CERN School of Computing

Multivariate Classification

Thomas Keck
thomas.keck2@kit.edu

KIT

CERN School of Computing 2016

Observation of a new boson at a mass of 125 GeV with the CMS experiment at the LHC

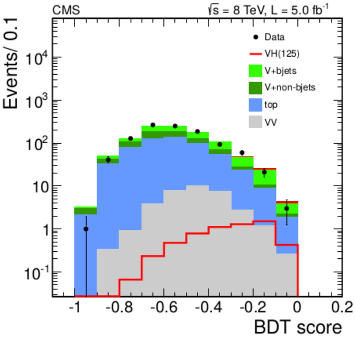


Figure 11: Distribution of BDT scores for the high-p_T subchannel of the $Z(\nu\nu)H(bb)$ search in the 8 TeV data set after all selection criteria have been applied. The signal expected from a Higgs boson ($m_H = 125\,\mathrm{GeV}$), including $W(\ell\nu)H$ events where the charged lepton is not reconstructed, is shown added to the background and also overlaid for comparison with the diboson background.

CERN School of Computing

Observation of a new boson at a mass of 125 GeV with the CMS experiment at the LHC

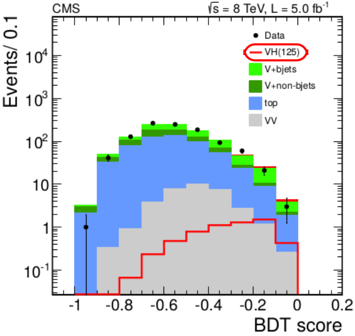


Figure 11: Distribution of BDT scores for the high-p_T subchannel of the Z($\nu\nu$)H(bb) search in the 8 TeV data set after all selection criteria have been applied. The signal expected from a Higgs boson (m_H = 125 GeV), including W($\ell\nu$)H events where the charged lepton is not reconstructed, is shown added to the background and also overlaid for comparison with the diboson background.

CERN School of Computing

Observation of a new boson at a mass of 125 GeV with the CMS experiment at the LHC

arXiv:1207.7235v2 [hep-ex] 28 Jan 2013

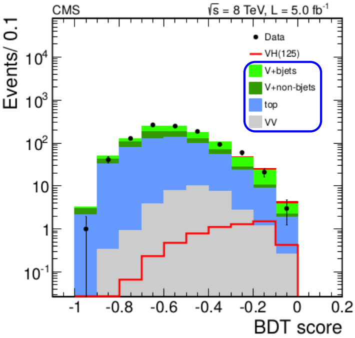


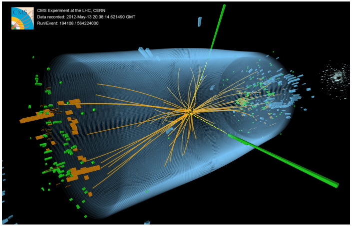
Figure 11: Distribution of BDT scores for the high-p_T subchannel of the $Z(\nu\nu)H(bb)$ search in the 8 TeV data set after all selection criteria have been applied. The signal expected from a Higgs boson ($m_H = 125$ GeV), including $W(\ell\nu)H$ events where the charged lepton is not reconstructed, is shown added to the background and also overlaid for comparison with the diboson background.

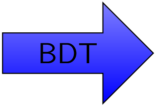
CERN School of Computing

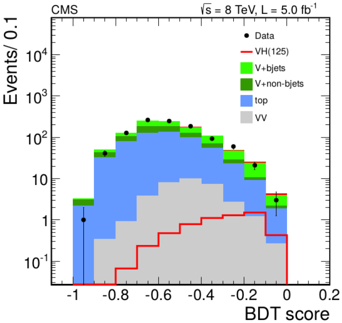
For the multivariate analysis, a boosted decision tree (BDT) [119, 120] is trained to give a high output value (score) for signal-like events and for events with good diphoton invariant mass resolution, based on the following observables: (i) the photon quality determined from electromagnetic shower shape and isolation variables; (ii) the expected mass resolution; (iii) the per-event estimate of the probability of locating the diphoton vertex within 10 mm of its true location along the beam direction; and (iv) kinematic characteristics of the photons and the diphoton system. The kinematic variables are constructed so as to contain no information about the invariant mass of the diphoton system. The diphoton events not satisfying the dijet selection are classified into five categories based on the output of the BDT, with category boundaries optimized for sensitivity to a SM Higgs boson. Events in the category with smallest expected signal-to-background ratio are rejected, leaving four categories of events. Dijet-tagged events with BDT scores smaller than the threshold for the fourth category are also rejected. Simulation studies indicate that the background in the selected event categories is dominated by the irreducible background from QCD production of two photons and that fewer than 30% of the diphoton events used in the analysis contain one or more misidentified photons (predominantly from γ+jet production).

For the multivariate analysis, a boosted decision tree (BDT) [119, 120] is trained to give a high output value (score) for signal-like events and for events with good diphoton invariant mass resolution, based on the following observables:

For the multivariate analysis, a boosted decision tree (BDT) [119, 120] is trained to give a high output value (score) for signal-like events and for events with good diphoton invariant mass resolution, based on the following observables:







CERN School of Computing

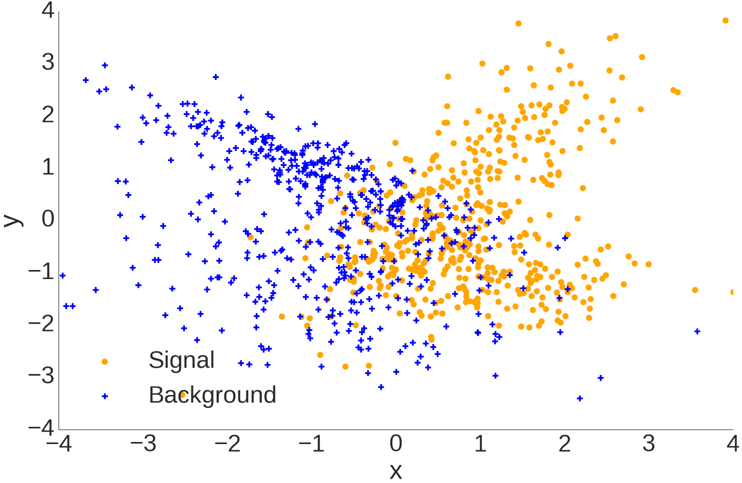
Outline

CERN School of Computing

Neyman-Pearson Lemma / Supervised learning

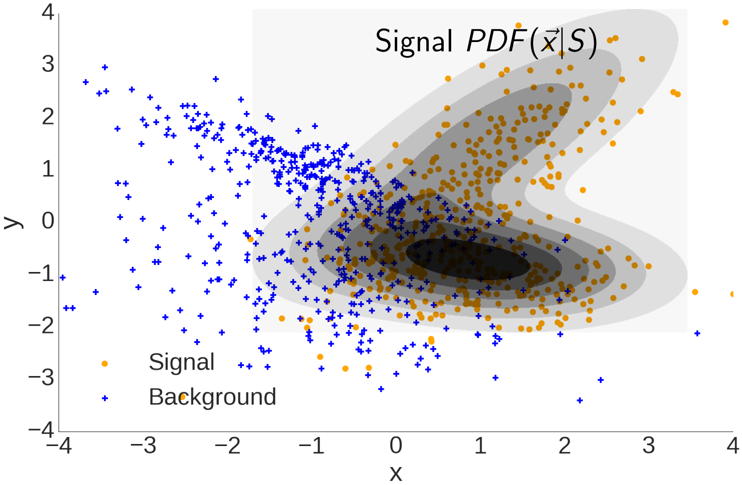
CERN School of Computing

Simple example in two dimensions



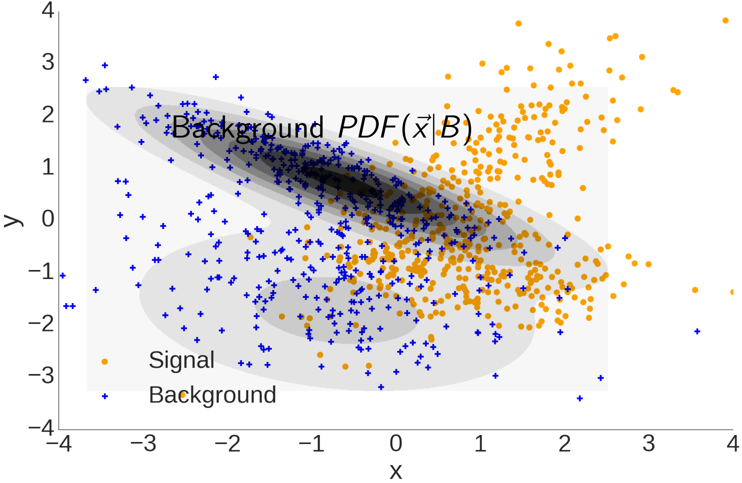
CERN School of Computing

Simple example in two dimensions



CERN School of Computing

Simple example in two dimensions

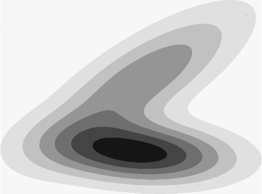


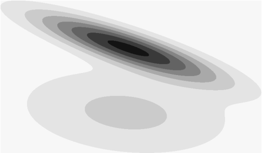
Neyman-Pearson Lemma

IX. *On the Problem of the most Efficient Tests of Statistical Hypotheses.*

By J. NEYMAN, *Nencki Institute, Soc. Sci. Lit. Varsoviensis, and Lecturer at the Central College of Agriculture, Warsaw, and* E. S. PEARSON, *Department of Applied Statistics, University College, London.*

