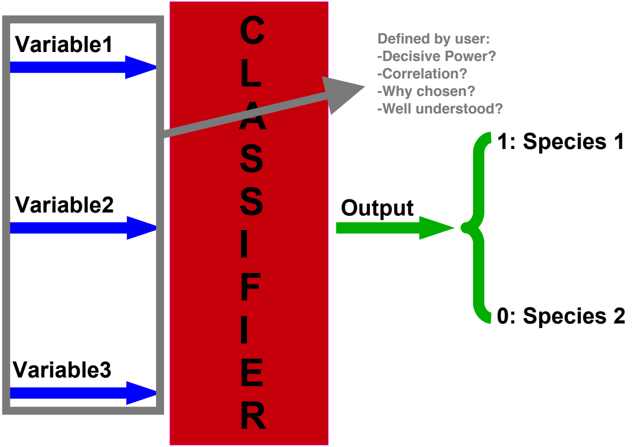
$\Rightarrow$ **Use machine learning based classifier to separate between the two species**

Idea of a Classifier



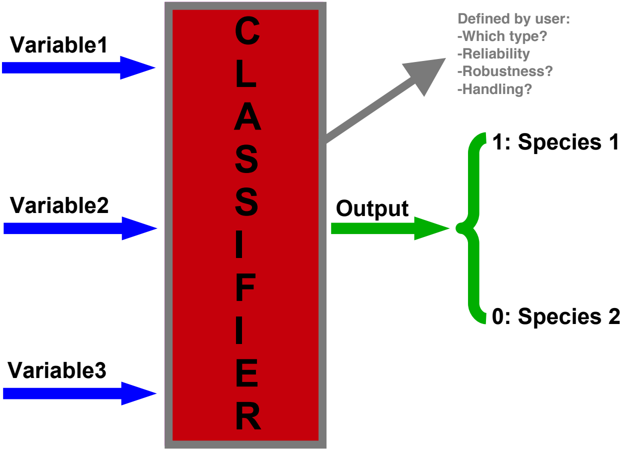
- Classifier is defined by internal parameters (e.g. cut threshold, weights,..)
- Possible classifiers are: Boosted Decision Trees, Neural Networks, Support Vector Machines, Nearest Neighbour Finders,...

Idea of a Classifier



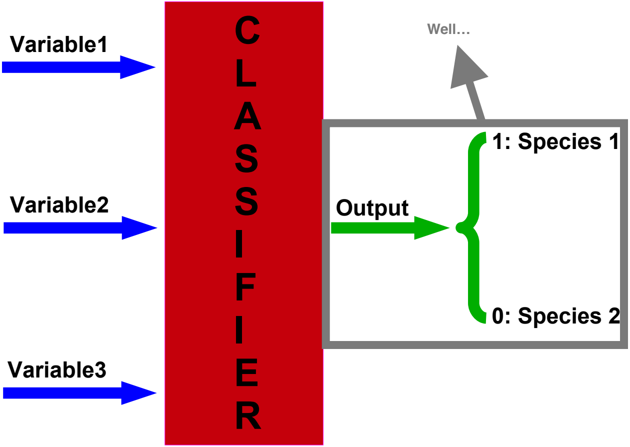
- Classifier is defined by internal parameters (e.g. cut threshold, weights,..)
- Possible classifiers are: Boosted Decision Trees, Neural Networks, Support Vector Machines, Nearest Neighbour Finders,...

Idea of a Classifier



- Classifier is defined by internal parameters (e.g. cut threshold, weights,..)
- Possible classifiers are: Boosted Decision Trees, Neural Networks, Support Vector Machines, Nearest Neighbour Finders,...

