A boost to Higgs Physics: new regimes at high energy

Silvia Biondi
University & INFN of Bologna
silvia.biondi@cern.ch / silvia.biondi@bo.infn.it

Corso di Dottorato - 2019/2020

Course outline

- Theory reminder
- Higgs boson production and decay modes
- Higgs boson discovery by ATLAS and CMS
- Higgs boson mass measurement by ATLAS and CMS
- Overview of ATLAS and CMS analyses about Higgs
- **Signal/background discrimination techniques**
  - boosted regimes
    - tagging, large-radius jets substructure, re-clustering
  - **multivariate analysis and deep neural network**
- Signal extraction techniques
  - likelihood and test statistic
  - CLs method
- ttH analysis: an example

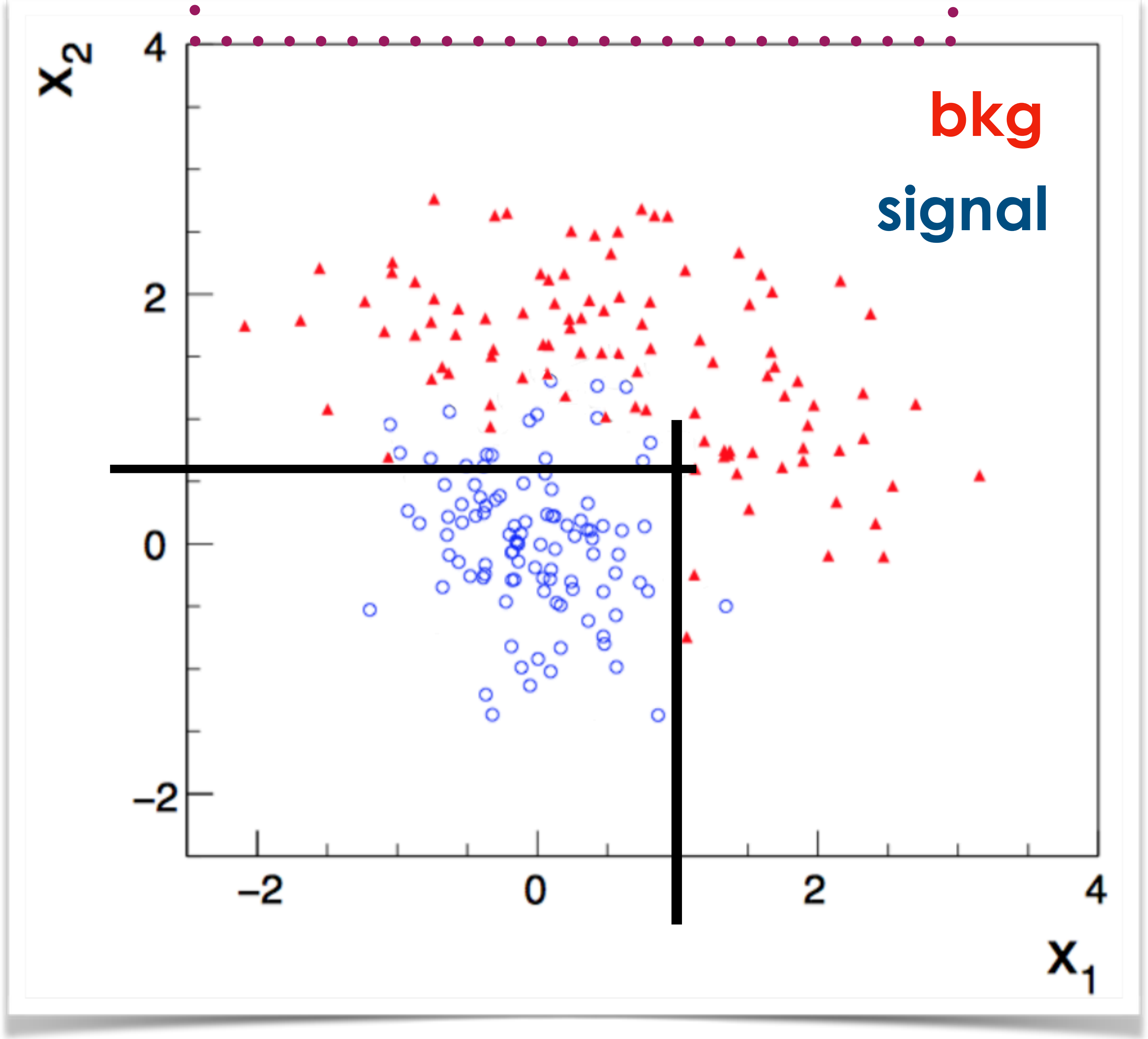
Signal/background discrimination techniques - II

Multivariate analysis technique

Problem

- Analysis aim: to identify **events that are both rare and overwhelmed** by a wide variety of processes that mimic the signal.
- Conventional approach by using cuts on individual kinematic variables **far from be optimal**!

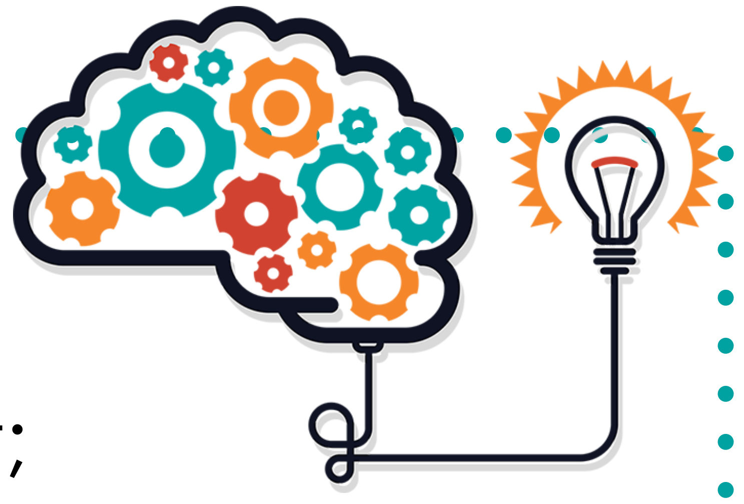
conventional treatment



Multivariate analysis technique

Problem

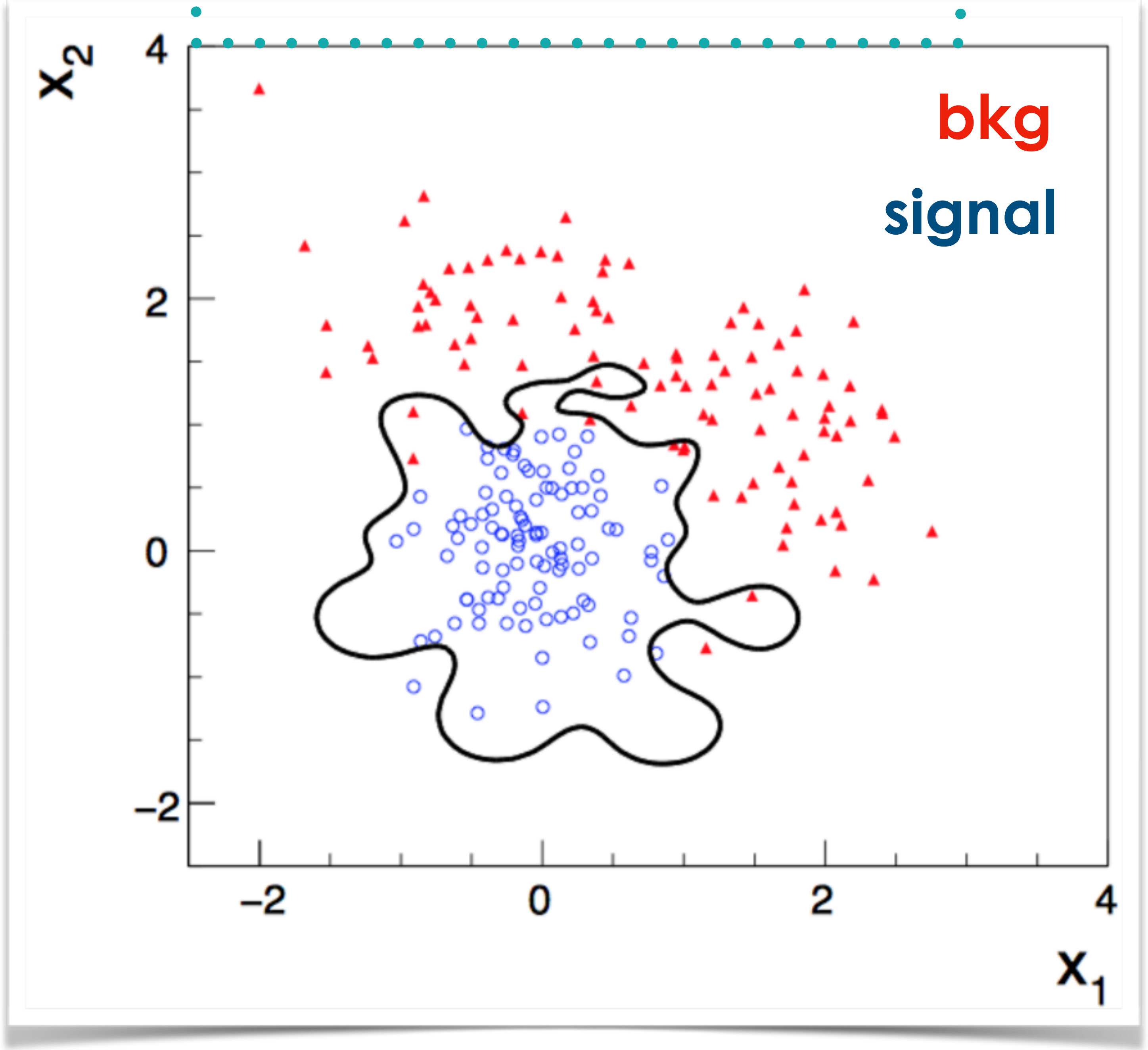
- Analysis aim: to identify **events that are both rare and overwhelmed** by a wide variety of processes that mimic the signal.
- Conventional approach by using cuts on individual kinematic variables **far from be optimal**!



Solution: MultiVariate Analysis (MVA)

1. choice of **set of variables**, characterising an event;
2. application of **non-linear cuts** on signal and background samples;
3. define a function (**classifier**) that, using the **discriminating variables**, is able to identify each event of the real data belonging to the signal or to the background category.

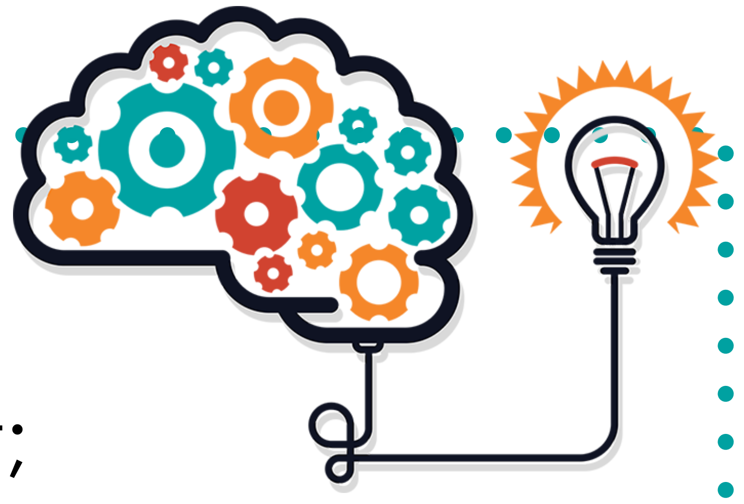
Multivariate treatment



Multivariate analysis technique

Problem

- Analysis aim: to identify **events that are both rare and overwhelmed** by a wide variety of processes that mimic the signal.
- Conventional approach by using cuts on individual kinematic variables **far from be optimal**!

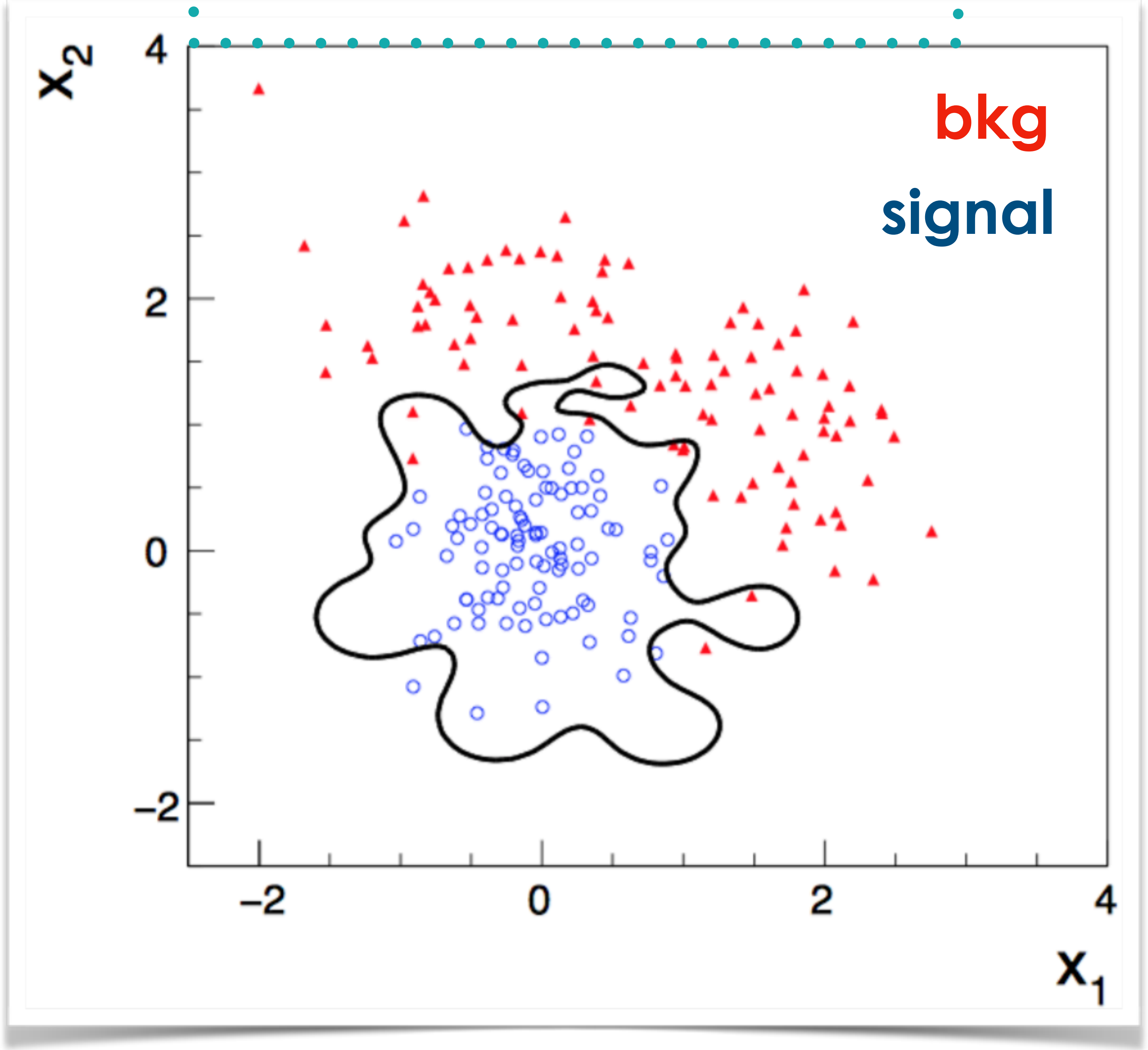


Solution: MultiVariate Analysis (MVA)

1. choice of **set of variables**, characterising an event;
2. application of **non-linear cuts** on signal and background samples;
3. define a function (**classifier**) that, using the **discriminating variables**, is able to identify each event of the real data belonging to the signal or to the background category.