$$f(\vec{x}) = \frac{PDF(\vec{x}|S)}{PDF(\vec{x}|B)} =$$

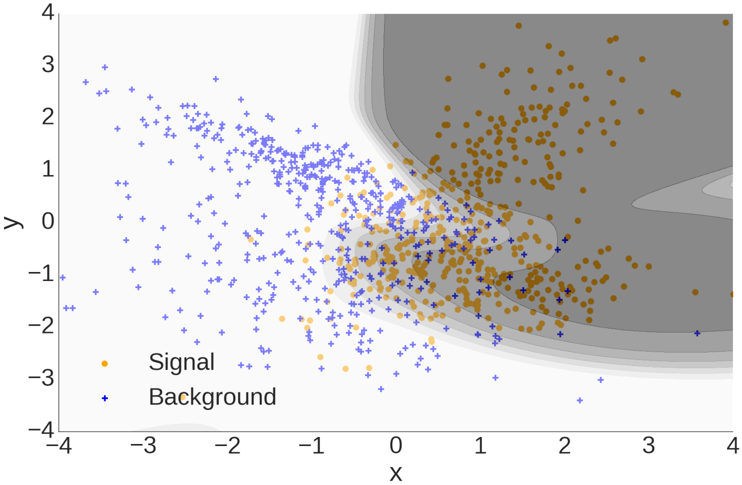




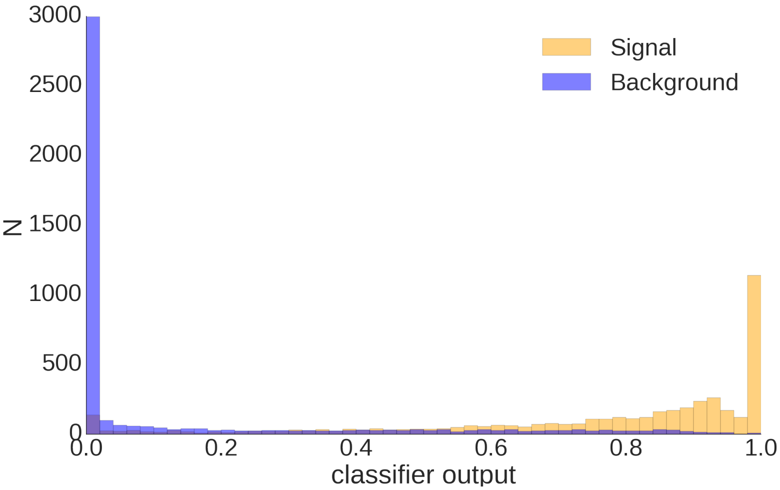
Most powerful test at a given significance level to distinguish between two simple hypotheses (signal or background)

CERN School of Computing

Neyman-Pearson Lemma



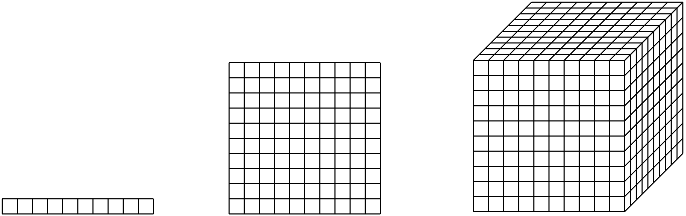
Neyman-Pearson Lemma



CERN School of Computing

Problem solved? No!

- Howto obtain the signal and background PDFs?
  - Usually unknown!
  - Multiple sources of signal and background
  - Non gaussian PDF
  - Nonlinear dependencies among observables
  - Cannot be sampled in high dimensions (e.g. cannot fill 80-dimensional histogram with enough statistics)
  - $\rightarrow$,,Curse of dimensionality"



CERN School of Computing

Solution: Approximate Neyman-Pearson

- Neyman-Pearson Lemma

$$f(\vec{x}) = \frac{PDF(\vec{x}|S)}{PDF(\vec{x}|B)}$$

- Generative Models
  - Analytical approx. (LDA, QDA)
  - Restricted Boltzmann machine
  - Kernel density estimator
  - Gaussian mixture model

$$f(\vec{x}|S) \approx PDF(\vec{x}|S)$$
$$f(\vec{x}|B) \approx PDF(\vec{x}|B)$$

- Discriminative Models
  - (Boosted) Decision Trees
  - Support Vector Machines
  - Artificial Neural Networks

$$f(\vec{x}) \approx \frac{PDF(\vec{x}|S)}{PDF(\vec{x}|B)}$$

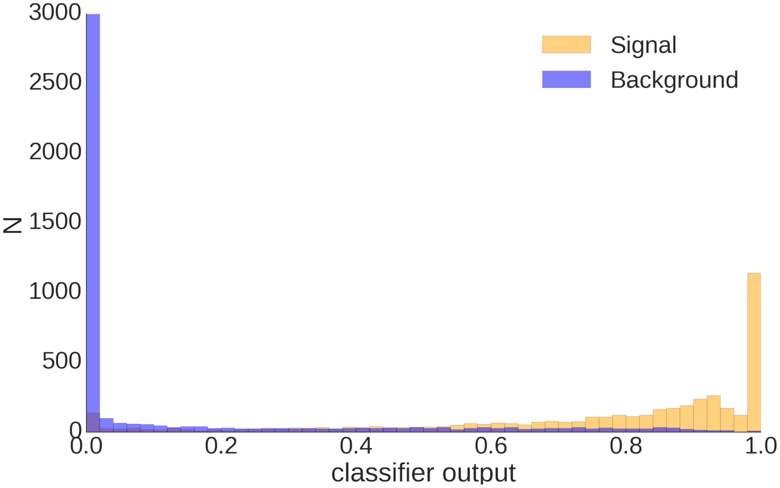
CERN School of Computing

More questions

- Howto obtain training data required for these models?
  - In Industry usually historical data ($\rightarrow$ time-series)
  - In HEP usually simulated data ($\rightarrow$ systematics)
  - Sometimes we can use real data ($\rightarrow$ data-driven techniques)
- Howto train, optimize and evaluate the quality of the models and compare them?
  - Train model on training data ($\rightarrow$ regularization techniques)
  - Optimize model on validation data ($\rightarrow$ hyper-parameter optimization)
  - Test model on test data ($\rightarrow$ ROC curves)
  - Apply model on unlabeled data ($\rightarrow$ systematics)