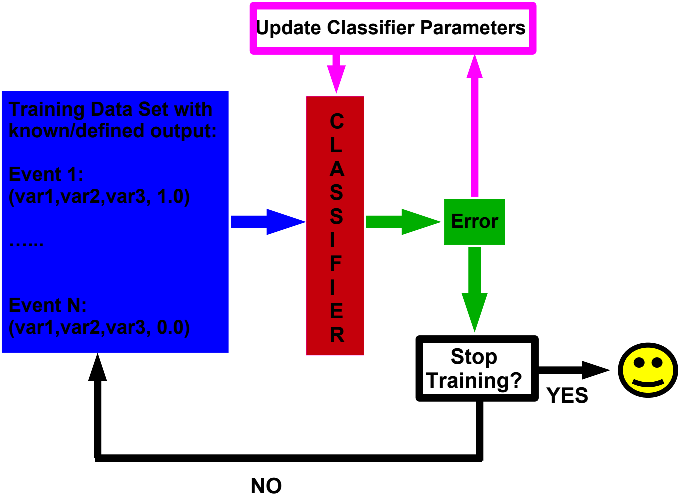
Idea of a Classifier



- Classifier is defined by internal parameters (e.g. cut threshold, weights,..)
- Possible classifiers are: Boosted Decision Trees, Neural Networks, Support Vector Machines, Nearest Neighbour Finders,...

Training the Classifier (Machine Learning)

- Parameters of the classifier are estimated/tuned by a training data set with defined output
- Usually: Use **Error** $= f($**classifier output**, **desired output in training data set**$)$ to iteratively update classifier parameters
- Several classifier-packages with learning algorithms available

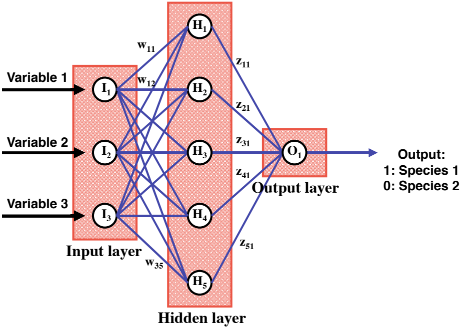


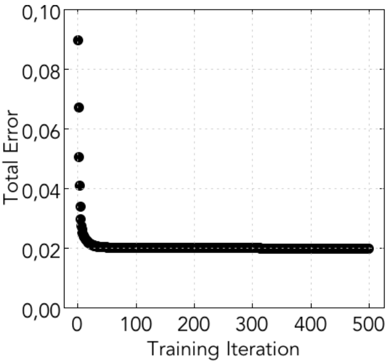
Available Frameworks/Packages

1. The ROOT TMVA-Package https://root.cern.ch/tmva
   - Designed for multivariable analysis
   - Use different classifier types for one data set in one script
   - Handling and preparation of input data
   - Dedicated monitoring of classifier outputs and variable dependencies
2. The SMILE-Package http://haifengl.github.io/smile/
   - Statistical Machine Intelligence and LEarning
   - Useable in Java, Scala or any JVM language
   - Several classifiers and learning algorithms available
   - GUI available
3. **The Apache Spark-Package** https://spark.apache.org/
   - Handling and manipulating of large data sets
   - Dedicated machine learning libraries
   - World wide community (companies, developers)
   - Quite similar to TMVA
4. **The Neuroph-Package** http://neuroph.sourceforge.net/
   - Java Neural Network Framework
   - Different sorts of neural nets available
     (Perceptron $\Leftrightarrow$ classifier, Hopefield $\Leftrightarrow$ image recognition, Kohonen $\Leftrightarrow$ Data-Mining)
   - Easy to use, high flexibility and modularity
   - Train your own net with a GUI $\rightarrow$ It is a lot of fun!

- No. 2 to 4 are not physics analysis related $\Rightarrow$ Used by wider machine learning community
- Examples shown in this talk are related to packages 3 and 4

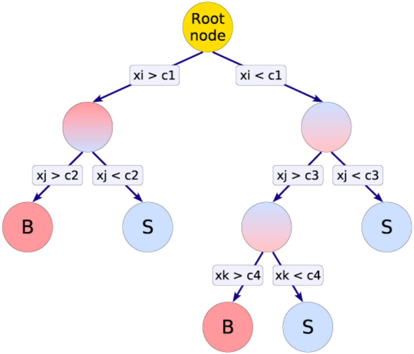
Classifier1: A Neural Network





- Internal Parameters: Weights w_{ij}, z_{ij}
- Training set is defined by: Variable 1,2,3
  and output 1 (for species 1), 0 (for species 2)
- Training iteratively done via back-propagation algorithm:
  Error $\propto$ [Output(Network) - Desired Output of training set]^2 is propagated backwards through the network to update weights
- Training curve is one (but not the only one) plot to monitor/check the behaviour of a classifier
- Weierstrass-Theorem $\Leftrightarrow$ Network Architecture