The algorithm "**learns**" signal and background characteristics (**training**) and assigns a weight to each event (~ probability that event is signal or background).

Multivariate treatment



Multivariate analysis technique: fundamental steps

Training (or learning) process

- as input a set of events, characterised by the **feature variables**;
- to define a function (**classifier**) that will be used in the classification step.

Multivariate analysis technique: fundamental steps

Training (or learning) process

- as input a set of events, characterised by the **feature variables**;
- to define a function (**classifier**) that will be used in the classification step.

Supervised training

a set of training events with correct **category association is given**

Unsupervised training

no "a priori" categories are given and the algorithm has to find them by itself.

Multivariate analysis technique: fundamental steps

Training (or learning) process

- as input a set of events, characterised by the **feature variables**;
- to define a function (**classifier**) that will be used in the classification step.

Supervised training

a set of training events with correct **category association is given**

Unsupervised training

no "a priori" categories are given and the algorithm has to find them by itself.

Linear classifier

- a group of **rectangular cuts** on selected variables.
- A sequence of univariate analyser, no combination of variables is achieved and **a cut on a variable does not depend on another one**.

Non-linear classifier

- "non-linear" function: **a single cut on a variable depends simultaneously on all the other variables**
- cuts not necessarily on a linear way

Multivariate analysis technique: fundamental steps

Training (or learning) process

- as input a set of events, characterised by the **feature variables**;
- to define a function (**classifier**) that will be used in the classification step.

Supervised training

a set of training events with correct **category association is given**

Unsupervised training

no "a priori" categories are given and the algorithm has to find them by itself.

Linear classifier

- a group of **rectangular cuts** on selected variables.
- A sequence of univariate analyser, no combination of variables is achieved and **a cut on a variable does not depend on another one**.

Non-linear classifier

- "non-linear" function: **a single cut on a variable depends simultaneously on all the other variables**
- cuts not necessarily on a linear way

Boosted Decision Tree

Deep Neural Network

Multivariate analysis technique: fundamental steps

Training (or learning) process

- as input a set of events, characterised by the **feature variables**;
- to define a function (**classifier**) that will be used in the classification step.

Testing process

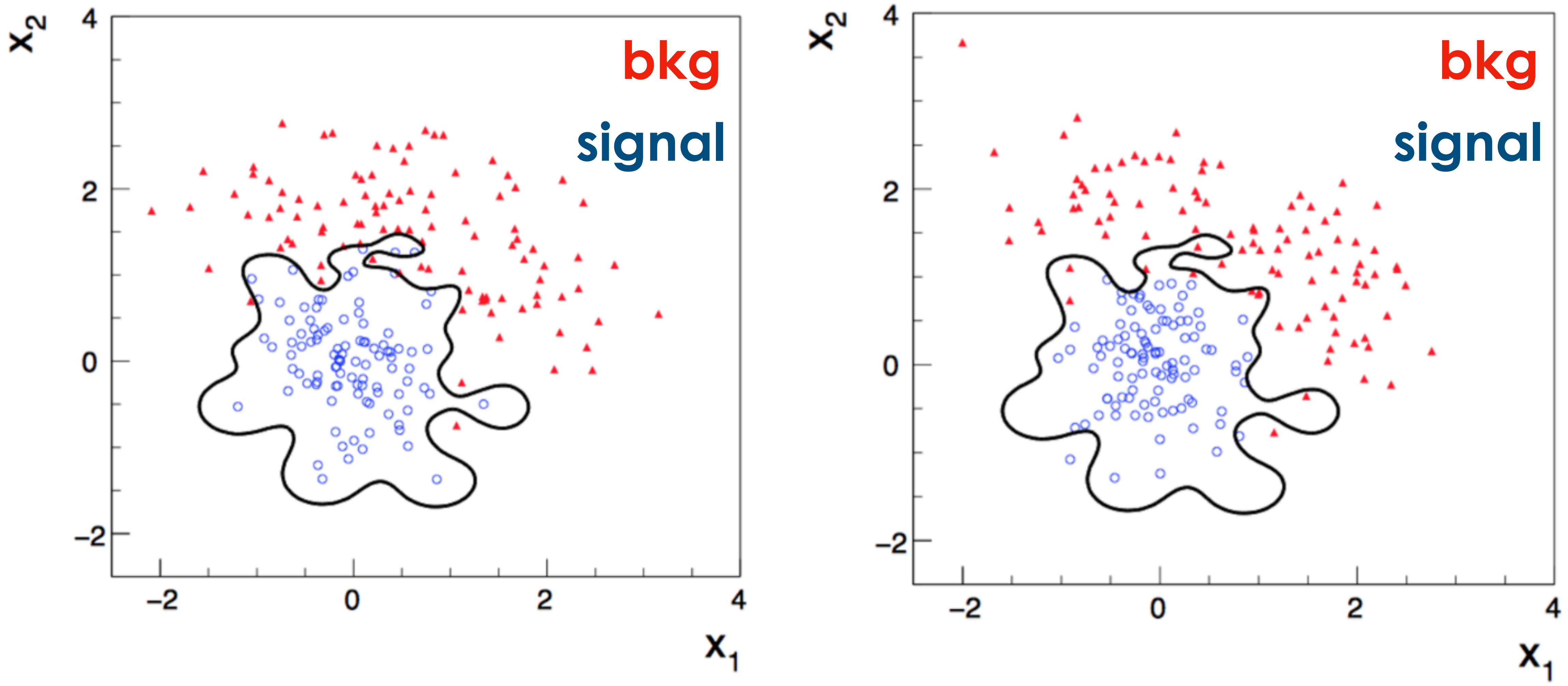
- discriminant variable distributions obtained from **other additional MC signal and bkg samples (statistically independent wrt training ones)**;
- comparison to those of the training test;
- **good agreement is crucial**: it assures that the definition of discriminating variables is not due to a specific feature of training sample.

Multivariate analysis technique: fundamental steps

Training (or learning) process
- as input a set of events, characterised by the **feature variables**;
- to define a function (**classifier**) that will be used in the classification step.

Testing process
- discriminant variable distributions obtained from **other additional MC signal and bkg samples (statistically independent wrt training ones)**;
- comparison to those of the training test;
- **good agreement is crucial**: it assures that the definition of discriminating variables is not due to a specific feature of training sample.

Overtraining
possible inconsistency between training and testing distributions, suggesting that the **definition of discriminating variables relies on features of the particular sample rather than on a general feature of the kind of events to be selected**.



Multivariate analysis technique: fundamental steps

Training (or learning) process

- as input a set of events, characterised by the **feature variables**;
- to define a function (**classifier**) that will be used in the classification step.

Testing process

- discriminant variable distributions obtained from **other additional MC signal and bkg samples (statistically independent wrt training ones)**;
- comparison to those of the training test;
- **good agreement is crucial**: it assures that the definition of discriminating variables is not due to a specific feature of training sample.

Overtraining

possible inconsistency between training and testing distributions, suggesting that the **definition of discriminating variables relies on features of the particular sample rather than on a general feature of the kind of events to be selected**.

Reasons

- too few degrees of freedom;
- too many model parameters of an algorithm adjusted to too few data points.

Consequences

- false increase in performance over the objectively achievable one, if measured on the training sample;
- effective performance decrease when measured in an independent testing sample.

Multivariate analysis technique: fundamental steps

Training (or learning) process

- as input a set of events, characterised by the **feature variables**;
- to define a function (**classifier**) that will be used in the classification step.

Testing process

- discriminant variable distributions obtained from **other additional MC signal and bkg samples (statistically independent wrt training ones)**;
- comparison to those of the training test;
- **good agreement is crucial**: it assures that the definition of discriminating variables is not due to a specific feature of training sample.

Overtraining

possible inconsistency between training and testing distributions, suggesting that the **definition of discriminating variables relies on features of the particular sample rather than on a general feature of the kind of events to be selected**.

Solution

- **optimisation of specific parameters** of the ML algorithm used (i.e., number of nodes, number of variables, deep of the tree);
- **independent samples** for test and training;
- **enough statistics** for samples to avoid fluctuations.

Multivariate analysis technique: fundamental steps

Training (or learning) process

- as input a set of events, characterised by the **feature variables**;
- to define a function (**classifier**) that will be used in the classification step.

Testing process

- discriminant variable distributions obtained from **other additional MC signal and bkg samples (statistically independent wrt training ones)**;
- comparison to those of the training test;
- **good agreement is crucial**: it assures that the definition of discriminating variables is not due to a specific feature of training sample.

Classification process

- **assign objects or events to one of the possible discrete classes** (e.g. signal and background) by the classifier found in the training step;
- after this, the events of **real data are split into signal and background classes**.

Multivariate analysis technique: fundamental steps

Training (or learning) process

- as input a set of events, characterised by the **feature variables**;
- to define a function (**classifier**) that will be used in the classification step.

Testing process

- discriminant variable distributions obtained from **other additional MC signal and bkg samples (statistically independent wrt training ones)**;
- comparison to those of the training test;
- **good agreement is crucial**: it assures that the definition of discriminating variables is not due to a specific feature of training sample.

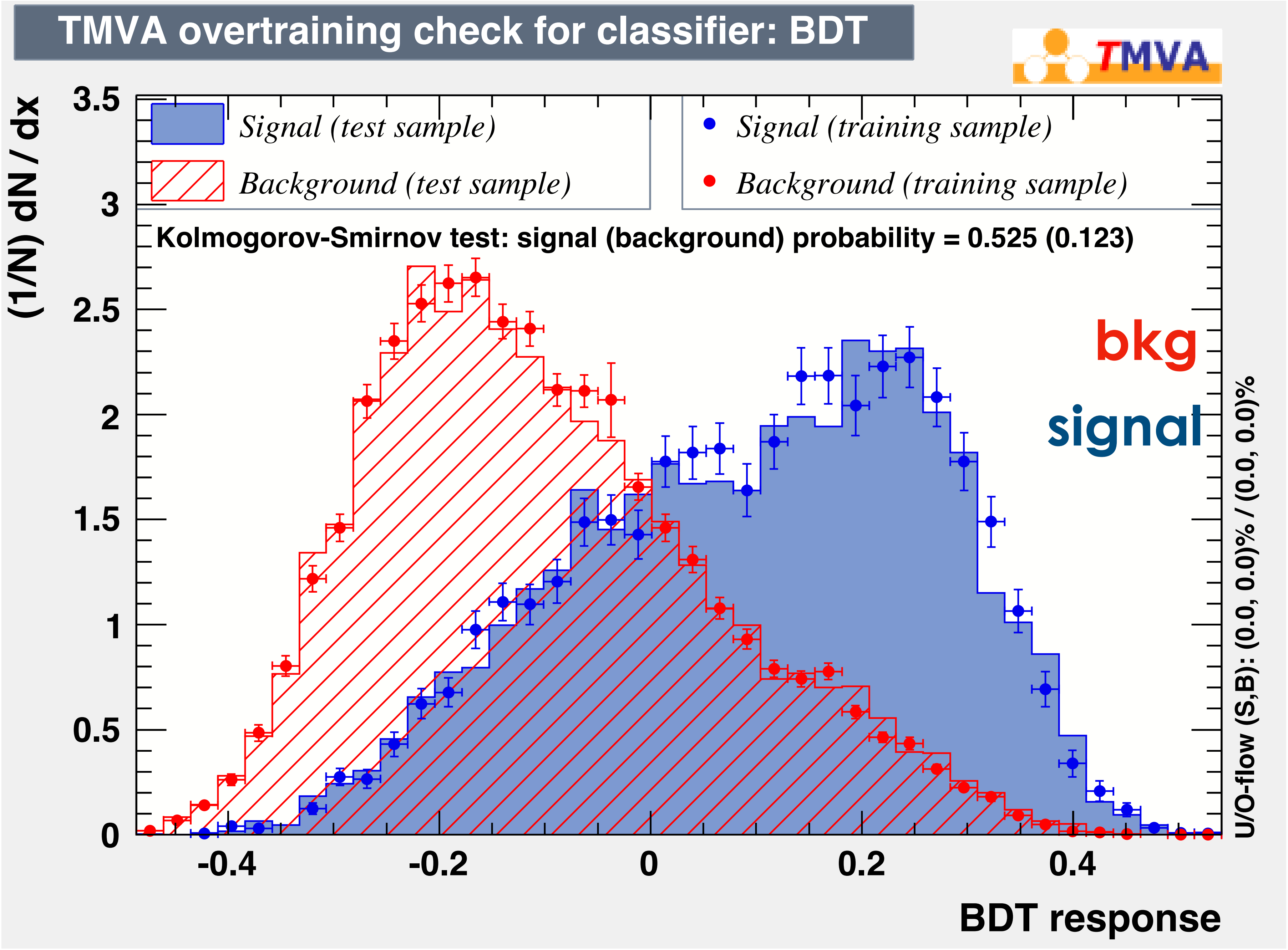
Classification process

- **assign objects or events to one of the possible discrete classes** (e.g. signal and background) by the classifier found in the training step;
- after this, the events of **real data are split into signal and background classes**.

Different evaluation methods

- separation;
- correlation;
- importance ranking.