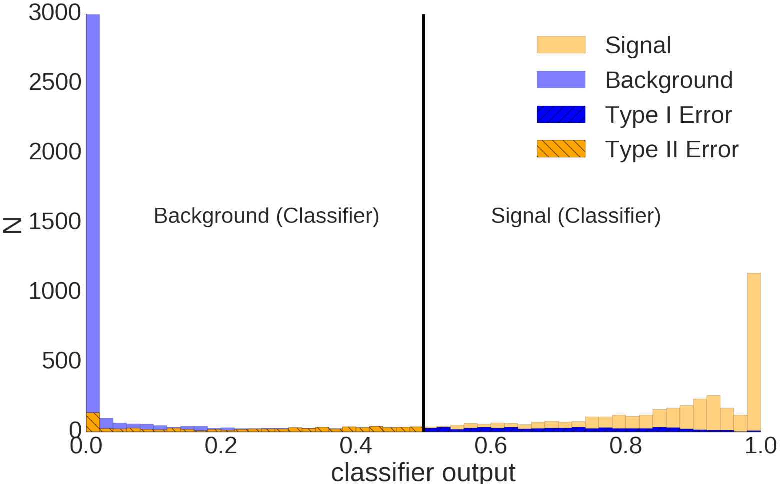
CERN School of Computing

Classification Quality

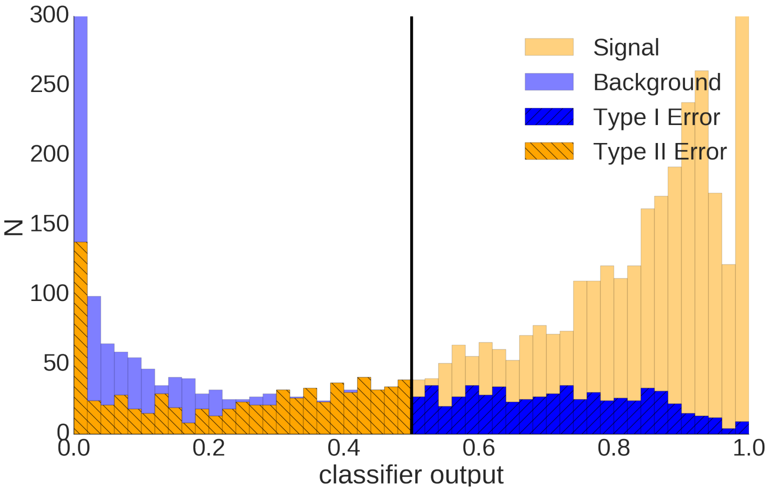


CERN School of Computing

Classification Quality

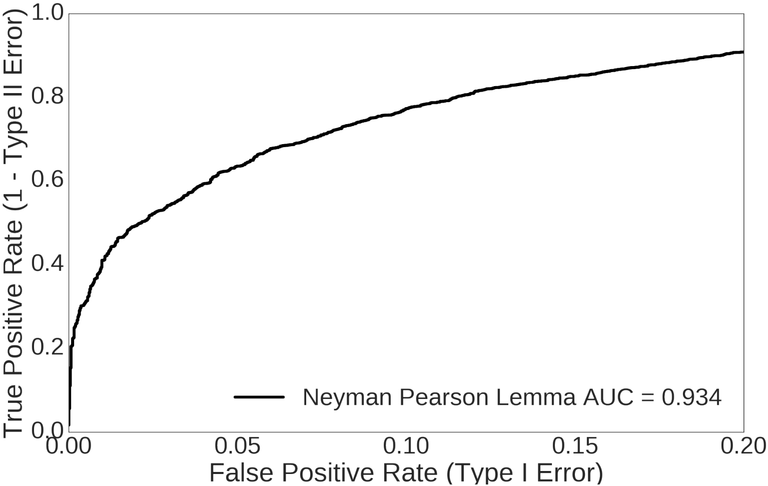


Classification Quality

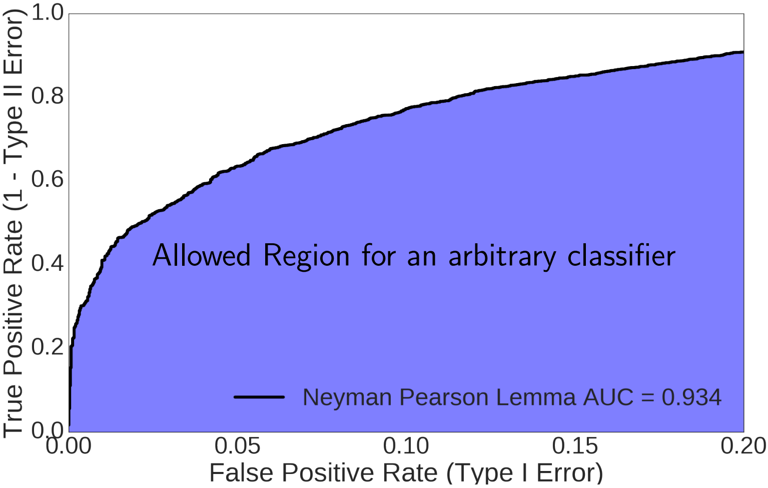


CERN School of Computing

Classification Quality



Classification Quality

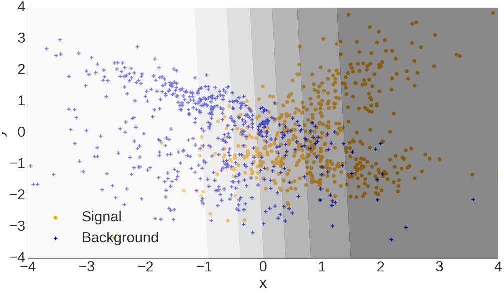


Discriminant Analysis / Analytical solutions

CERN School of Computing

Linear discriminant analysis

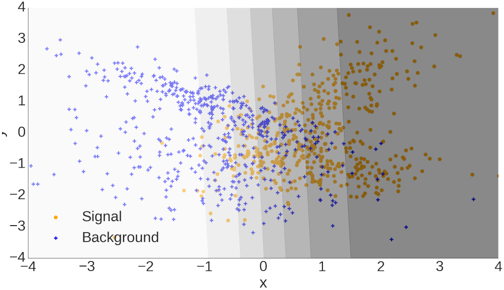
- Assumes conditional PDFs are normally distributed
- Assumes identical covariances of signal and background
- Equivalent to commonly used Fisher's discriminant
- Requires only means and covariances of sample
- Separating hyperplane is linear



CERN School of Computing

Linear discriminant analysis

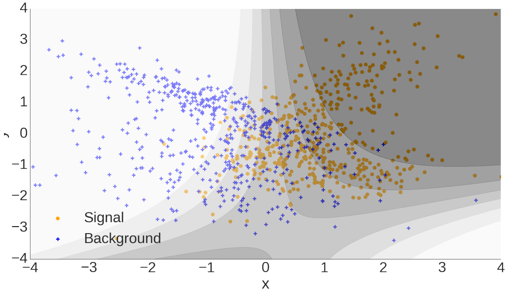
- Assumes conditional PDFs are normally distributed
- **Assumes identical covariances of signal and background**
- Equivalent to commonly used Fisher's discriminant
- Requires only means and covariances of sample
- Separating hyperplane is linear



Quadratic discriminant analysis

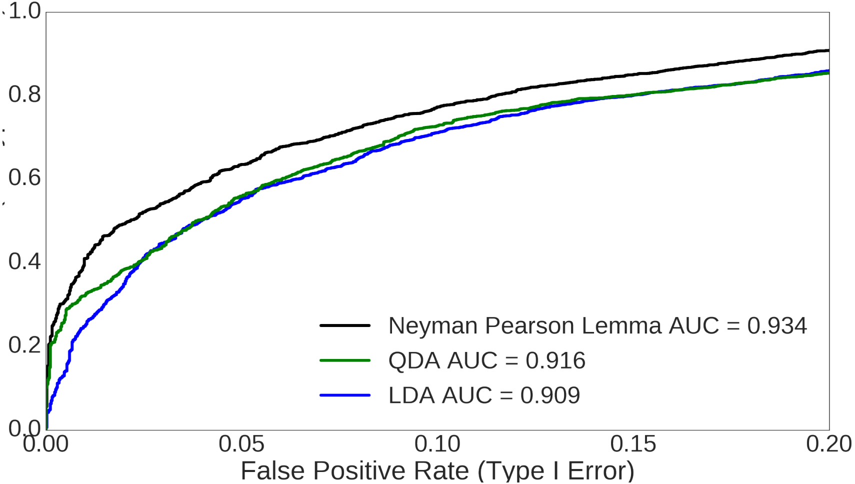
- Assumes conditional PDFs are normally distributed
- Requires only means μ_y and covariances Σ_y of sample
- Separating hyperplane is quadratic

$$f(\vec{x}) = \frac{\sqrt{2\pi|\Sigma_{y=0}|}\exp\left(-\frac{1}{2}\left(x-\mu_{y=1}\right)^T\Sigma_{y=1}\left(x-\mu_{y=1}\right)\right)}{\sqrt{2\pi|\Sigma_{y=1}|}\exp\left(-\frac{1}{2}\left(x-\mu_{y=0}\right)^T\Sigma_{y=0}\left(x-\mu_{y=0}\right)\right)}$$



CERN School of Computing

Example Classifier Quality

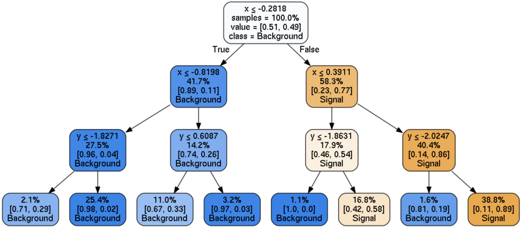


Decision Tree / Model complexity

CERN School of Computing

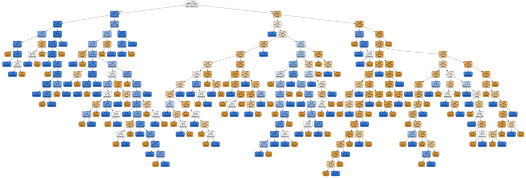
Simple Decision Tree

- Classifies using a number of consecutive rectangular cuts
- Each cut locally maximizes a separation gain measure
- Signal probability given by the purity in each leave
- Interpretable (white box) model

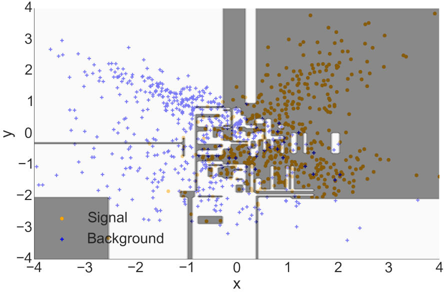


CERN School of Computing

Complete Decision Tree



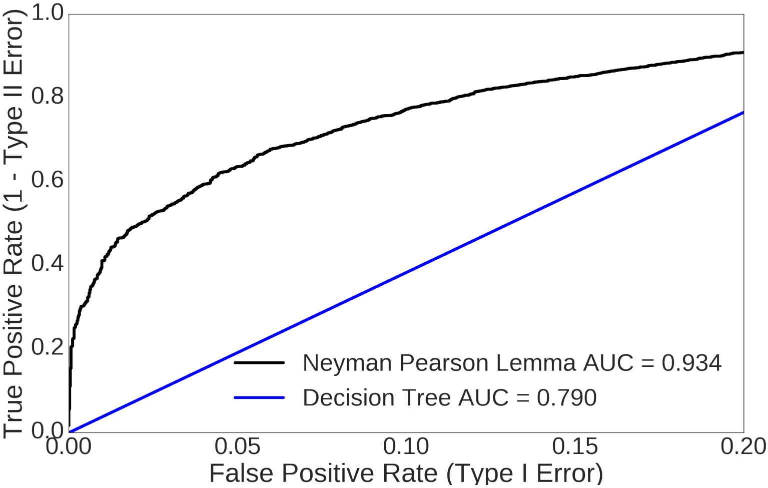
Complete tree with pure leaves $\rightarrow$ Complex model