Classifier2: A boosted Decision Tree



- Internal Parameters: Thresholds c_i and weights
- Use same training set as for neural network
- Adjust cut parameters c_i with respect to maximum separation between (B)ackground and (S)ignal
- Define weights for each event according to misidentification error (boosting)
  $\Rightarrow$ New tree with updated cut parameters c_i
- Iteratively repeat procedure until misidentification error is minimal
  $\Rightarrow$ Forest of trees

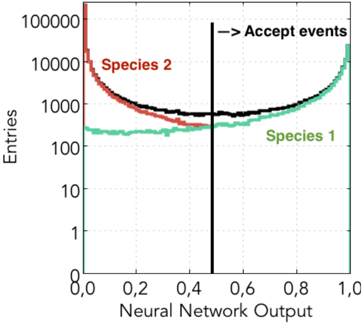
Output and Performance of the Network/Classifier

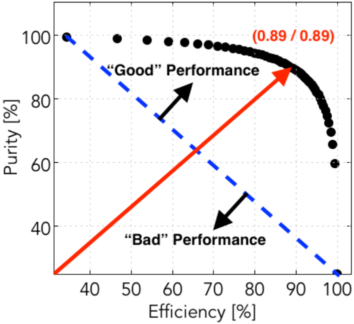
- Further plots[1] to characterise a classifier: output variable and purity vs efficiency

⇒ Efficiency: $\frac{\#(\text{events correctly identified as species 1})}{\#(\text{all events with species 1})}$

⇒ Purity: $\frac{\#(\text{events correctly identified as species 1})}{\#(\text{events identified as species 1})}$

- Here: Use a cut at 46% with: efficiency = 89% and purity = 89%
- Plot on the bottom right: Reciever-Operating-Characteristic (ROC) Curve[2]
- Apache Spark: Output of the classifier is directly translated to 1: signal and 0: background ⇔ According to roc-curve

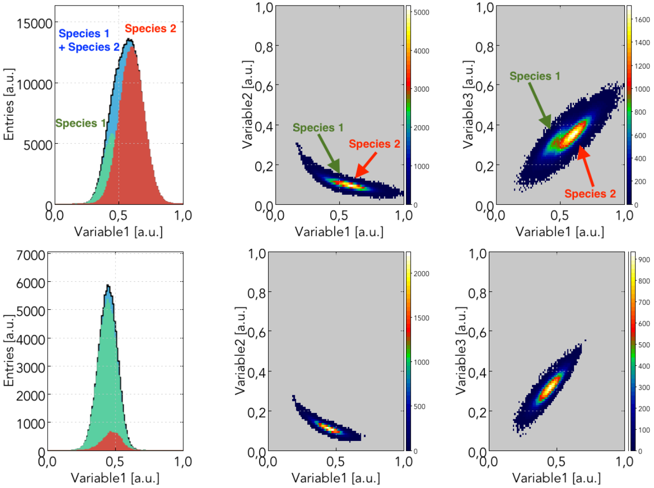




[1] Plots have been generated using the Neuroph-package and x-checked with Apache Spark

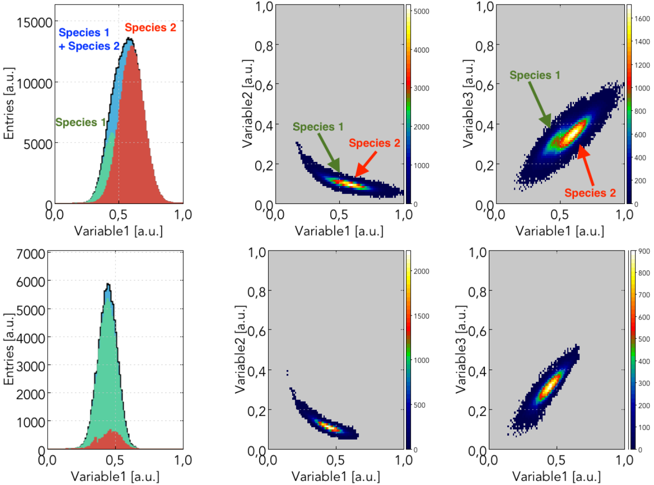
[2] See also: Data Analysis in High Energy Physics, O.Behnke et. al