Multivariate analysis technique: classification evaluation



Separation

- y_S and y_B are the **signal and background probability density functions of y**, respectively;
- zero for identical signal and background shapes and 1 for shapes with no overlap.

$$< S^2 >= \frac{1}{2}\int \frac{(\hat{y}_S(y) - \hat{y}_B(y))^2}{\hat{y}_S(y) + \hat{y}_B(y)} dy$$

Correlation

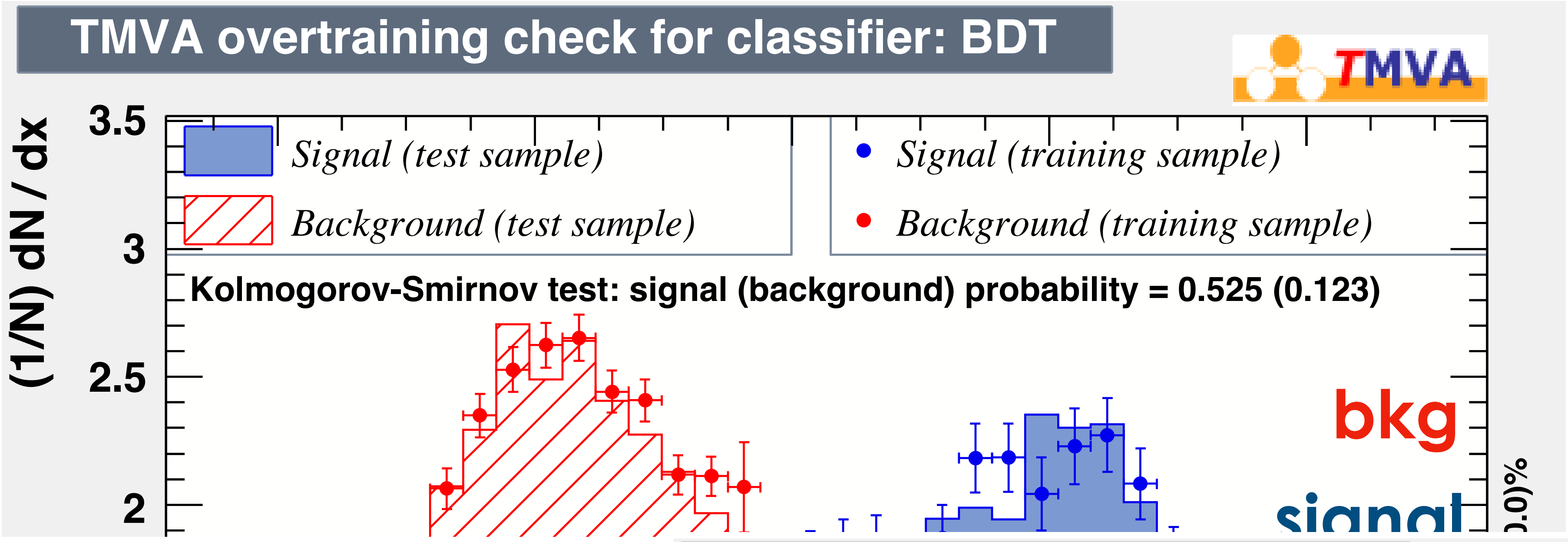
- two random variables X and Y;
- cov is the **covariance** and σ(X) (σ(Y)) is the **variance** of X (Y).

$$\rho(X,Y) = \frac{cov(X,Y)}{\sigma_X \sigma_Y}$$

Importance ranking

- by evaluating the **number of times the variables are used to split decision tree nodes**;
- by weighting each split occurrence (by using the same variable) by the separation achieved and by the number of events in the node.

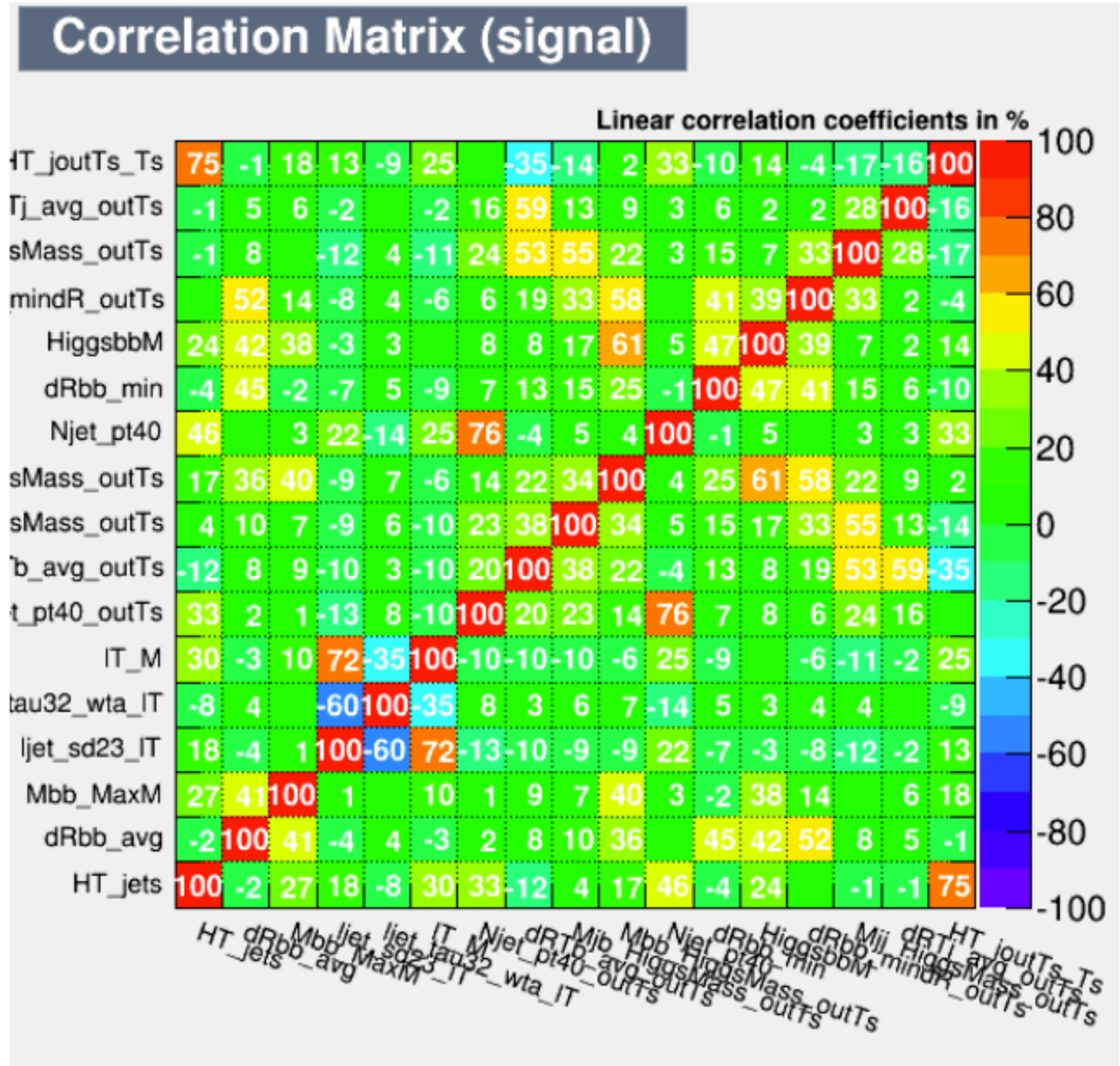
Multivariate analysis technique: classification evaluation

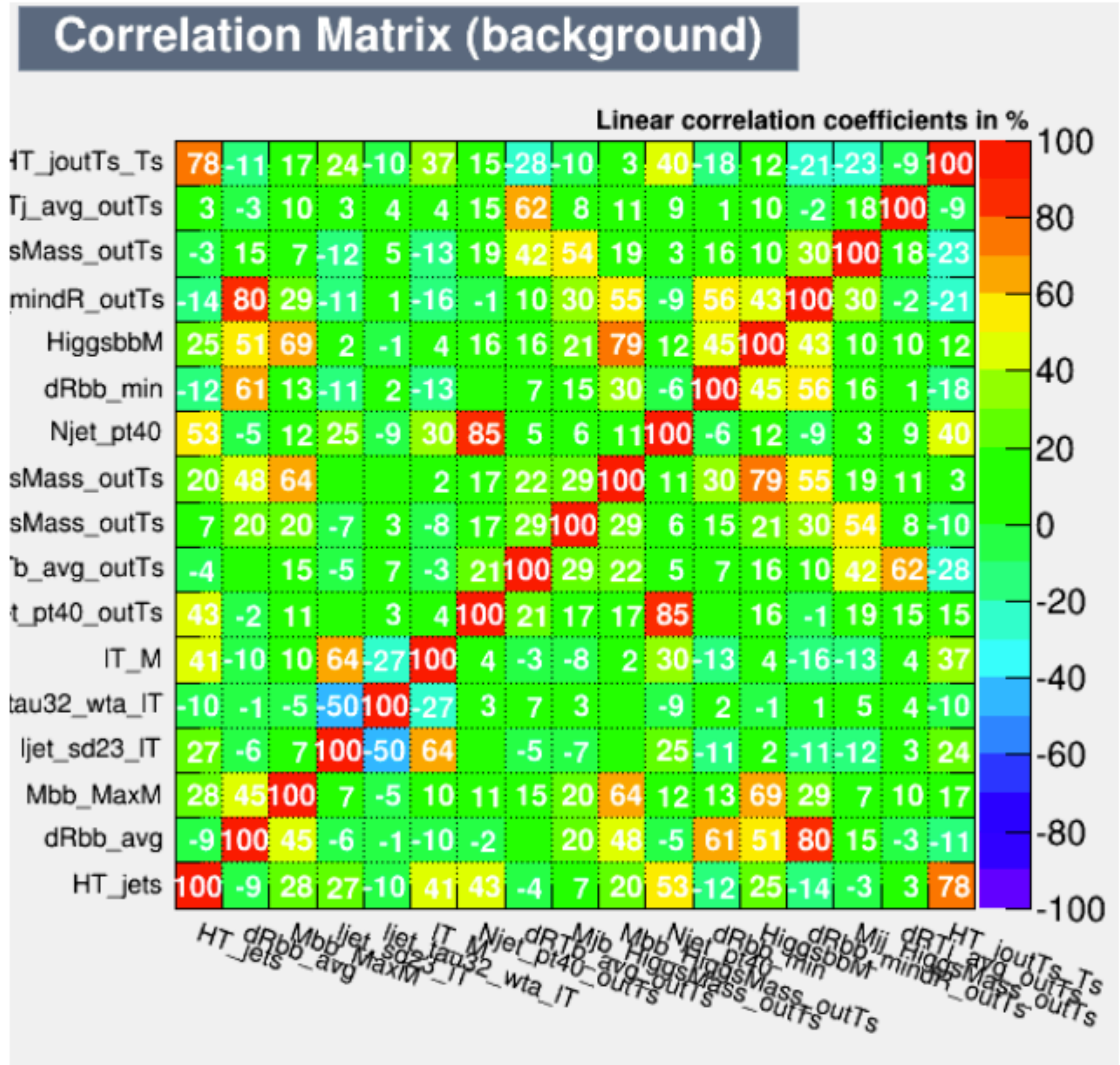


Separation

- y_S and y_B are the **signal and background probability density functions of y**, respectively;
- zero for identical signal and background shapes and 1 for shapes with no overlap.

$$< S^2 >= \frac{1}{2} \int \frac{(\hat{y}_S(y) - \hat{y}_B(y))^2}{\hat{y}_S(y) + \hat{y}_B(y)} dy$$





Correlation

- two random variables X and Y;
- cov is the **covariance** and σ(X) (σ(Y)) is the **variance** of X (Y).

$$\rho(X,Y) = \frac{cov(X,Y)}{\sigma_X \sigma_Y}$$

Importance ranking

- by evaluating the **number of times the variables are used to split decision tree nodes**;
- by weighting each split occurrence (by using the same variable) by the separation achieved and by the number of events in the node.

Multivariate analysis technique: classification evaluation

Separation

- y_S and y_B are the **signal and background probability density functions of y**, respectively;
- zero for identical signal and background shapes and 1 for shapes with no overlap.

$$< S^2 >= \frac{1}{2} \int \frac{(\hat{y}_S(y) - \hat{y}_B(y))^2}{\hat{y}_S(y) + \hat{y}_B(y)} dy$$