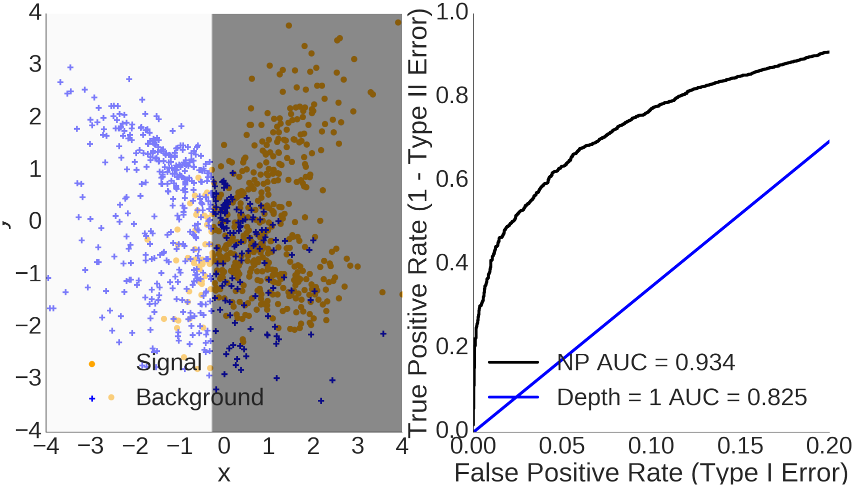
CERN School of Computing

Complete Decision Tree

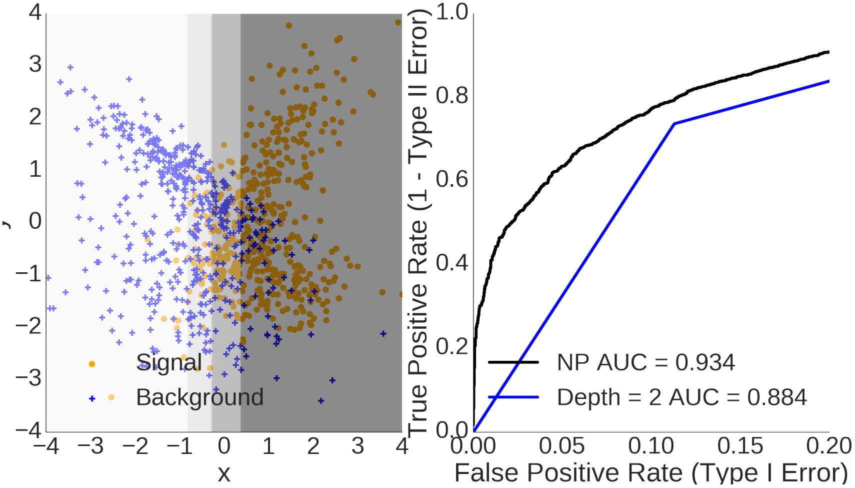
Complex model performs poorly due to **overfitting**