Results after using an Apache Spark Neural Network



- Top Row: Before classification / Bottom Row: After classification
  (similar results observed using the Neuroph-package)
- Reconstructed* ratio (Signal / Background): $R = 1 \ / \ 3.014$
  $\star$ Including background

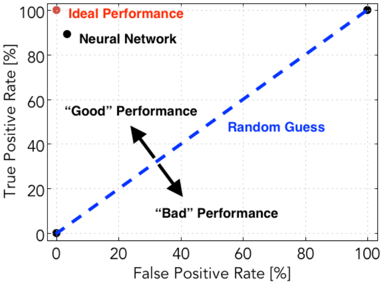
Results after using an Apache Spark Boosted Decision Tree

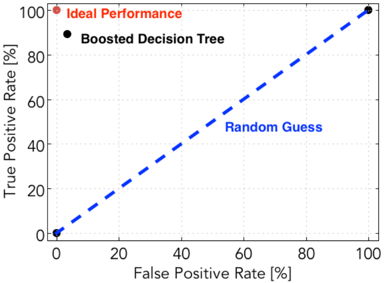


- Top Row: Before classification / Bottom Row: After classification
- Reconstructed* ratio (Signal / Background): $R = 1 / 2.959$

$\star$ Including background

Comparing Classifier and their Performance





- Apache Spark has built in tools (e.g. roc-curve) to analyse/judge the quality/performance
- Roc-curve shown here/used* in Apache Spark:

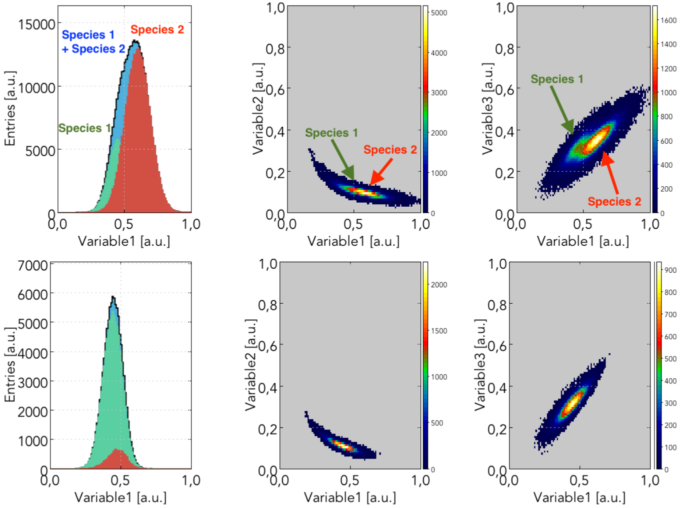
  * Definitions and Plots here: https://en.wikipedia.org/wiki/Receiver_operating_characteristic

  True Positive Rate (TPR) (how many signal events are identified as signal?) vs.
  False Positive Rate (FPR) (how many background events are identified as signal?)
- Also available: Area under roc-curve, purity, accuracy,..

Classifier	Efficiency [%]	Purity [%]	TPR [%]	FPR [%]
NN	89	89	89	3.5
BDT	89	88	89	3.5

⇒ **Both classifier show same performance**

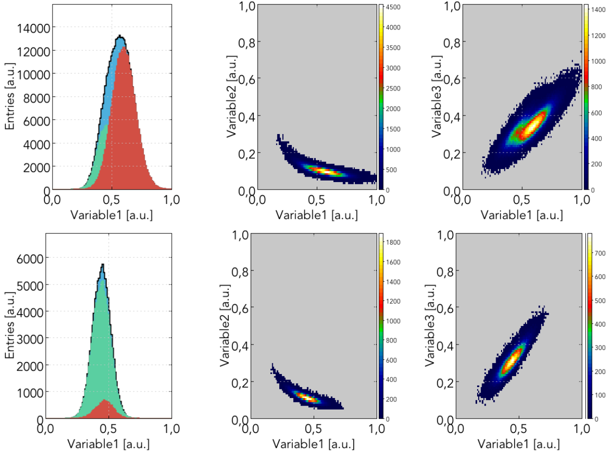
Uncertainties and Errors: Response to untrained Features