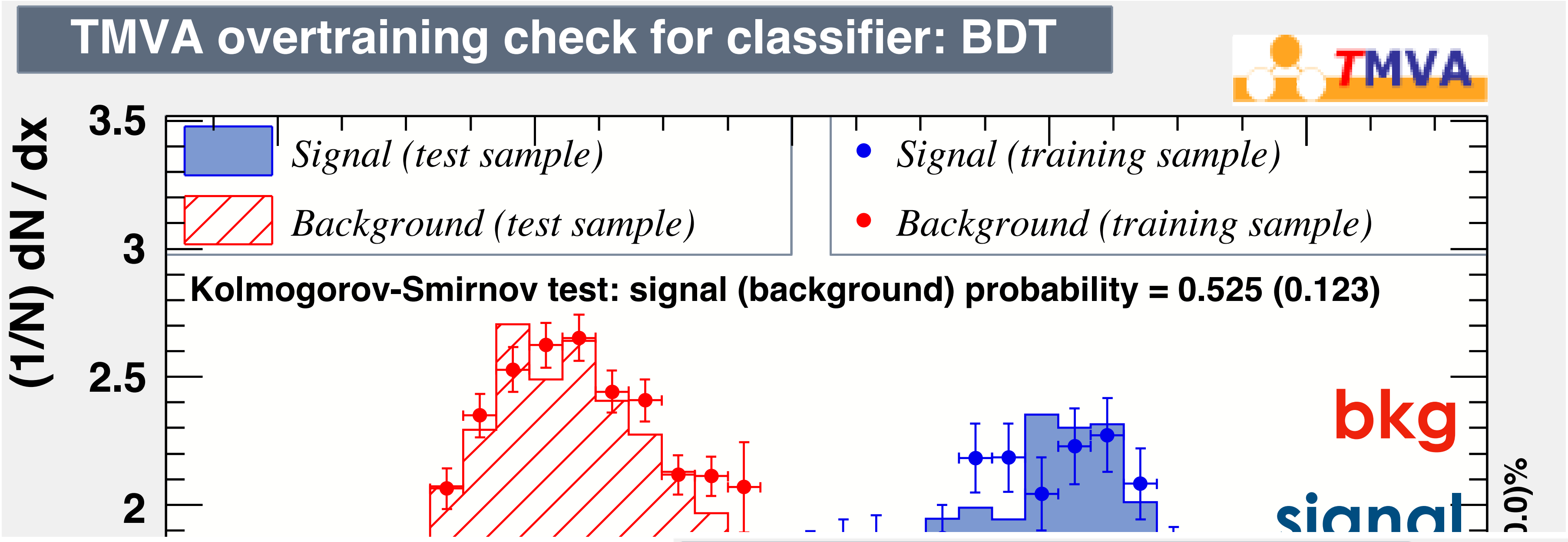
Correlation

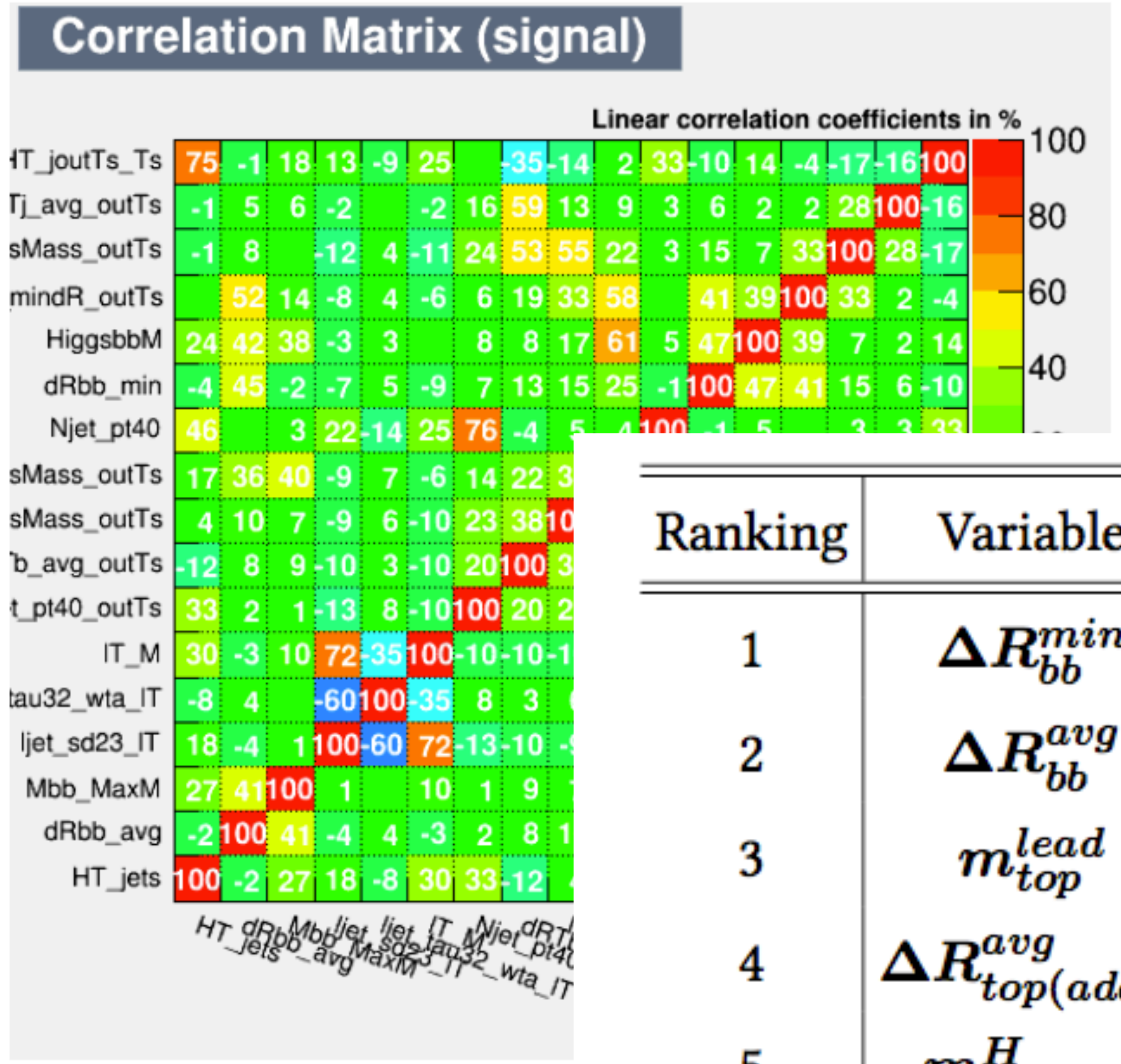
- two random variables X and Y;
- cov is the **covariance** and σ(X) (σ(Y)) is the **variance** of X (Y).

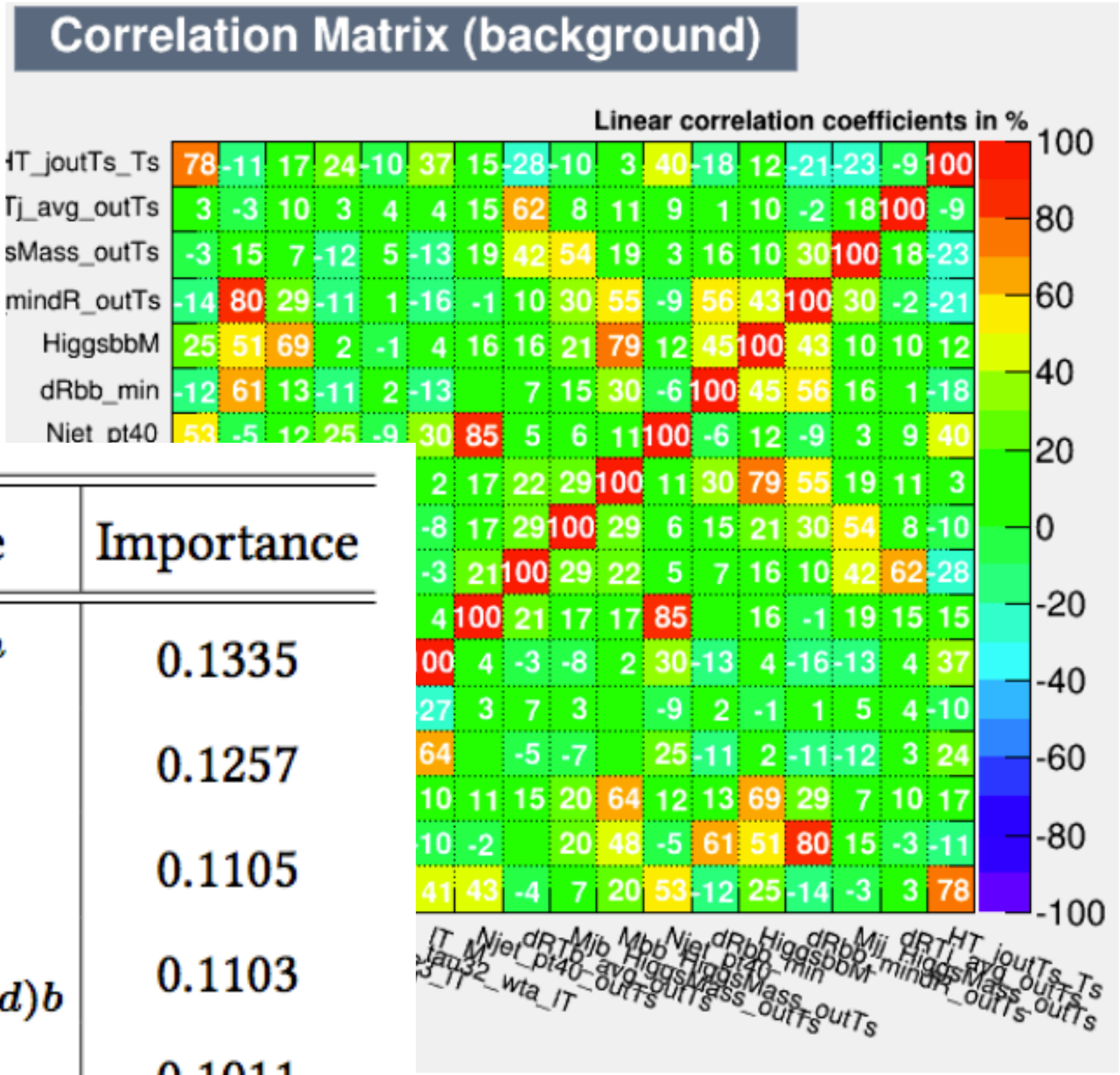
$$\rho(X,Y) = \frac{cov(X,Y)}{\sigma_X \sigma_Y}$$

Importance ranking

- by evaluating the **number of times the variables are used to split decision tree nodes**;
- by weighting each split occurrence (by using the same variable) by the separation achieved and by the number of events in the node.



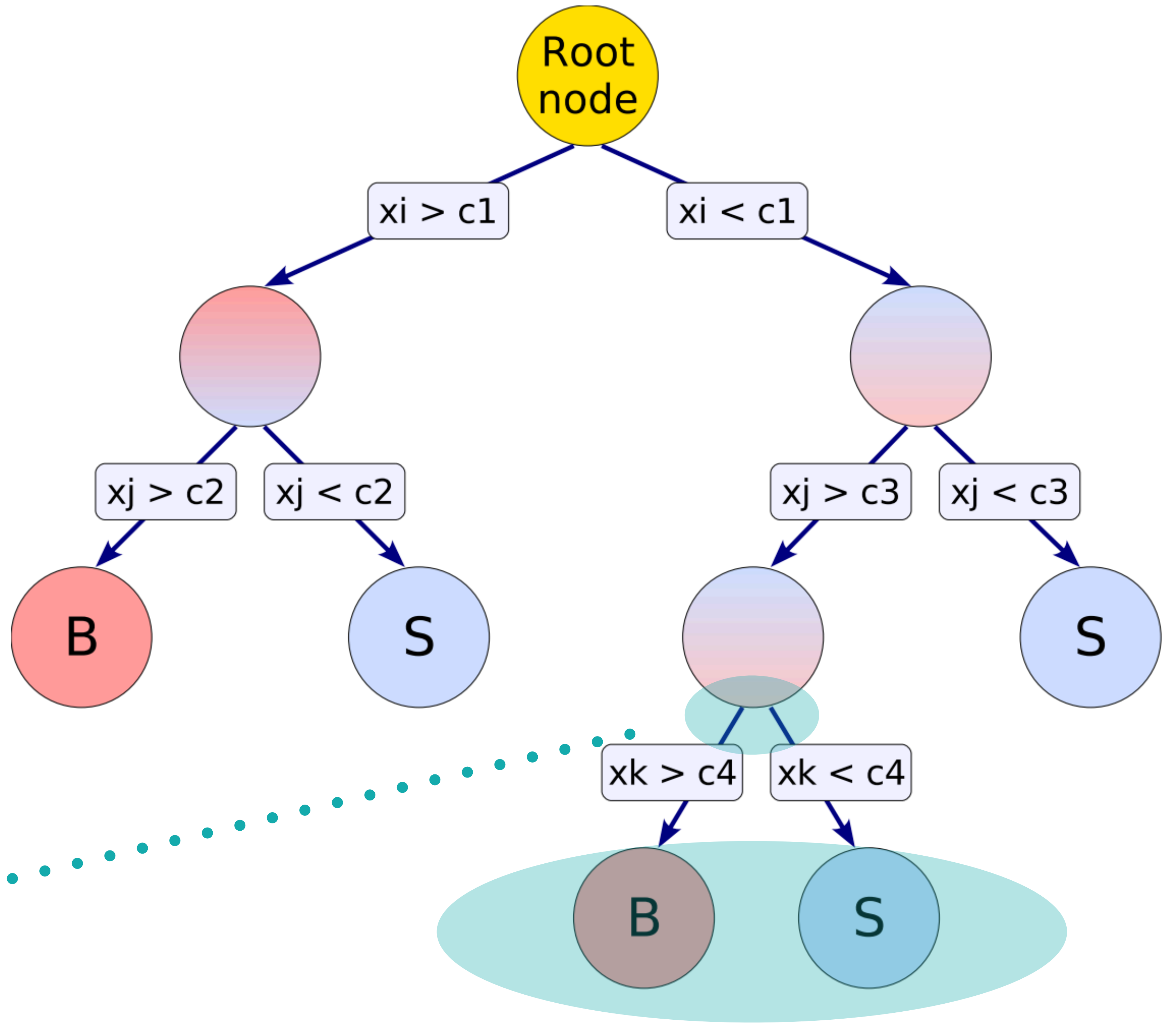




Ranking	Variable	Importance
1	ΔR_{bb}^{min}	0.1335
2	ΔR_{bb}^{avg}	0.1257
3	m_{top}^{lead}	0.1105
4	$\Delta R_{top(add)b}^{avg}$	0.1103
5	$m_{(add)bb}^{H}$	0.1011
6	$\Delta R_{(add)bb}^{min}$	0.0947
7	N_j^{40}	0.0867
8	m_{bb}^{max}	0.0808
9	$\tau_{32,top}^{lead}$	0.0794
10	H_T^{jet}	0.0775

Boosted Decision Tree - BDT

- Decision tree is a **binary tree structure**:
  - repeated binary decisions (yes/no) taken on one single variable at a time, until stop criteria fulfilled;
- **phase space split into many regions** eventually classified as signal or bkg
  - depending on majority of training events that end up in the final leaf node.
- a sequence of **binary splits applied to the data**, using discriminating variables.