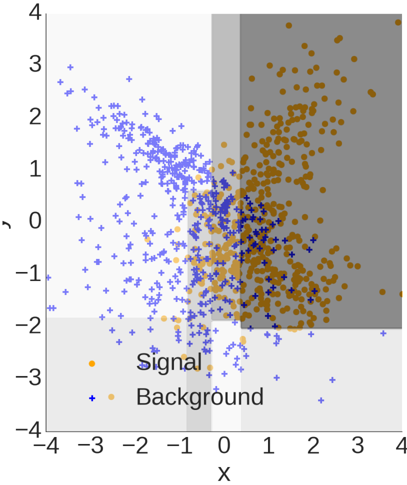
Depth dependency of decision trees

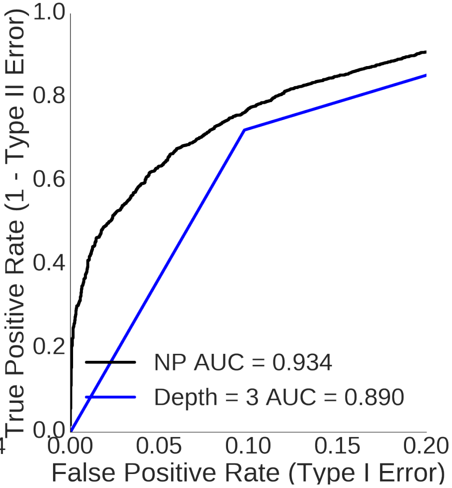


Depth dependency of decision trees



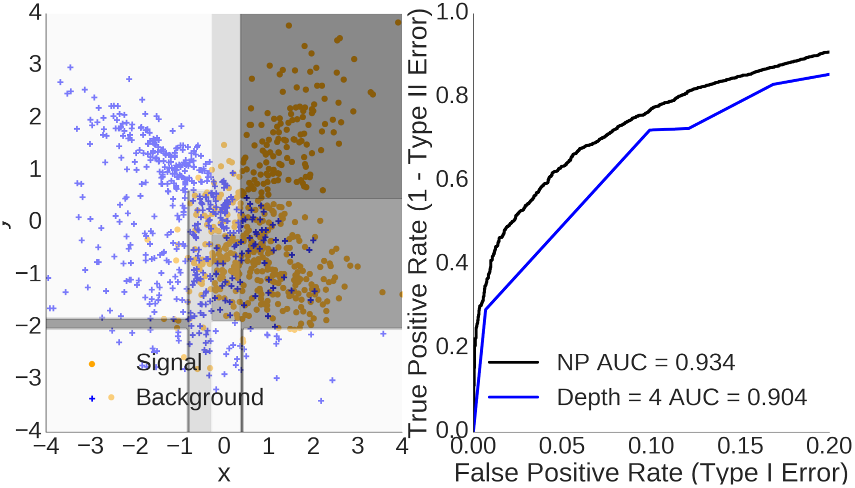
Depth dependency of decision trees



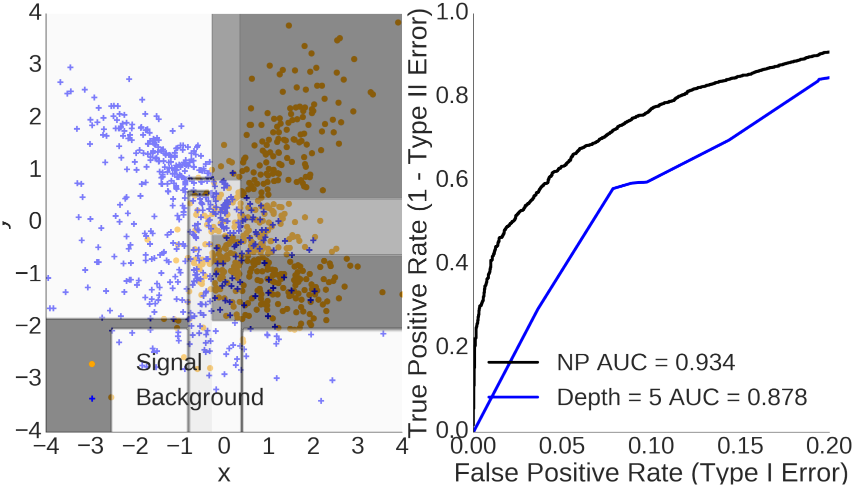


CERN School of Computing

Depth dependency of decision trees

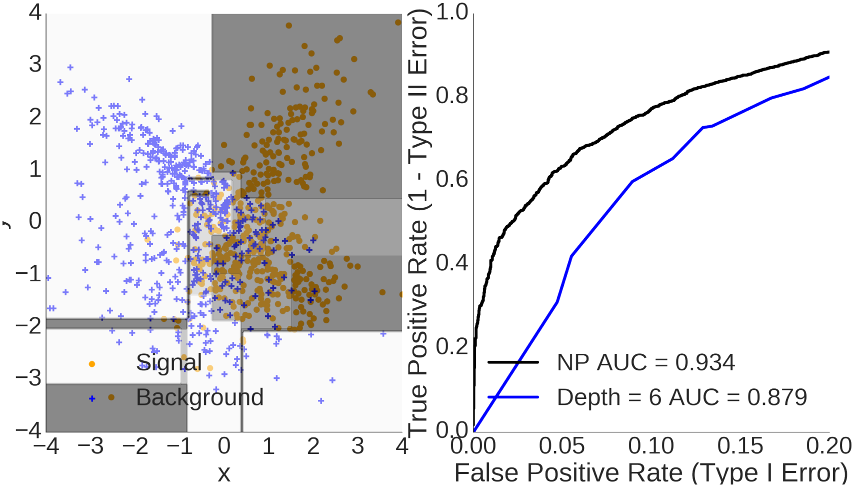


Depth dependency of decision trees



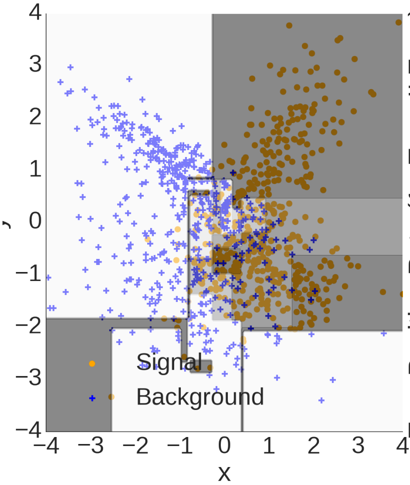
CERN School of Computing

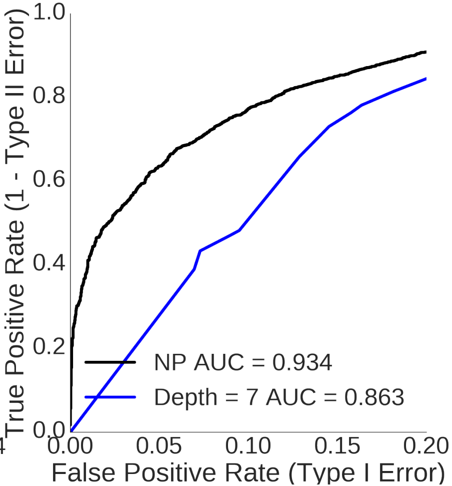
Depth dependency of decision trees



CERN School of Computing

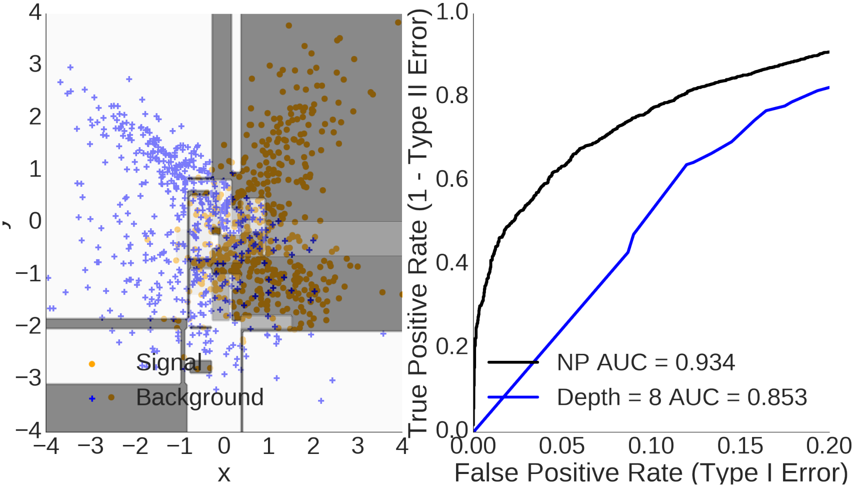
Depth dependency of decision trees





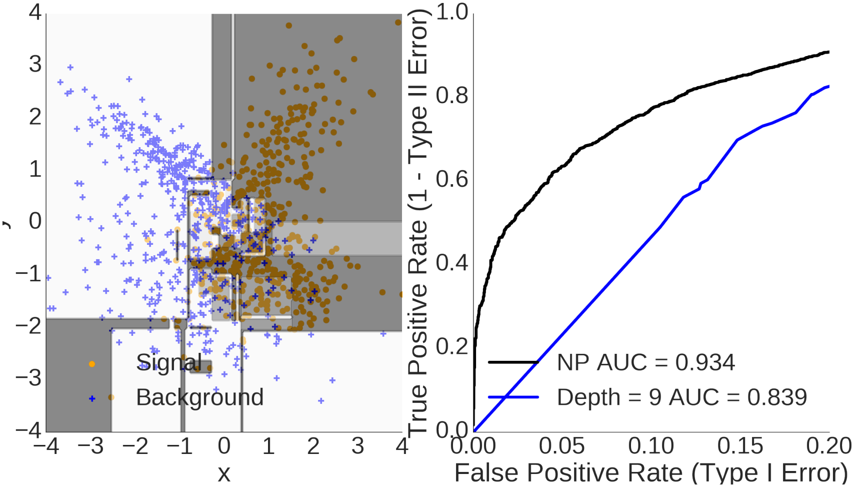
CERN School of Computing

Depth dependency of decision trees



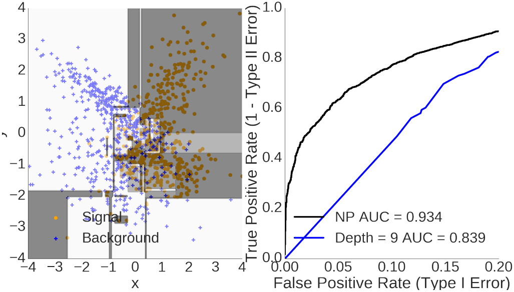
CERN School of Computing

Depth dependency of decision trees



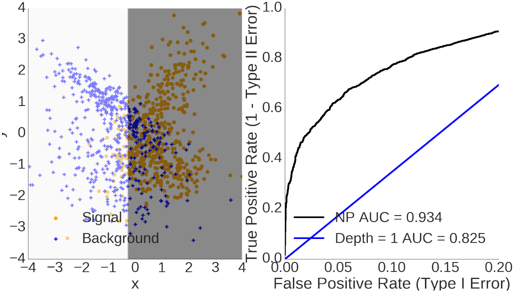
Overfitting

- Model is too complex
- Statistical fluctuations in the training data dominate predictions
- Model does not generalize → poor performance on new data
- Need to check for this on an independent test dataset!



Underfitting

- Model is too simple
- Relevant aspects of the data are ignored

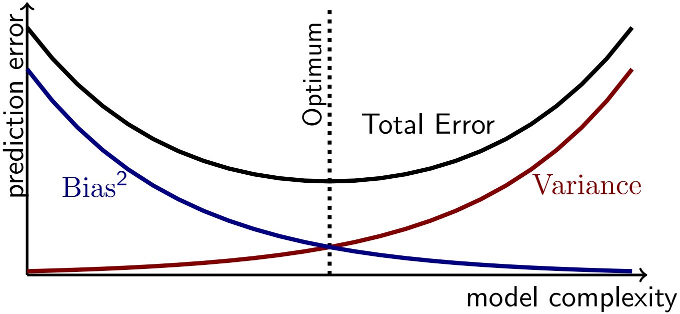


Bias-Variance dilemma

Three sources of errors:

- Bias due to wrong modeling of the data (underfitting)
- Variance due to sensitivity to statistical fluctuations (overfitting)
- Irreducible error due to noise in the problem itself

$$\mathrm{E}\left[(y - \widehat{f}(\vec{x}))^2\right] = \mathrm{Bias}\left[\widehat{f}(\vec{x})\right]^2 + \mathrm{Var}\left[\widehat{f}(\vec{x})\right] + \mathrm{Var}\left[y\right]$$



CERN School of Computing

Model complexity

Number of Degrees of freedom (NDF) of the model
($\approx$ number of parameters)

- Input dataset
  - Reduce dimensionality
  - Higher statistic
- Hyperparameters (control NDF)
  - Depth of the tree
  - Minimum number of data points per leave
  - Separation gain measure (entropy, gini-index)
  - Optimized using search-algorithm
- Regularization (reduce effective NDF)
  - Prune overfitted branches of the tree
  - Include tree structure in separation gain measure
  - Ensemble methods

Always test on an independent test dataset in the end!

CERN School of Computing

Boosted Decision Trees / Ensemble methods

Idea

Average many simple models to obtain a robust complex model

$$F(\vec{x}) = \sum_m \gamma_m f_m(\vec{x})$$

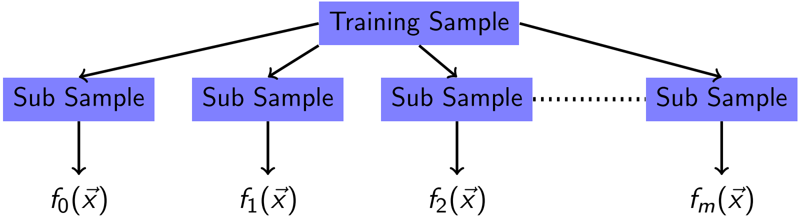
Boosting

$$f_m(\vec{x}) = f_{m-1} + \arg\min_f \sum_i^N L\left(y_i, f_{m-1}(\vec{x}_i) - f(\vec{x}_i)\right)$$

$$\text{Training Sample} \longrightarrow f_0(\vec{x})$$
$$\downarrow$$
$$\text{Weighted Sample} \longrightarrow f_1(\vec{x})$$
$$\downarrow$$
$$\text{Weighted Sample} \longrightarrow f_2(\vec{x})$$
$$\downarrow$$
$$\text{Weighted Sample} \longrightarrow f_3(\vec{x})$$
$$\vdots$$
$$\text{Weighted Sample} \longrightarrow f_m(\vec{x})$$

- Reweight events w.r.t current prediction
- Individual classifiers are simple to avoid overfitting (weak-learners)
- Focus on events near the optimal separation hyper-plane
- Loss function L is crucial
  - Least square $\rightarrow$ Regression
  - Binomial deviance $\rightarrow$ GradientBoost classification
  - Exponential loss $\rightarrow$ AdaBoost classification

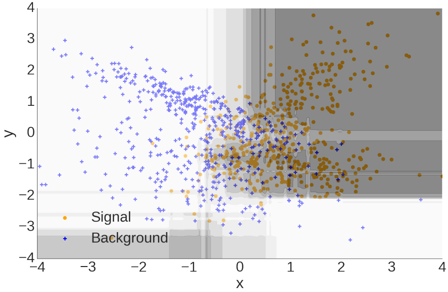
Bagging



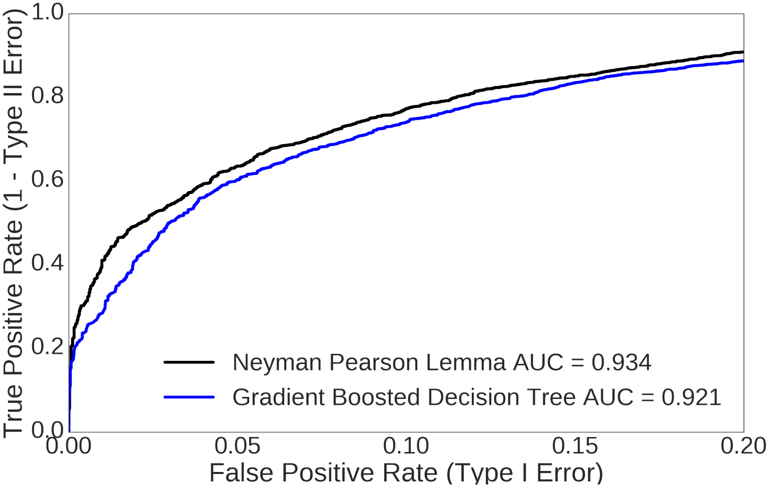
- Bagging – Use only fraction of events / features per classifier
- Robustness against statistical fluctuations in the data
- Embarrassingly parallel
- Sampling method is crucial:
    - Draw random events with replacements → Bagging
    - Draw random events without replacement → Pasting
    - Draw random features → Random Subspaces