- Checked performance of the classifier on well-known training/test data set (see above)
- How does Classifier respond to unknown data regions/features it was not trained to?
- Possible Check/Test: Apply uncertainty to test data set and check classifiers response

Uncertainties and Errors: Response to untrained Features

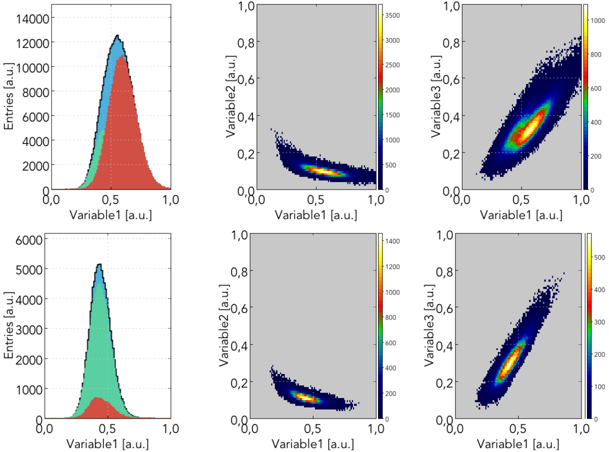
Uncertainty = 5%



- Example here: Use neural network shown on slide 6
  (Did the same study with the boosted decision tree)
- Top Row: Before classification / Bottom Row: After classification
- Apply uncertainty on variable 1,2 and 3 ⇒ What is the classifiers response?

Uncertainties and Errors: Response to untrained Features

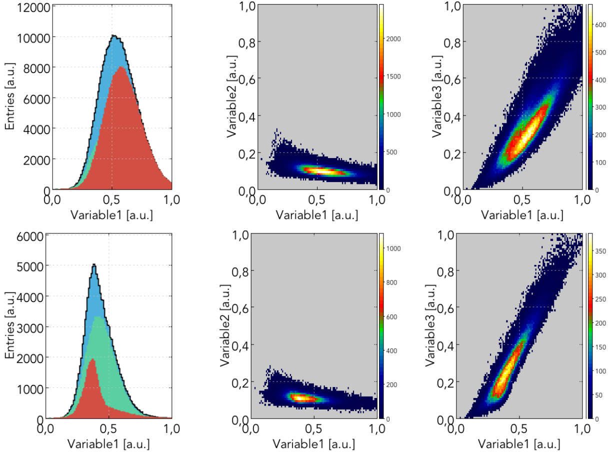
Uncertainty = 10%



- Example here: Use neural network shown on slide 6
  (Did the same study with the boosted decision tree)
- Top Row: Before classification / Bottom Row: After classification
- Apply uncertainty on variable 1,2 and 3 $\Rightarrow$ What is the classifiers response?

Uncertainties and Errors: Response to untrained Features

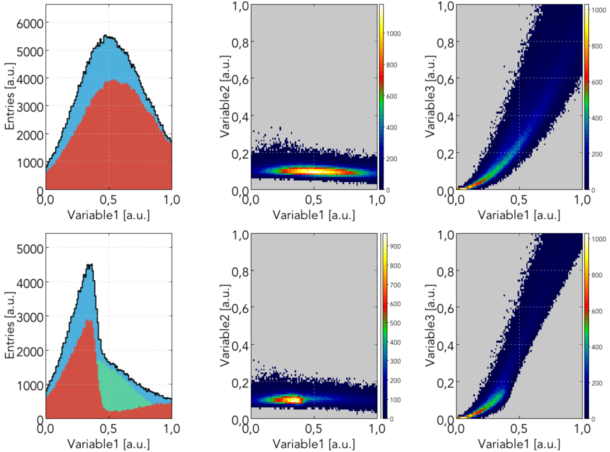
Uncertainty = 20%



- Example here: Use neural network shown on slide 6
  (Did the same study with the boosted decision tree)
- Top Row: Before classification / Bottom Row: After classification
- Apply uncertainty on variable 1,2 and 3 ⇒ What is the classifiers response?