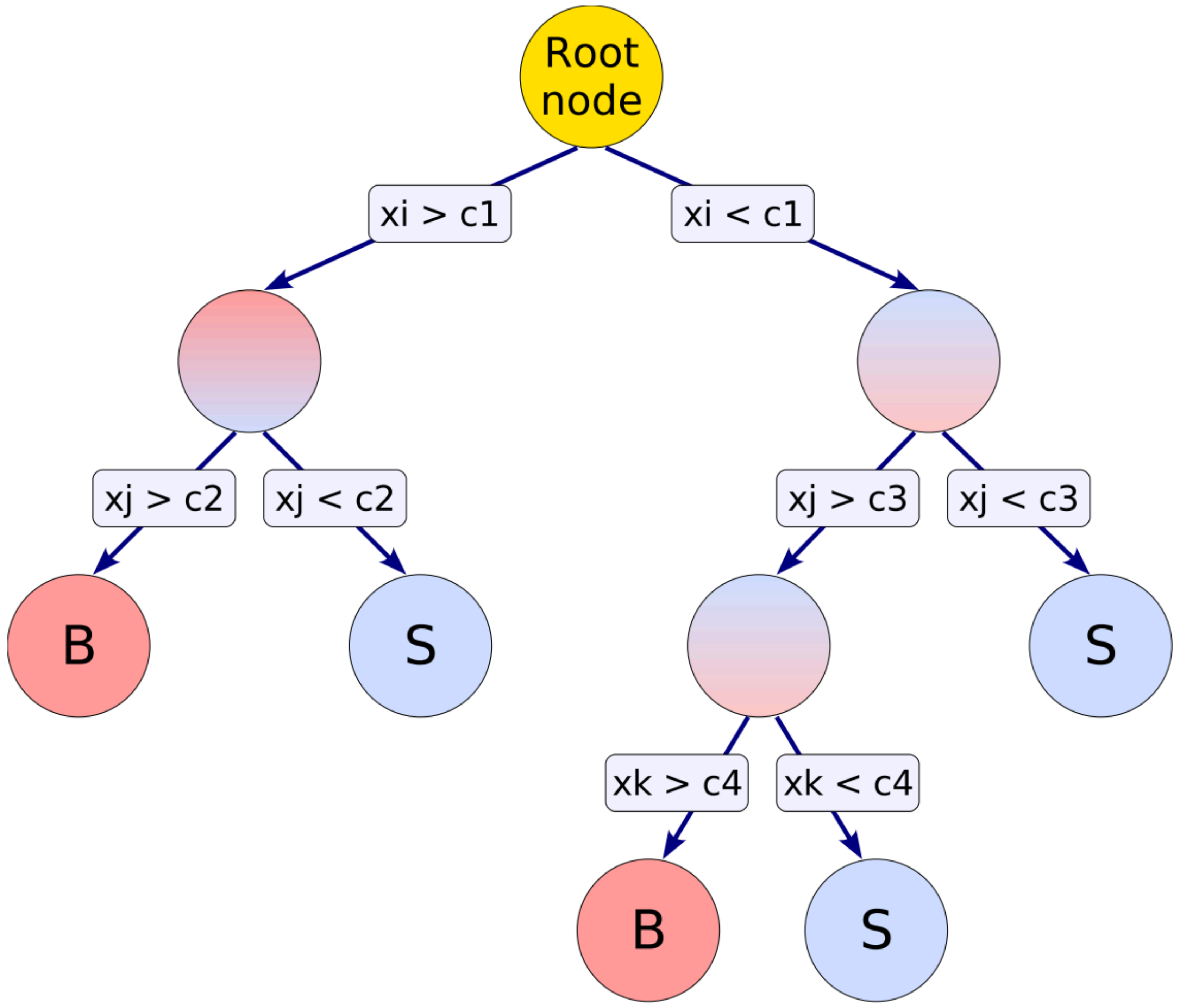
Nodes
where a decision is made

Final leaves
where classification of events lives

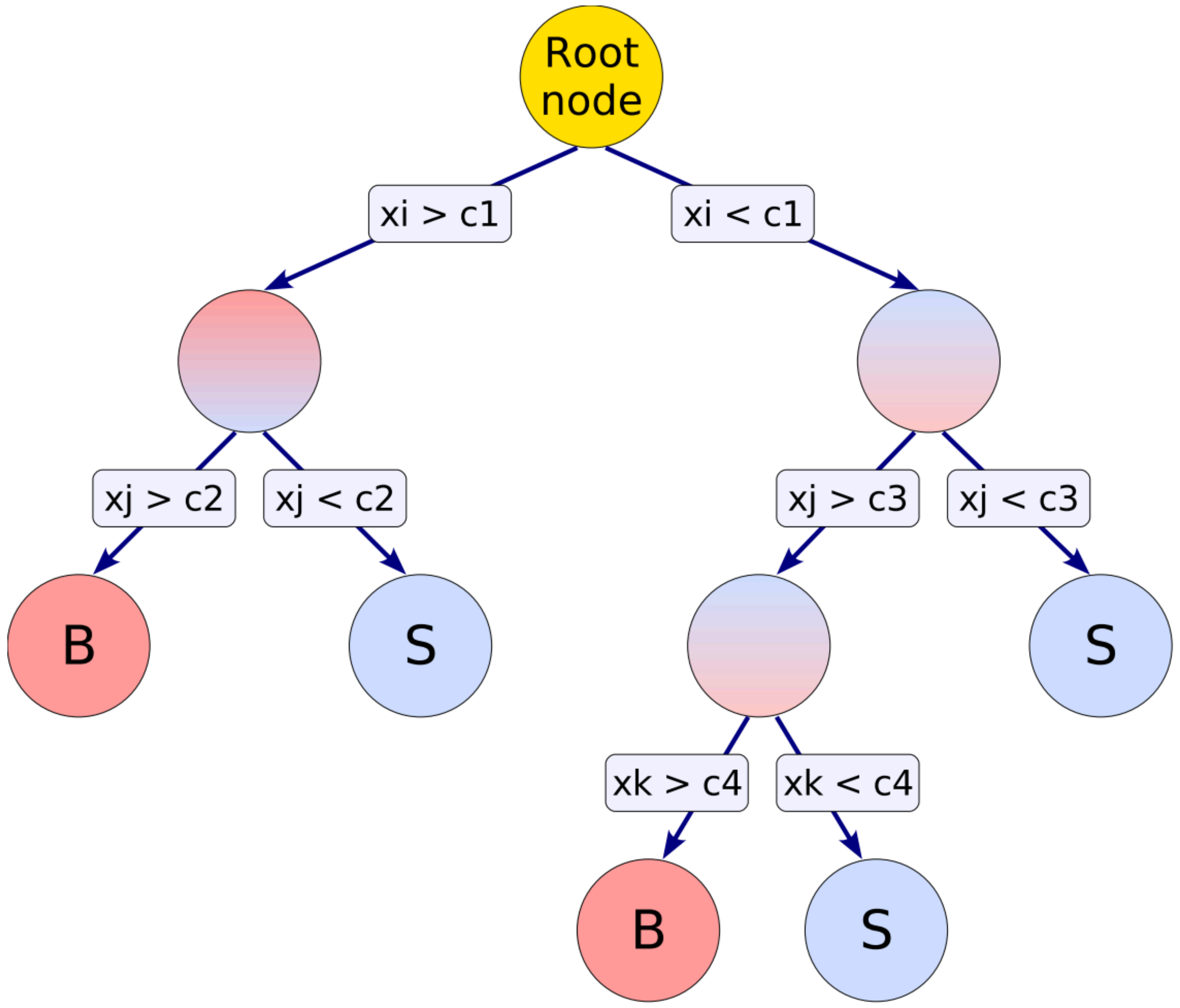
Boosted Decision Tree - BDT

- Decision tree is a **binary tree structure**:
  - repeated binary decisions (yes/no) taken on one single variable at a time, until stop criteria fulfilled;
- **phase space split into many regions** eventually classified as signal or bkg
  - depending on majority of training events that end up in the final leaf node.
- a sequence of **binary splits applied to the data**, using discriminating variables.

- Instability with respect to **statistical fluctuations** in the training sample.



Boosted Decision Tree - BDT



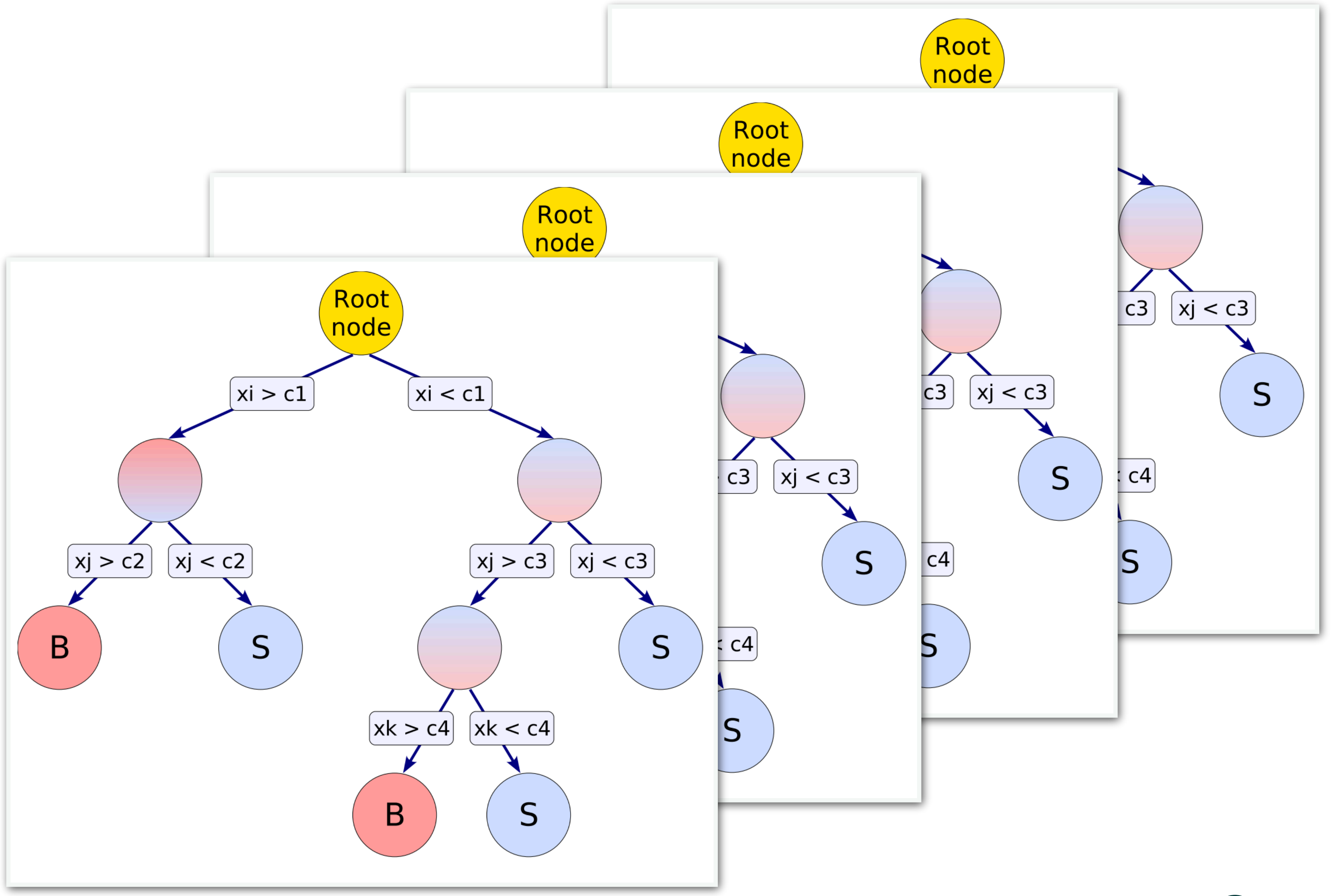
- Decision tree is a **binary tree structure**:
  - repeated binary decisions (yes/no) taken on one single variable at a time, until stop criteria fulfilled;
- **phase space split into many regions** eventually classified as signal or bkg
  - depending on majority of training events that end up in the final leaf node.
- a sequence of **binary splits applied to the data**, using discriminating variables.

- Instability with respect to **statistical fluctuations** in the training sample.

BOOST!

Boosting

- Extension of this concept from one tree to several trees which form a "**forest**";
- trees are derived from the same training ensemble by **reweighting events**;
- finally combined into a **single classifier** which is given by an **average of the individual decision trees**.



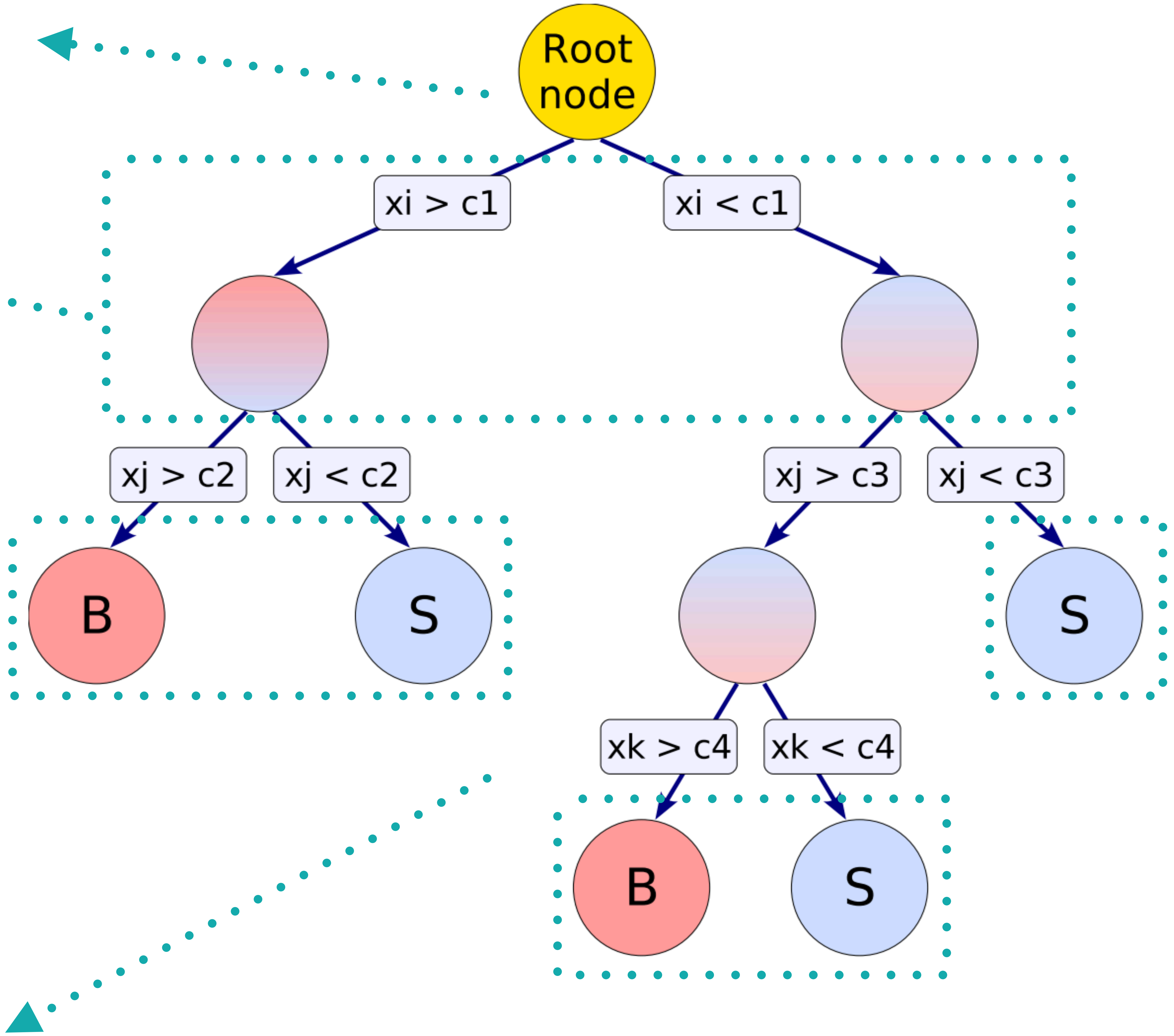
Advantages

more stable response of the decision trees, wrt fluctuations in the training sample, thus **enhancing the performance wrt a single tree**