Stochastic Boosted Decision Trees

- Good out-of-the-box performance
- Robust against over-fitting
- Supports classification and regression
- Widely used in HEP



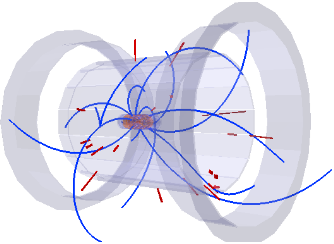
Example Classifier Quality



Further Ensemble Methods

Categorization

- Divide feature-space into sub-spaces
- Different behavior of the data in the chosen subspaces
- e.g. train separate classifier for Barrel and Endcap



Combination

- Combine different classifiers
- Different regularization methods learn different aspects of the data
- e.g. combine neural network, BDT and SVM

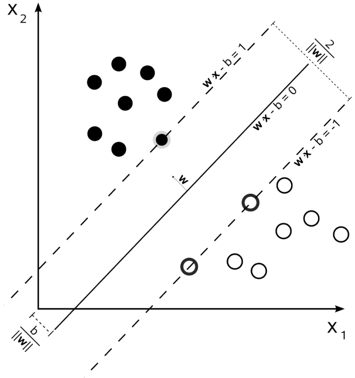
CERN School of Computing

Support Vector Machines / Kernel trick

CERN School of Computing

Support vector machines

- Maximum margin classifier
- Quadratic problem → can be solved efficiently in $O(N^2)$
- Optimal for linearly separable problems
- Slope variables allow for misclassification



CERN School of Computing

Kernel trick

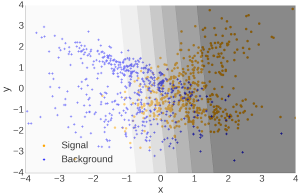
- SVM Algorithm depends only on scalar product!
- Replace scalar product with an arbitrary kernel function
- Solves problem in implicitly high-dimensional space

$$\max g(c_1, \dots, c_n) = \sum_i c_i - \frac{1}{2} \sum_i \sum_j y_i c_i (\vec{x}_i \cdot \vec{x}_j) y_j c_j$$

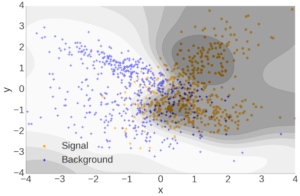
CERN School of Computing

Kernel trick

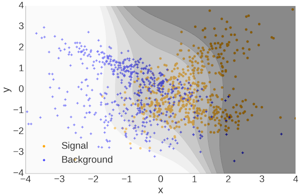
Linear $k(\vec{x}_i, \vec{x}_j) = \vec{x}_i \cdot \vec{x}_j$



Gausian $k(\vec{x}_i, \vec{x}_j) = \exp(-\gamma||\vec{x}_i - \vec{x}_j||^2)$

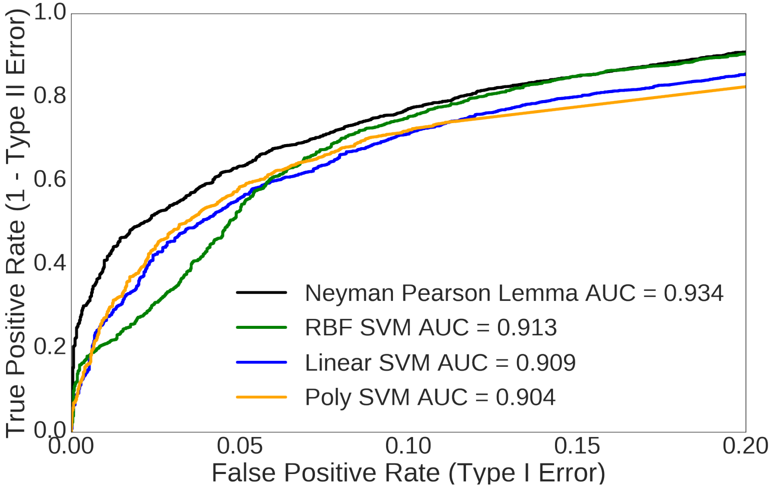


Polynomial $k(\vec{x}_i, \vec{x}_j) = (\vec{x}_i \cdot \vec{x}_j)^d$



CERN School of Computing

Example Classifier Quality



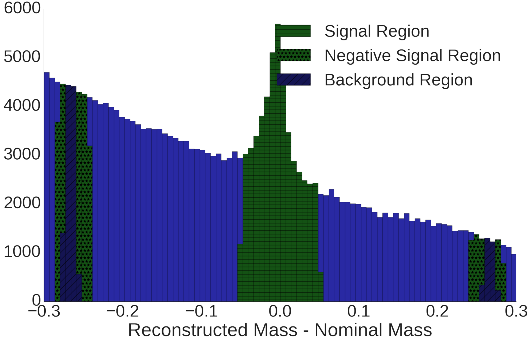
sPlot / Data-driven techniques

Reweighting

- Requires MC and data events
- Train classifier to distinguish data events and MC events
- Reweight MC events using output of classifier
- Train classifier to distinguish signal and background using reweighted MC

Side-band subtraction

- Requires number of events in signal region and sideband
- Compensates background events in signal region with negative signal events from the sideband

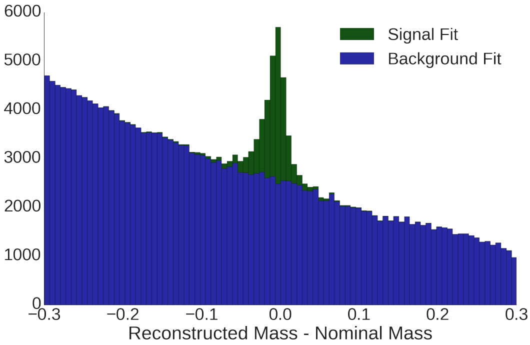


CERN School of Computing

sPlot

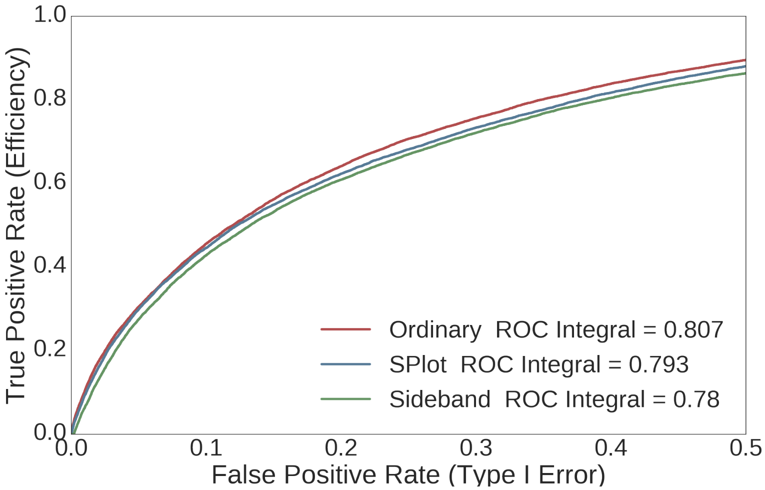
- Requires yields and covariance of fitted signal + background model
- Uses every event twice, as signal and as background with sPlot weight

$$w(\vec{x}_i) = \frac{V_{SS} PDF(\vec{x}_i|S) + V_{SB} PDF(\vec{x}_i|B)}{N_S PDF(\vec{x}_i|S) + N_B PDF(\vec{x}_i|B)}$$



CERN School of Computing

Example Classifier Quality



Artifical neural networks / Deep learning

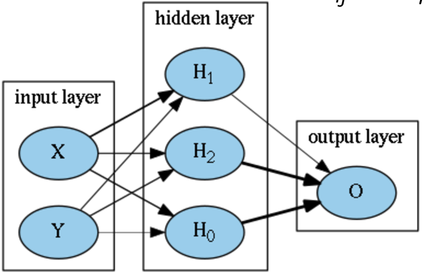
Simplest Form: Feed-Forward Network

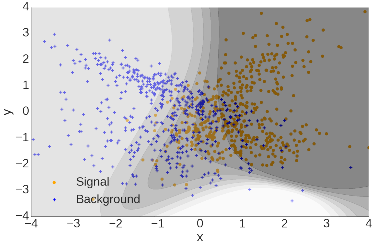
- The data flows from the input to the output layer (feed-forward)

$$f(\vec{x}) = \sigma\left(\sum w_j^{\mathrm{hid}} \sigma\left(\sum w_i^{\mathrm{inp}} x_i\right)\right)$$

- Neurons sum up inputs and apply activation function $\sigma =$ 
- The gradient of the loss-function flows from the output to the input layer and modifies the weights (back-propagation)

$$\Delta w_{ij} = -\eta \frac{\partial L}{\partial w_{ij}} \qquad L = (f(\vec{x}) - t)^2$$