Uncertainties and Errors: Response to untrained Features

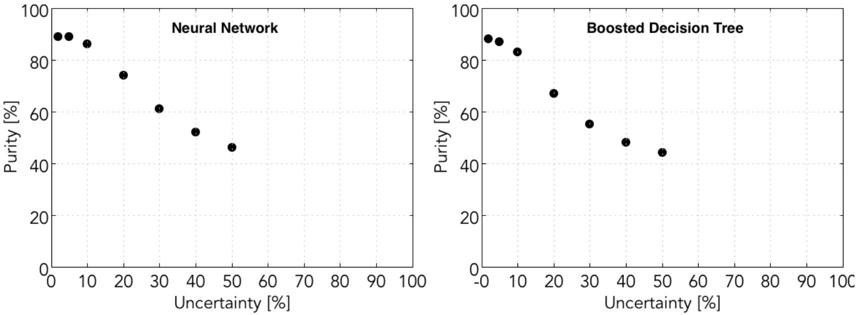
Uncertainty = 50%



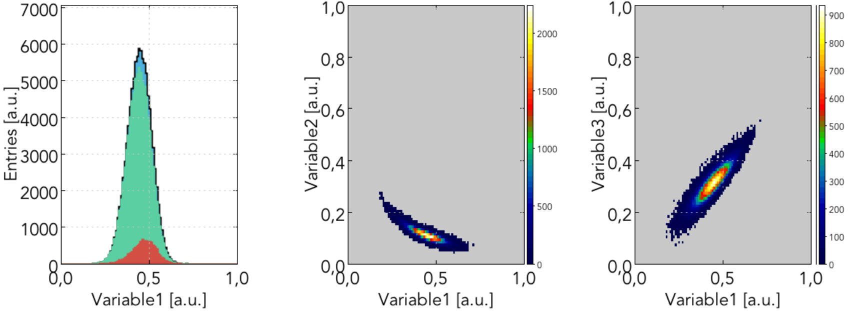
- Example here: Use neural network shown on slide 6
  (Did the same study with the boosted decision tree)
- Top Row: Before classification / Bottom Row: After classification
- Apply uncertainty on variable 1,2 and 3 $\Rightarrow$ What is the classifiers response?

Uncertainties and Errors: Response to untrained Features

- Monitor Purity as a function of the uncertainty
- Both classifier show similar behaviour/response
- If uncertainty/disagreement between training data and actual data is known
  $\Rightarrow$ Get rough estimate for accuracy of classifier
- Or: If certain accuracy is required (precision measurement)
  $\Rightarrow$ To which level is tuning between training data and actual data needed/possible?

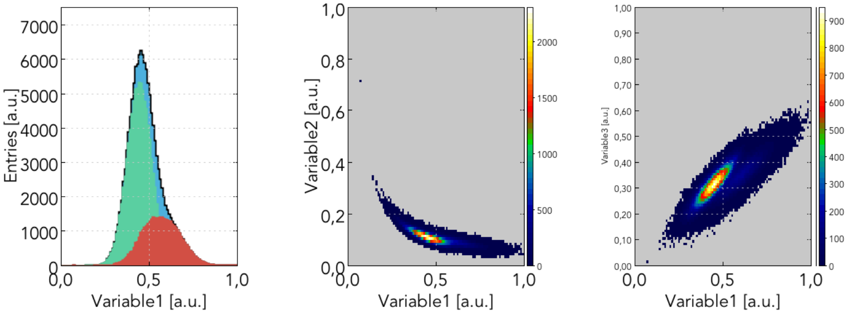


Scaling of Background-Events



- Up to now: Classifier output $= \begin{cases} 1 : \text{Accept Event} \\ 0 : \text{Reject Event} \end{cases}$
- Effect on final data sample:
  - Very "clean" - Species1 is dominating
  - Background and signal shape are identical
    $\Rightarrow$ Might need another (independent) observable for further analysis
    (e.g estimation of R)
  - Loss of information $\Leftrightarrow$ Depending on at which stage of analysis the classifier is used