Boosted Decision Tree - BDT

Training a decision tree

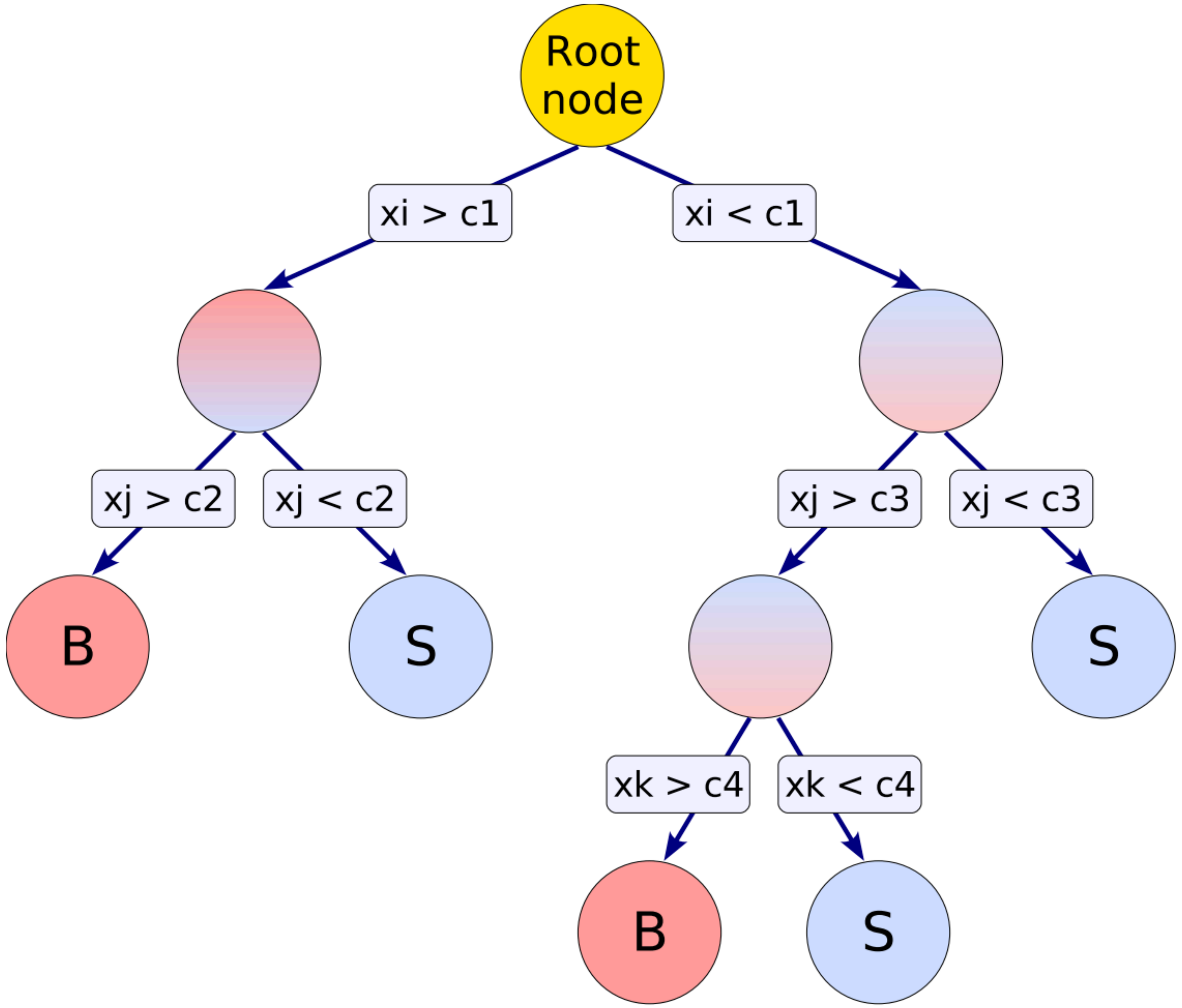
- starts with the **root node**, where an initial splitting criterion for the full training sample is determined;
- split results in **two subsets of training events**, each going through the same algorithm determining the splitting criteria of the next nodes;
- procedure is repeated until the whole tree is built.
- At each node, the split is determined by **finding the variable and the corresponding cut value that provides the best separation** between signal and background events that reach that node
  - optimised by scanning over the variable range
  - bin granularity plays an important role!
- Addition of nodes stops once the number of events that should be split is **below a threshold** which is specified in the BDT configuration;
- **final leaves** are classified as signal or background according to the class the majority of events belongs to.



Boosted Decision Tree - BDT

When do we stop?

- In principle, the splitting could continue until each leaf node contains only signal or only background events;
- such a decision tree would be strongly **overtrained**!



Boosted Decision Tree - BDT

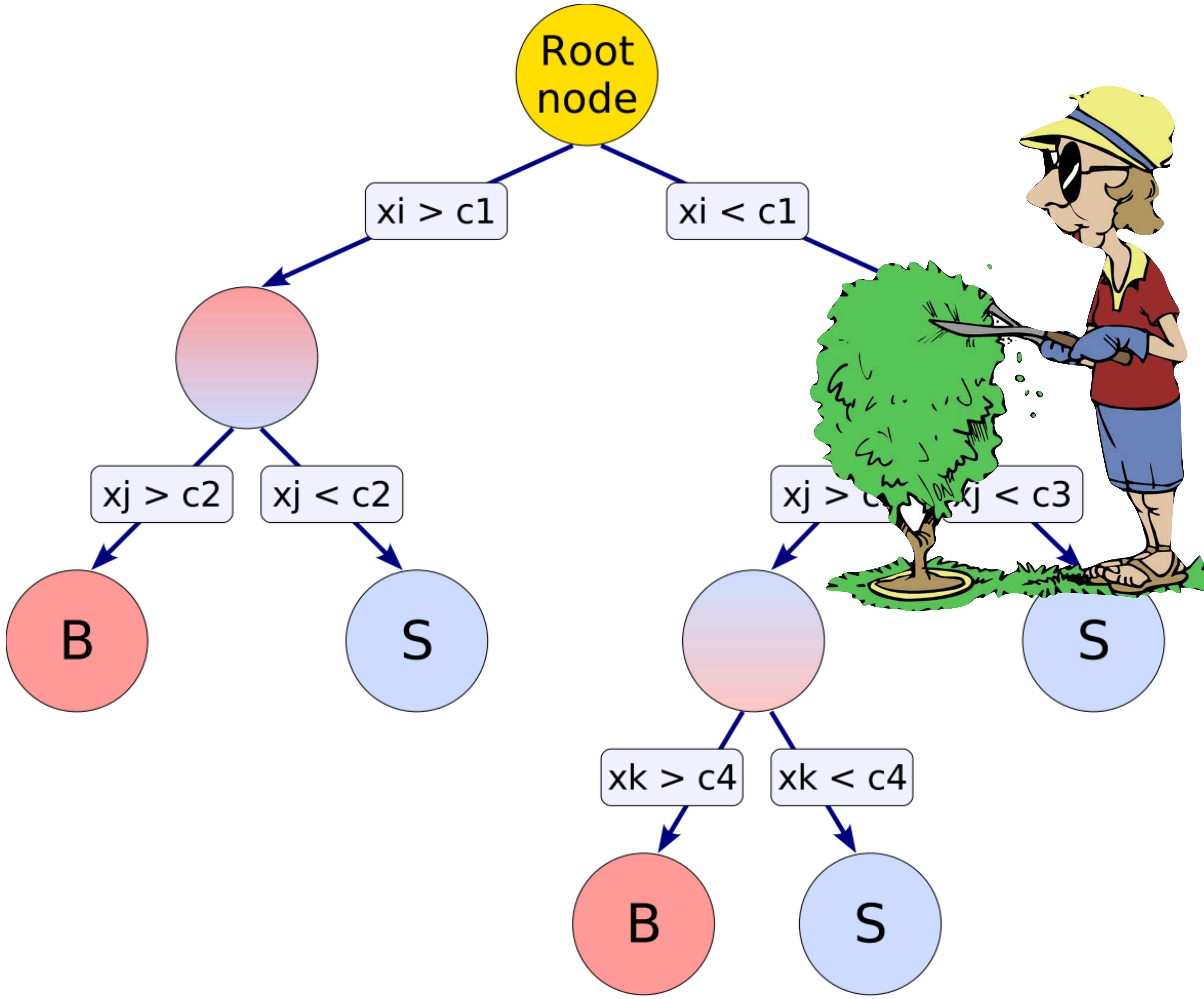
When do we stop?

- In principle, the splitting could continue until each leaf node contains only signal or only background events;
- such a decision tree would be strongly **overtrained**!

Pruning

- process of **cutting back a tree from the bottom up** after it has been built to its maximum size;
- **remove statistically insignificant nodes** and thus reduce the overtraining of the tree;
- **why from the bottom up and not directly interrupting the growing?**
  - apparently insignificant splits can nevertheless lead to good splits further down the tree.

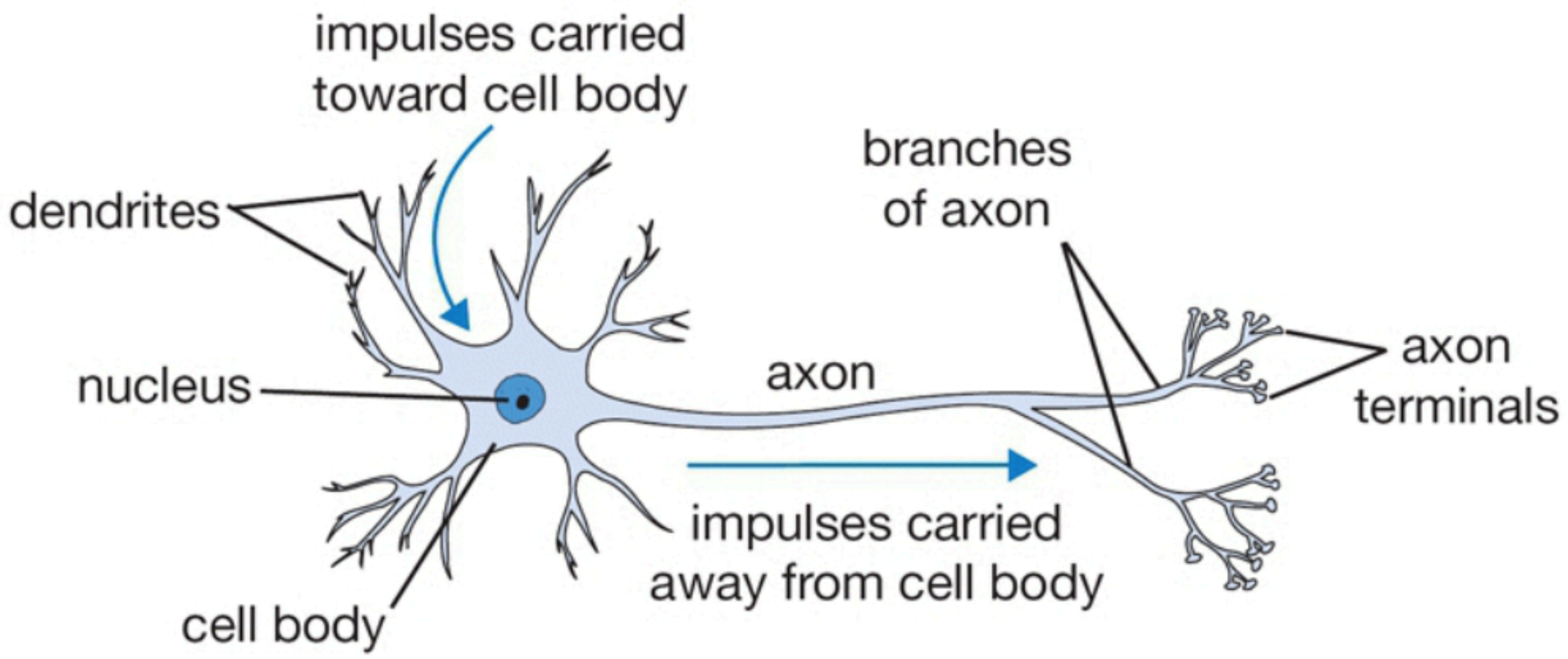
- Different algorithms available, to be optimised for the specific case through parameters.