CERN School of Computing

Selected aspects of training

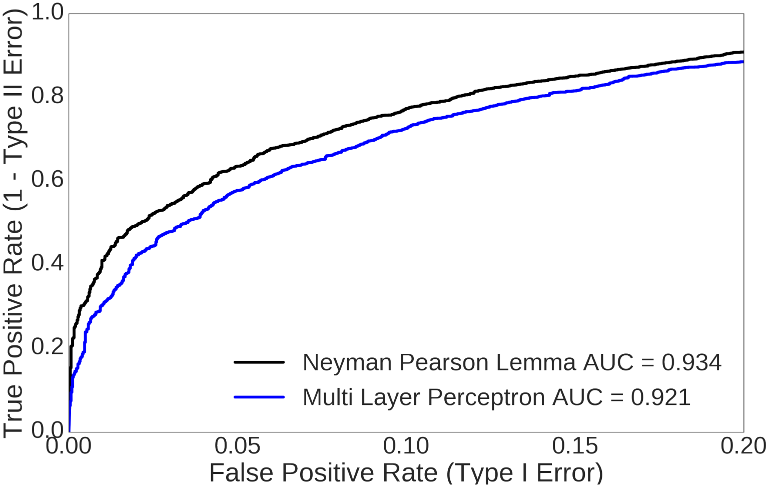
ANN yields small and fast models, but training can be challenging

$$\Delta w_{ij} = -\eta \frac{\partial L}{\partial w_{ij}}$$

- Stochastic gradient-descent algorithm (Backpropagation)
  - Batch-size
  - Learning rate
  - Momentum term (Adam)
  - Hesse Matrix (BFGS)
- Regularization
  - Weight decay $\alpha \vec{w}^T \vec{w}$
  - Dropout (ensemble)
- Architecture
  - Number of neurons
  - Number of layers
  - Activation function
- Initalization
  - Usually gaussian distributed
  - Xavier initialization $\mathrm{Var}(w_{ij}) = \frac{1}{N}$

CERN School of Computing

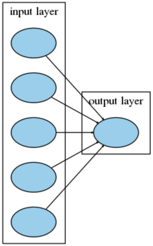
Example Classifier Quality



History (stay with me it's fun!)

- Field started in the 1950s
- ... and died 1969 (Minsky and Papert)
- Assumed incapability to perform operations like exclusive-or
- Lack of computing power

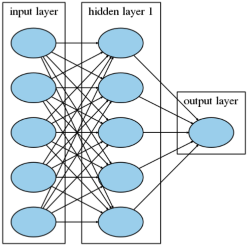
Perceptron



History (stay with me it's fun!)

- Field revolutionized in the 1980s by backpropagation algorithm
- Slowly superseded by methods like SVM, BDTs in the 1990s
- Assumed incapability to train many layers due to local minima
- Lack of computing power

Multi-Layer Perceptron

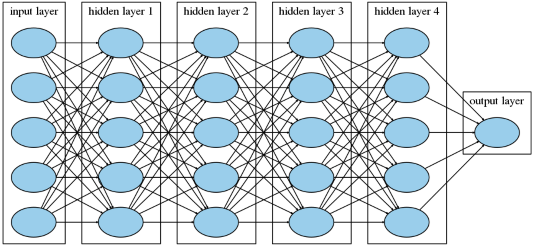


CERN School of Computing

History (stay with me it's fun!)

- Field revolutionized in the 2000s by deep learning
- Advances it algorithms $\rightarrow$ training with many layer is possible
- More statistic (big data)
- Massive boost in computing power (due to GPUs)

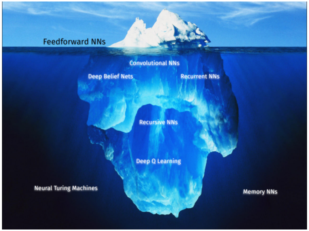
Deep neural network



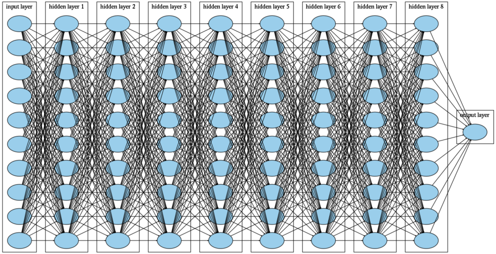
CERN School of Computing

Today (all aboard the hype train)

- Representation learning
- Feed in low-level features
  → learn high-level features automatically
- HEP is getting into it as well!



Deep neural network



Physics example

Received 19 Feb 2014 | Accepted 4 Jun 2014 | Published 2 Jul 2014
DOI: 10.1038/ncomms5308

Searching for exotic particles in high-energy physics with deep learning

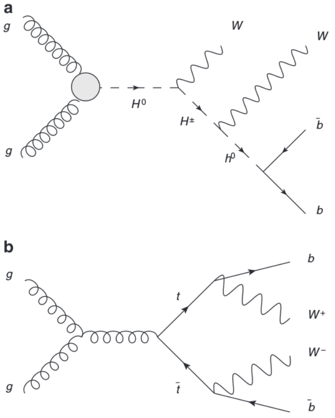


Table 1 | Performance for Higgs benchmark.

Technique	Low-level	High-level	Complete
AUC			
BDT	0.73 (0.01)	0.78 (0.01)	0.81 (0.01)
NN	0.733 (0.007)	0.777 (0.001)	0.816 (0.004)
DN	0.880 (0.001)	0.800 ($<$0.001)	0.885 (0.002)
Discovery significance			
NN	2.5σ	3.1σ	3.7σ
DN	4.9σ	3.6σ	5.0σ

CERN School of Computing

Image recognition example

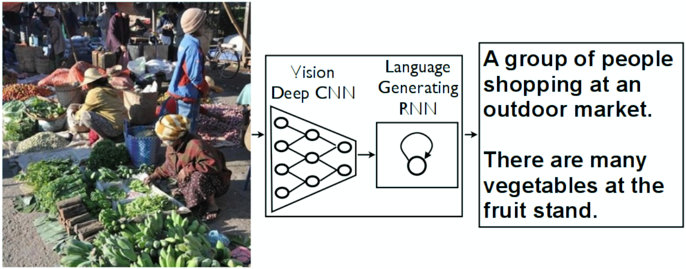
Show and Tell: A Neural Image Caption Generator

Oriol Vinyals
Google
vinyals@google.com

Alexander Toshev
Google
toshev@google.com

Samy Bengio
Google
bengio@google.com

Dumitru Erhan
Google
dumitru@google.com



- Extract information using convolutional neural network
- Generate description using recurrent neural network

CERN School of Computing

Conclusion

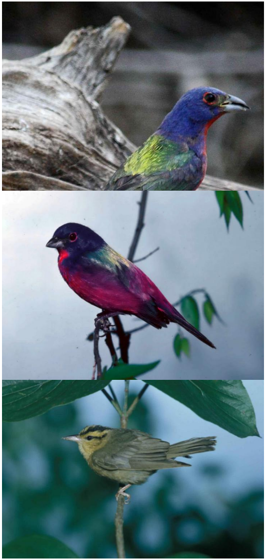
CERN School of Computing

Key messages of the day

- Use multivariate analysis / classification algorithms
- Always test / validate on an independent dataset
- There is a revolution in the field right now!

Backup

Convolutional neural network



http://xkcd.com/1425/

Lots of different birds in different poses, scales and positions!

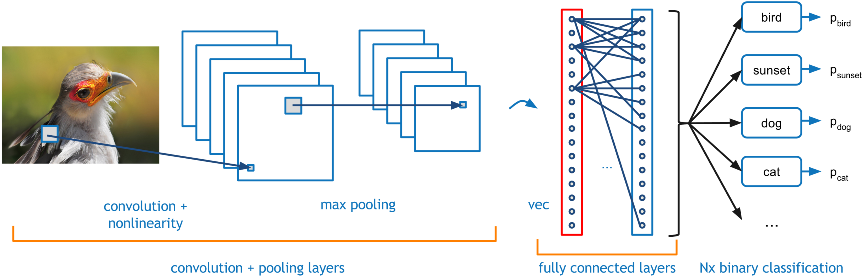
Invariance under Transformations

Different strategies to build a classifier which is invariant under given transformations in the input space:

- Extract hand-crafted features that are invariant
- Use transformed copies during the training phase
- Penalize change in the output under input transformation → Tangent propagation
- **Build invariance properties into structure of neural network**

CERN School of Computing

Park or Bird?



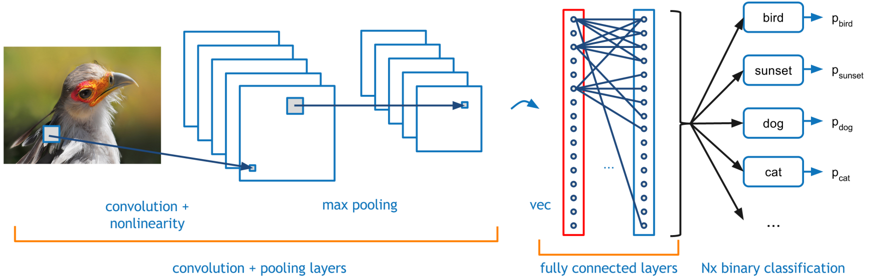
`http://parkorbird.flickr.com/`

Convolutional layer

- Learnable filters (e.g. edge detector) organized in feature maps
- All units take inputs only from small subregions
- Al units are **share the same weights**
- All units detect same pattern but in different locations

CERN School of Computing

Park or Bird?