Scaling of Background-Events



- Alternative: Classifier output $= \begin{cases} 1 : \text{Accept Event} \\ 0 : \text{Accept every 10th Event} \end{cases}$
- Effect on final data sample:
  - Quite "clean" - Species1 is still dominating
  - Background and signal shape are less identical
    $\Rightarrow$ Use tail to estimate background contribution
  - Larger background sample available $\Leftrightarrow$ Systematic x-checks

Data Flow and Analysis Chain

INPUT

- Decisive Power
- Additional preparation needed ?(e.g. normalisation)
- How strong correlated?
- Use measured data or MC?
- Avoid bias
- Impact/Importance on Classifier performance?

→ **Know detector**
→ **Calibration**
→ **Match between data/MC**

ANALYSIS

OUTPUT

- How used in further Analysis?
- Used at which analysis Stage?
- Assigned error?
- Trustworthy?

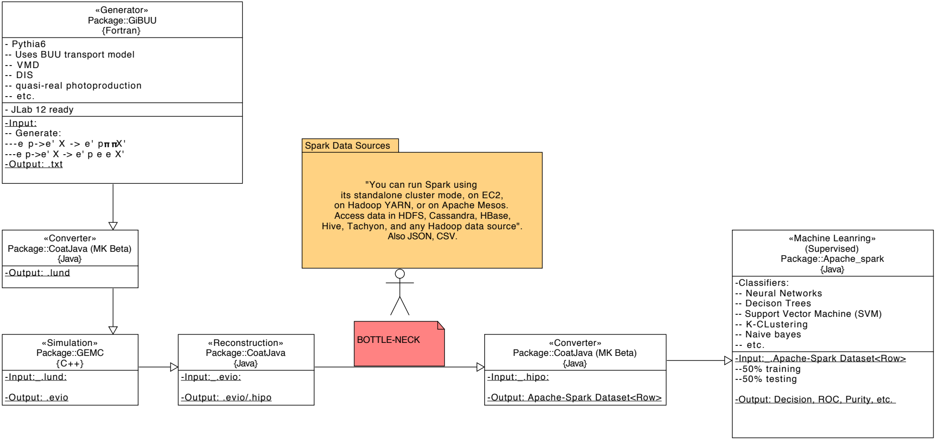
→ **Systematic studies**
→ **Error handling**

CLASSIFIER

- Which type?
- How to train?
- Implementaion / Handling?
- Influence on systematics?
- Response to unknown data?
- Reliability?
- Optimization (i.e. Tuning the classifier)

→ **Training curve**
→ **ROC plot**
→ **Monitoring plots**
→ **Output variable**
→ **Use dedicated frameworks**
→ **Do not reinvent the wheel**

Data Flow and Analysis Chain: Application of Apache Spark on CLAS12 Data



Work done by / figure taken from Michael C. Kunkel

Summary and Outlook

- Performed machine learning based analysis within the Apache Spark framework:
  - Identified a (particle) species out of a background dominated (fake) data set
  - Used a Neural Network and a Boosted Decision Tree
    $\Rightarrow$ both performed similar
  - Checked performance/accuracy of each classifier under various conditions
- Analysis chain and data processing setup for CLAS12 PID with Apache Spark (Thanks to Michael C. Kunkel)
- $\Rightarrow$ Start implementation of classification algorithms
- Need:
  - i) Dedicated people to work on implementation and usage of classifiers
    $\Rightarrow$ Set up/establish machine learning group
  - ii) Work closely with detector calibration and simulation subgroups
  - iii) Always an additional pair of eyes
- Other frameworks (e.g. Neuroph-Package) available $\Leftrightarrow$ X-checks ?
- Use machine learning algorithms also for:
  - Regression - Fitting of data
  - Parameter estimation
  - Pattern recognition
- For further reading: https://www.jlab.org/indico/event/213/session/6/contribution/23/material/slides/0.pdf (Talk by Michael Williams)

Summary and Outlook

You just have to know what you are doing

Picture taken from: http://screenrant.com/things-you-did-not-know-about-wile-e-coyote/