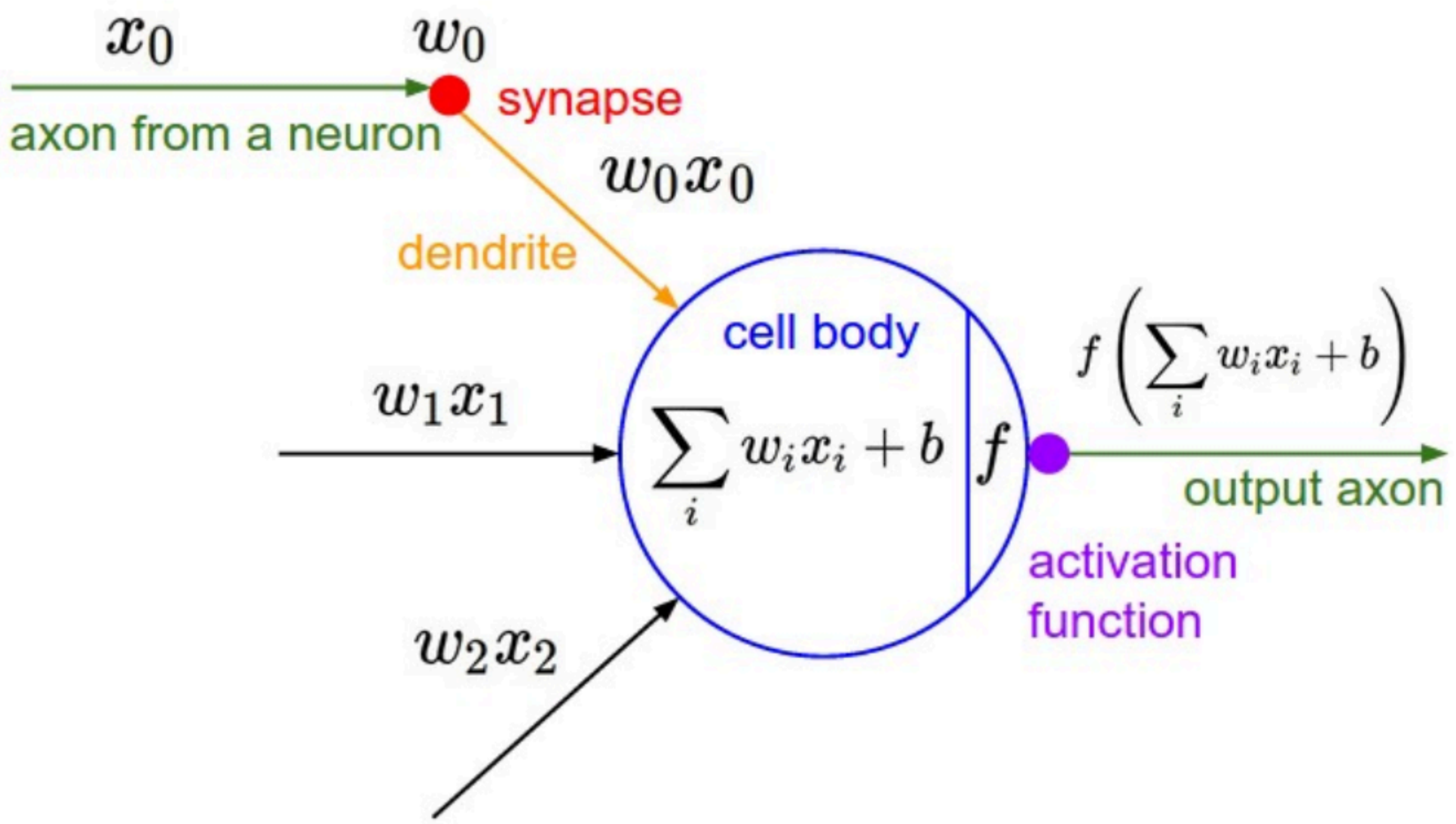


Deep Neural Network - DNN

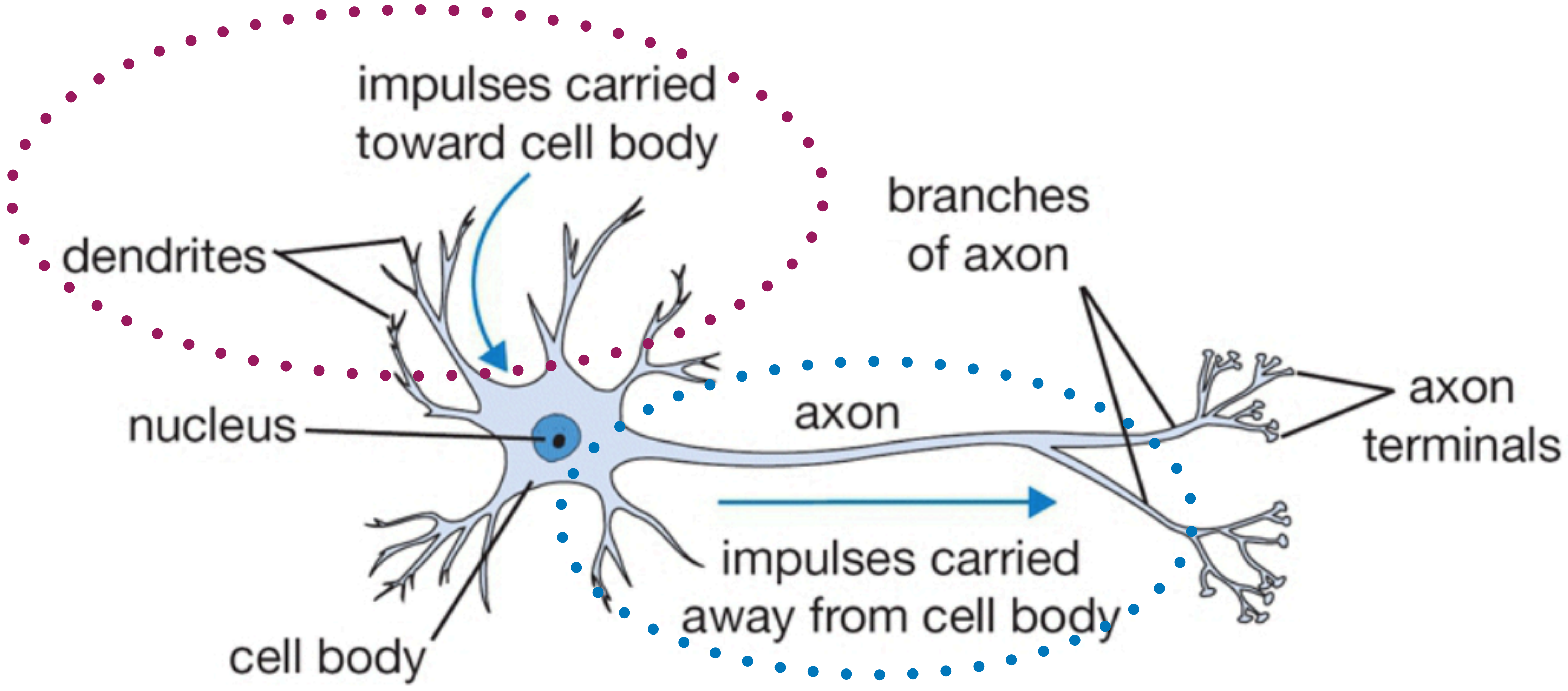
Biological motivations and connections

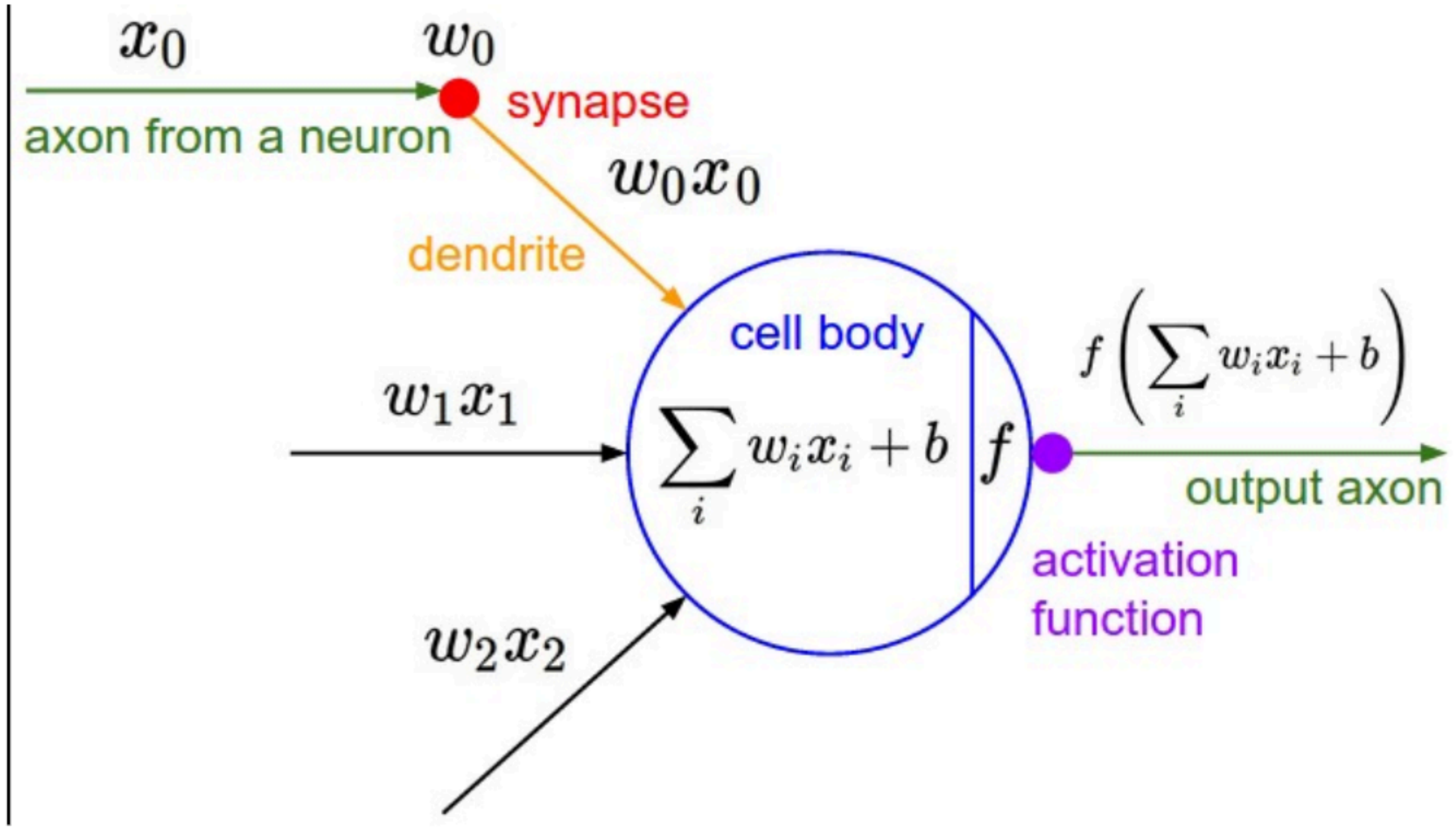
- Basic computational unit of the brain is a **neuron**;
- ~86 billion neurons can be found in the human nervous system and they are connected with approximately 10^{14} - 10^{15} **synapses**;
- Each neuron receives input signals from its **dendrites** and produces output signals along its (single) **axon**;
- axon eventually branches out and **connects via synapses to dendrites of other neurons**.
- In the computational model of a neuron, the signals that travel along the axons (e.g. x_0) interact multiplicatively (e.g. w_0x_0) with the dendrites of the other neuron based on the synaptic strength at that synapse (e.g. w_0);
- idea is that the **synaptic strengths** (the weights *w*) **are learnable and control the strength of influence** (and its direction: **excitory** (positive weight) **or inhibitory** (negative weight)) **of one neuron on another.**





Deep Neural Network - DNN





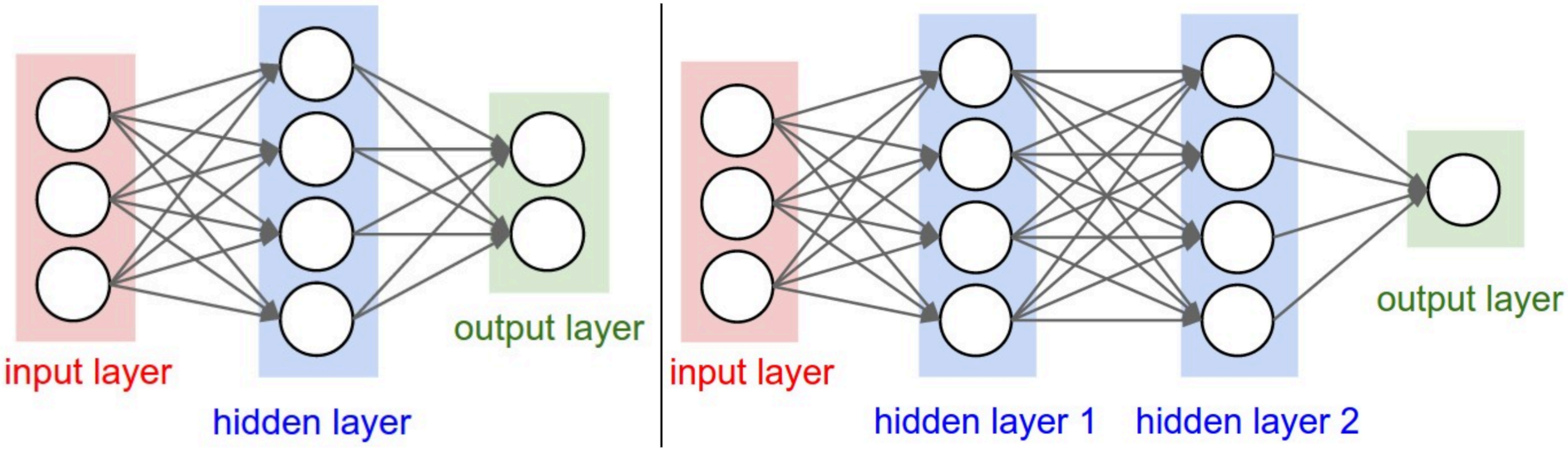
Basic model

- the **dendrites carry the signal to the cell body** where they all get summed.
- If the final sum is above a certain threshold, the neuron can fire, sending a **spike along its axon**.

Computational model

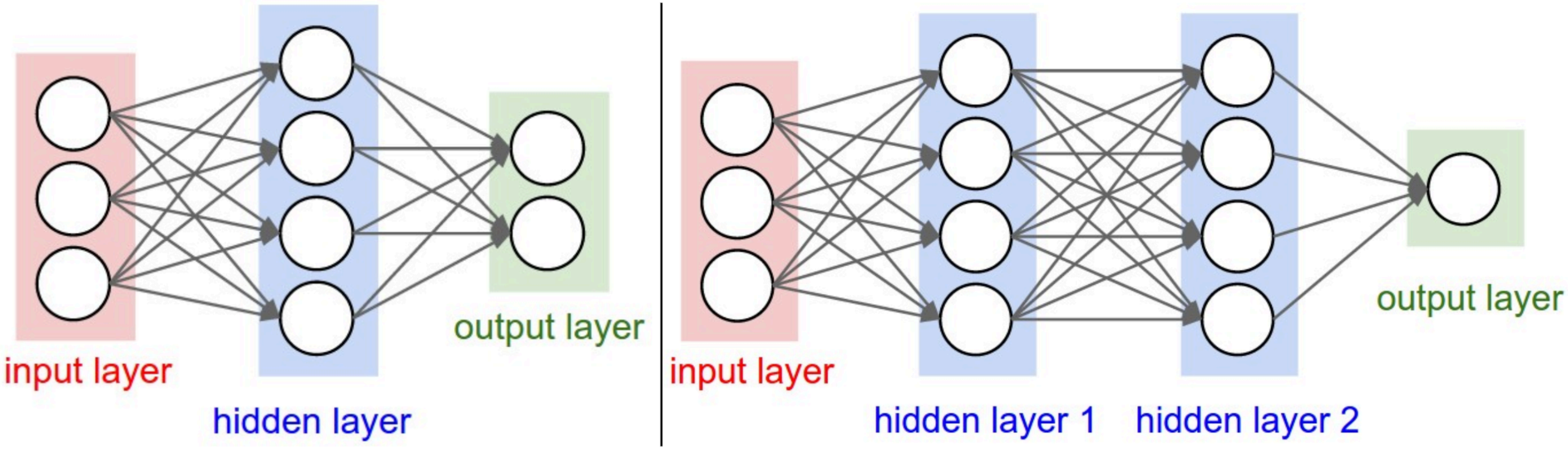
- we assume that the precise timings of the spikes do not matter, and that **only the frequency of the firing communicates information**;
- model the firing rate of the neuron with an **activation function f**, which represents the frequency of the spikes along the axon.

Deep Neural Network - DNN



- Neural Networks are modeled as **collections of neurons** that are connected in an acyclic graph.
- Outputs of some neurons can become inputs to other neurons.
- **Cycles are not allowed** since that would imply an infinite loop in the forward pass of a network:
  - instead of an amorphous blobs of connected neurons, **Neural Network models are often organized into distinct layers of neurons**.
- A most common layer type is the **fully-connected layer** in which neurons between two adjacent layers are fully pairwise connected, but neurons within a single layer share no connections.