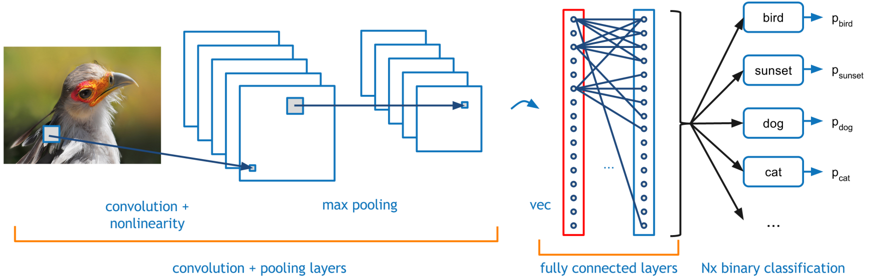
`http://parkorbird.flickr.com/`

Pooling layer

- Take inputs from **small receptive fields** in the feature maps
- Reduce resolution and computation in following layers
- Increases insensitivity against small shifts

CERN School of Computing

Park or Bird?

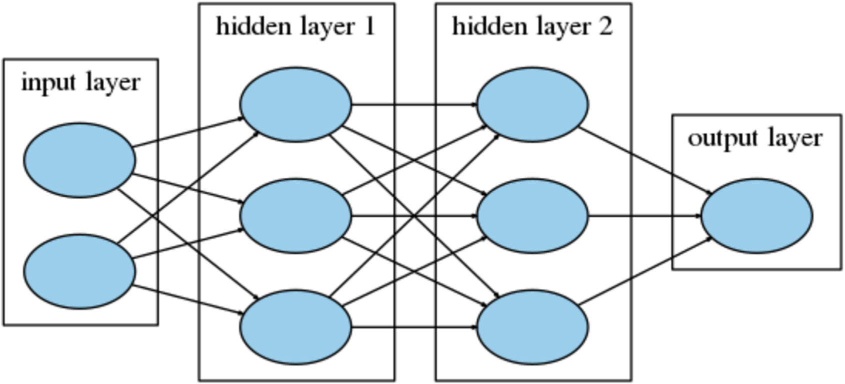


http://parkorbird.flickr.com/

Multiple pairs of convolution and pooling layers

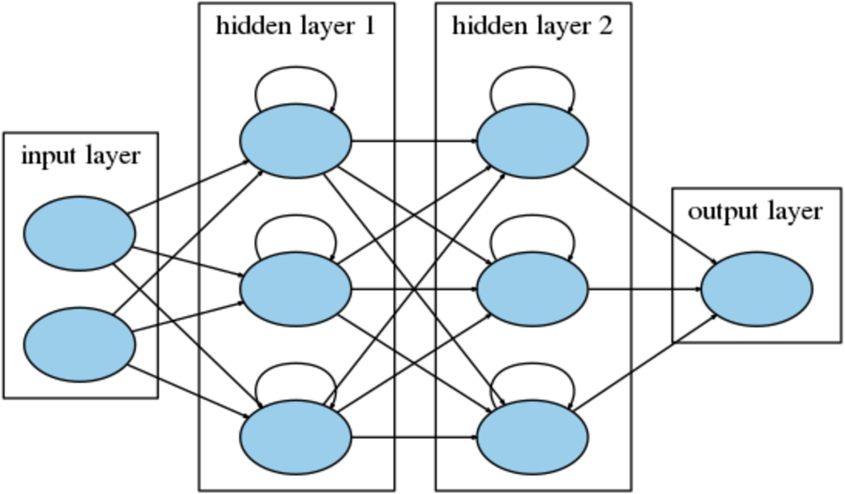
- Each stage has a larger degree of invariance
- Number of features increases as resolution is reduced
- Final layer is fully connected with softmax output

Recurrent neural network



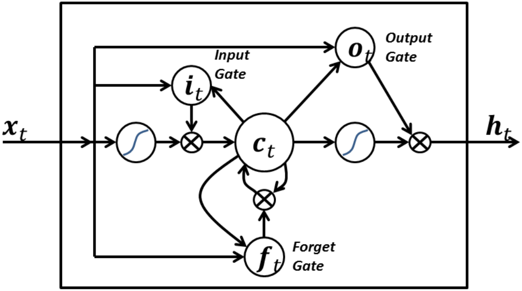
CERN School of Computing

Recurrent neural network



CERN School of Computing

Long Short-Term Memory

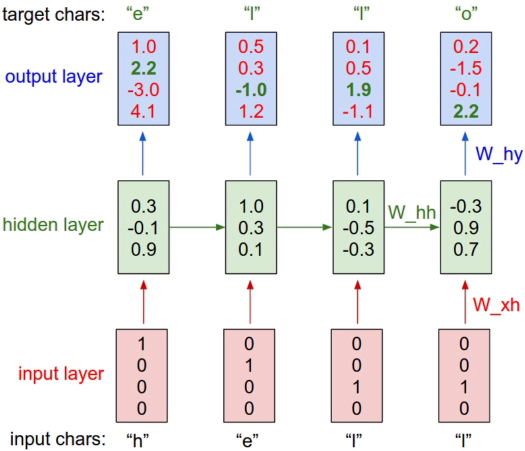


https://en.wikipedia.org/wiki/File:Long_Short_Term_Memory.png

- Can remember a value for a long time period
- Input gate decides when to update the stored value
- Output gate decides when to output the stored value
- Forget gate decides when to forget the stored value

  → **Can process sequential data (e.g. text and speech)**

Character level language model



http://karpathy.github.io/2015/05/21/rnn-effectiveness/

CERN School of Computing

Applied on C-Code

```c
/*
 * Increment the size file of the new incorrect UI_FILTER group information
 * of the size generatively.
 */
static int indicate_policy(void)
{
  int error;
  if (fd == MARN_EPT) {
    if (ss->segment < mem_total)
      unblock_graph_and_set_blocked();
    else
      ret = 1;
    goto bail;
  }
  for (i = 0; i < blocks; i++) {
    seq = buf[i++];
    bpf = bd->bd.next + i * search;
    if (fd) {
      current = blocked;
    }
  }
  return segtable;
}
```

http://karpathy.github.io/2015/05/21/rnn-effectiveness/

CERN School of Computing

Bayes' theorem

$$\underbrace{p(H|D)}_{\text{Posterior distribution}} = \frac{\overbrace{p(D|H)}^{\text{Likelihood}}\overbrace{p(H)}^{\text{Prior distribution}}}{\underbrace{p(D)}_{\text{Evidence}}}$$

Probability is defined as the degree of belief

The Bayesian approach to machine learning

Why should our prior of the model complexity (hypothesis) change with the size of the training data?

CERN School of Computing

The Bayesian approach to machine learning

Why should our prior of the model complexity (hypothesis) change with the size of the training data?

- Assume prior $p(\vec{w})$ for all (hyper-) parameters in the model
- Maximize the posterior $p(\vec{w}|D) \sim p(\vec{w})p(D|\vec{w})$
- Prediction is performed by marginalizing with respect to the posterior distribution

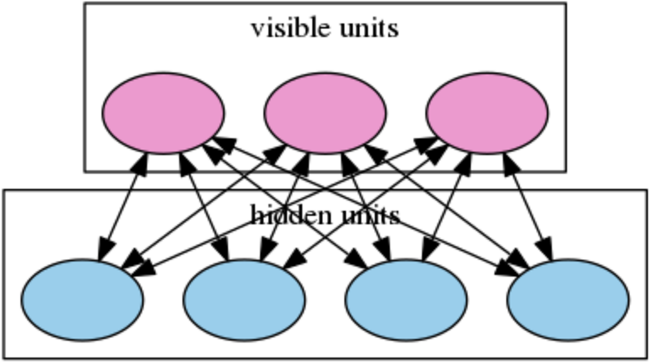
$$p(t|\vec{x}, D) = \int p(t|\vec{x}, \vec{w})p(\vec{w}|D)\mathrm{d}\vec{w}$$

- Mathematically complex due to analytically intractable integrals
- Reproduces weight-decay in case of gaussian prior

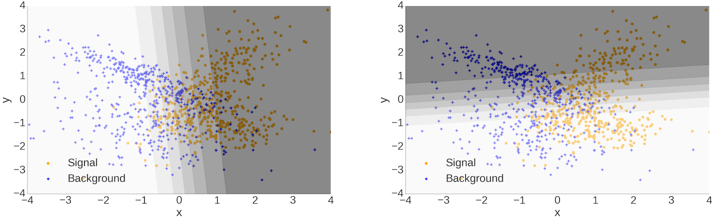
Hyper parameters can be chosen automatically using bayesian methods e.g. automatic relevance determination (ARD)

Restricted Boltzmann machines

- Unsupervised learning of an representation
- Hidden (latent) Variables try to reproduce input layer activation
- Can be stacked on top of each other



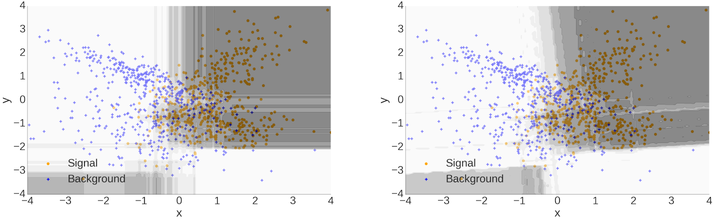
Two dimensional example



Hidden layer activation of a RBM with two input and two hidden neurons

CERN School of Computing

Combined with BDT



BDT trained on original input data (left) and on hidden representation of the RBM (right)

CERN School of Computing

Example Classifier Quality