Deep Neural Network - DNN



General structure

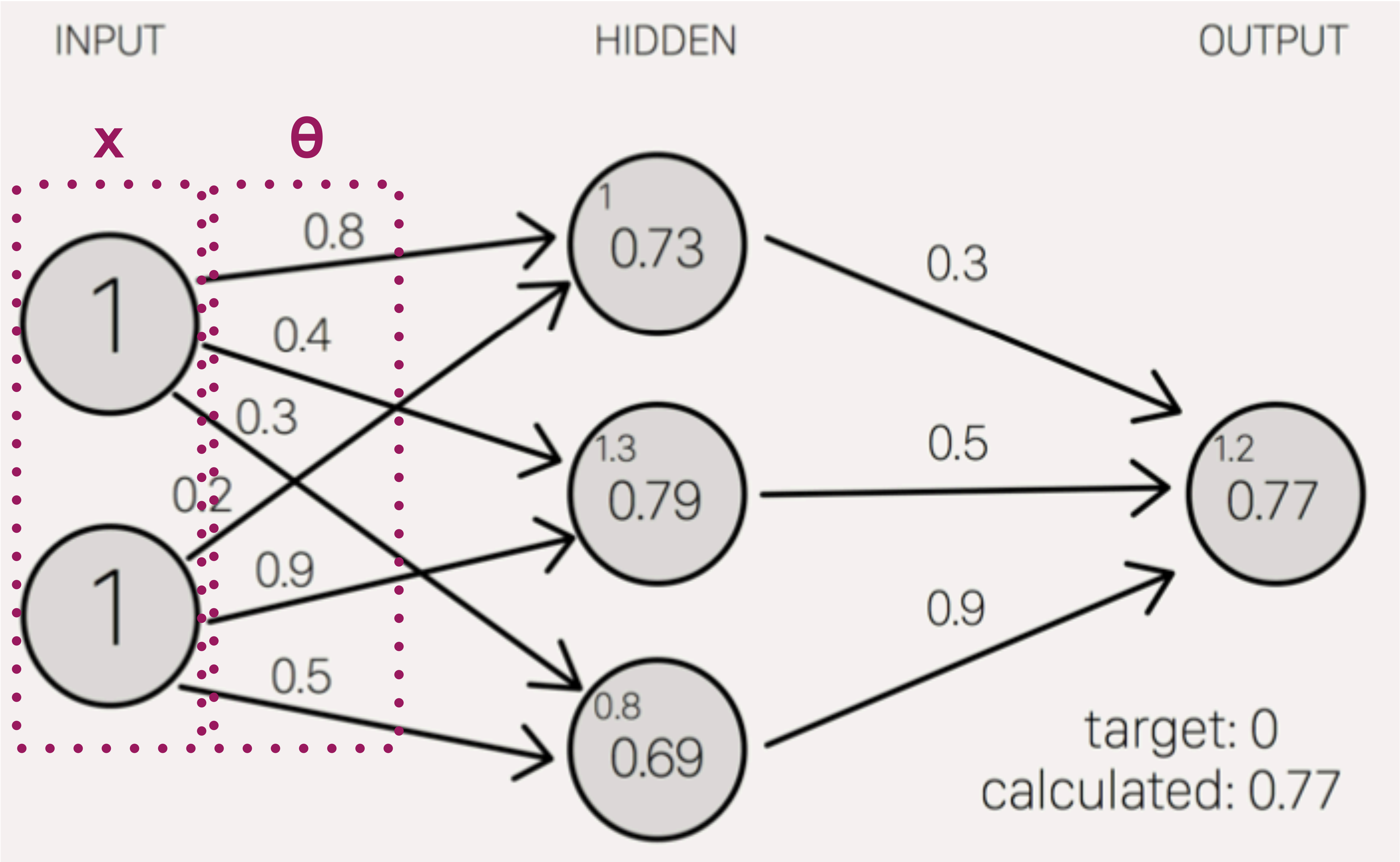
- Each layer consists of many nodes;
- **input layer has one node per input feature**;
- output layer can have **as many outputs as desired**:
  - just one for the case of binary classification or one per class for multi class output;
- each **hidden layer** in between can have an **arbitrary number of nodes**.

- Connections between each of these nodes are the associated parameters **θ (weights)**:
  - no prior intuition for the initialisation of weights;
  - **θ can be initialised randomly**, and learning happens via the **updating of the weights after seeing some training data (back propagation)**.

Deep Neural Network - DNN

Training a Neural Network

- series of forward passes:
  - vector of **inputs x** is multiplied by the **weights θ** connecting the first hidden layer;

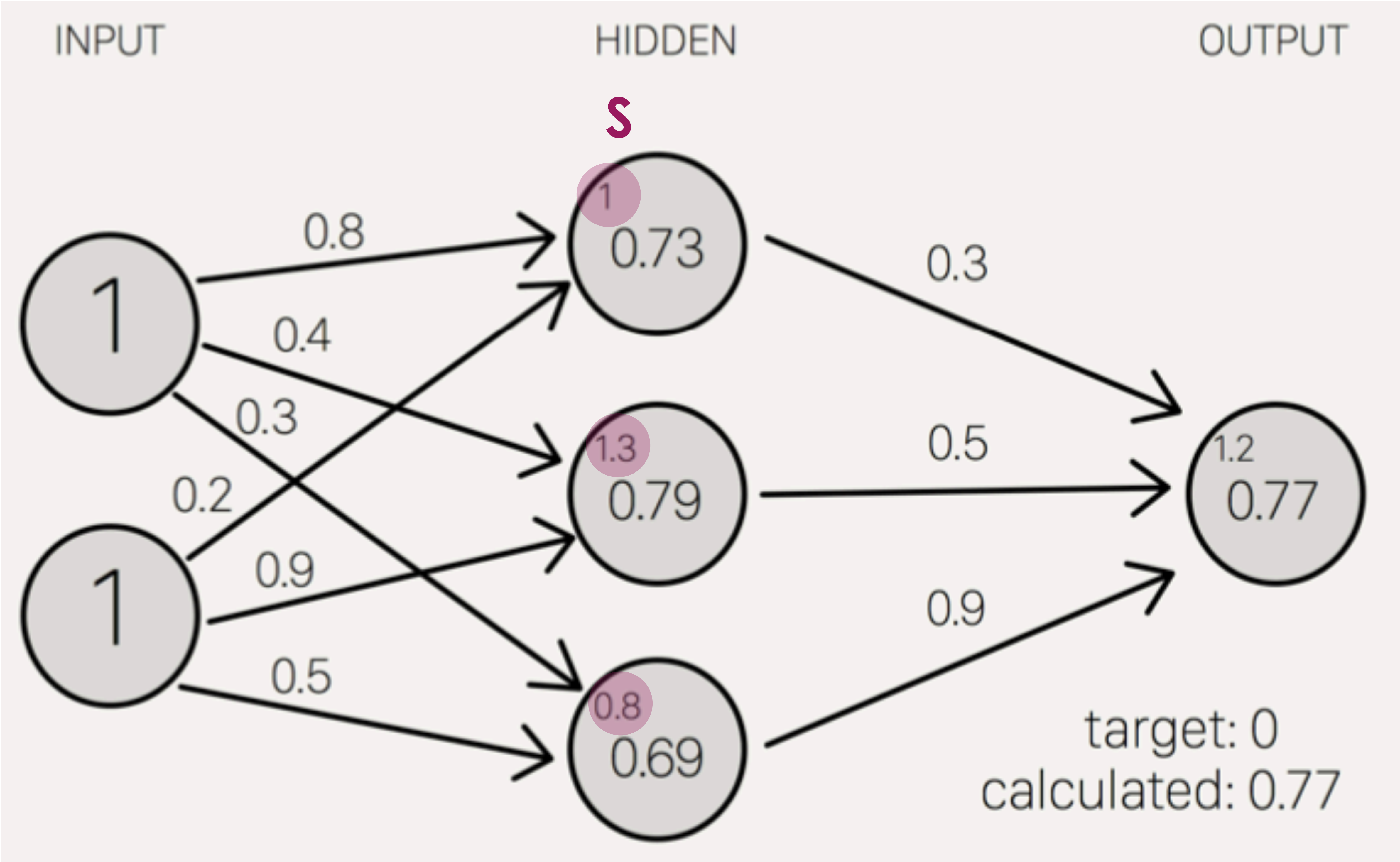


Deep Neural Network - DNN

Training a Neural Network

- series of forward passes:
  - vector of inputs x is multiplied by the weights θ connecting the first hidden layer;

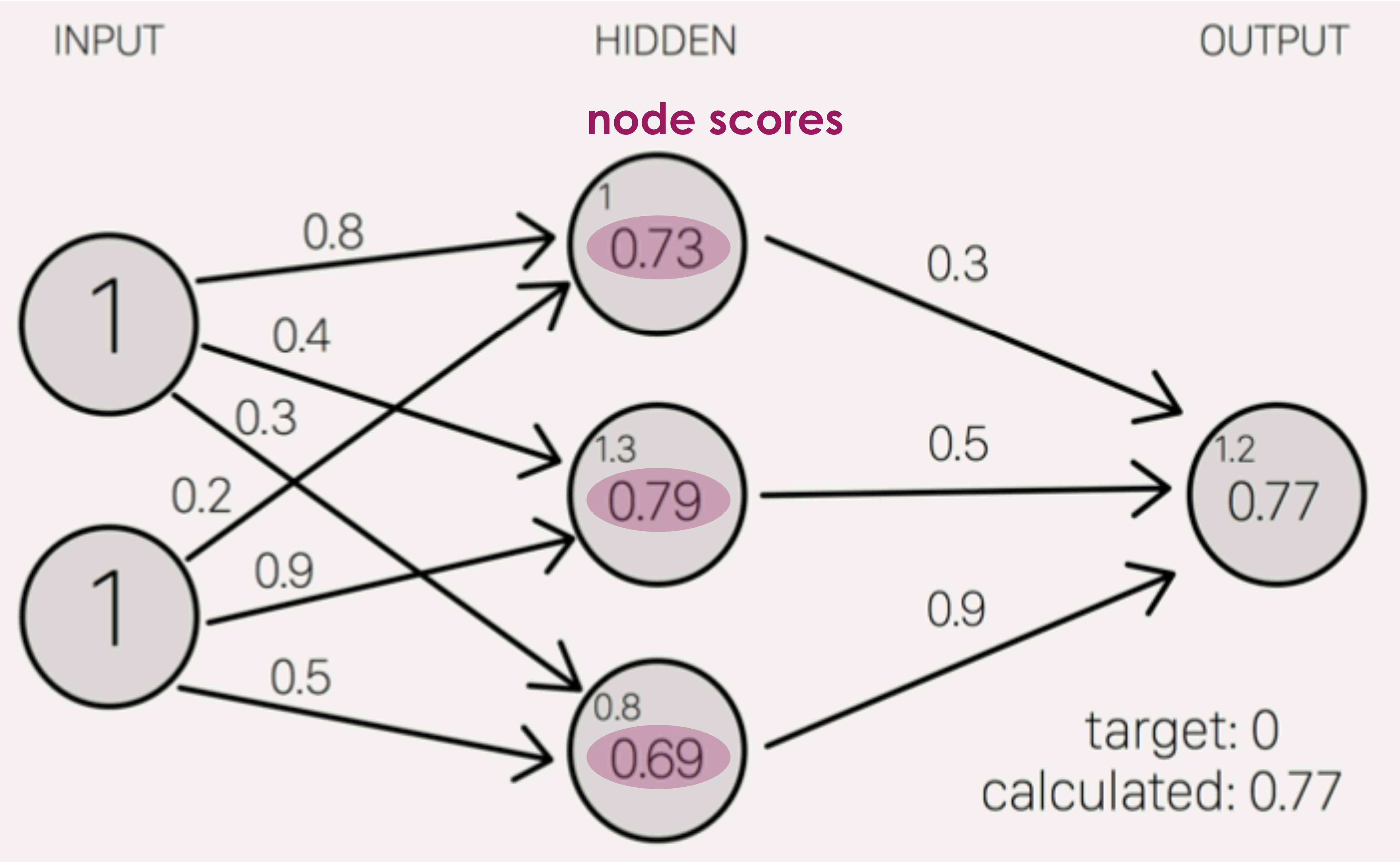
  - all connections to each node in the hidden layer are **summed (S)**;



Deep Neural Network - DNN

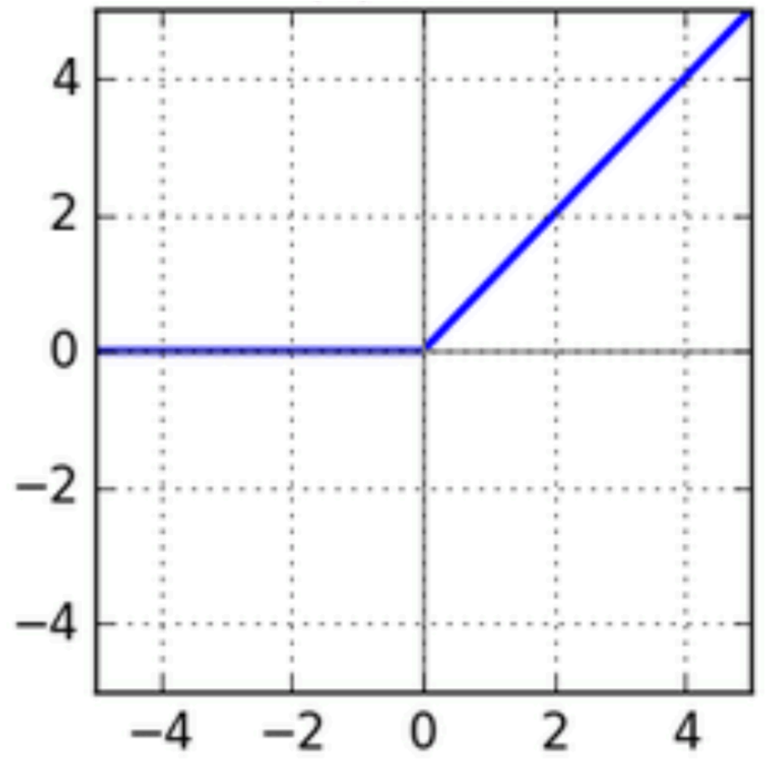
Training a Neural Network

- series of forward passes:
  - vector of inputs x is multiplied by the weights θ connecting the first hidden layer;
  - all connections to each node in the hidden layer are summed (S);
  - S is passed through the **activation function**, to calculate the **node scores**:
    - analogous to an input feature from the input layer.



Activation function

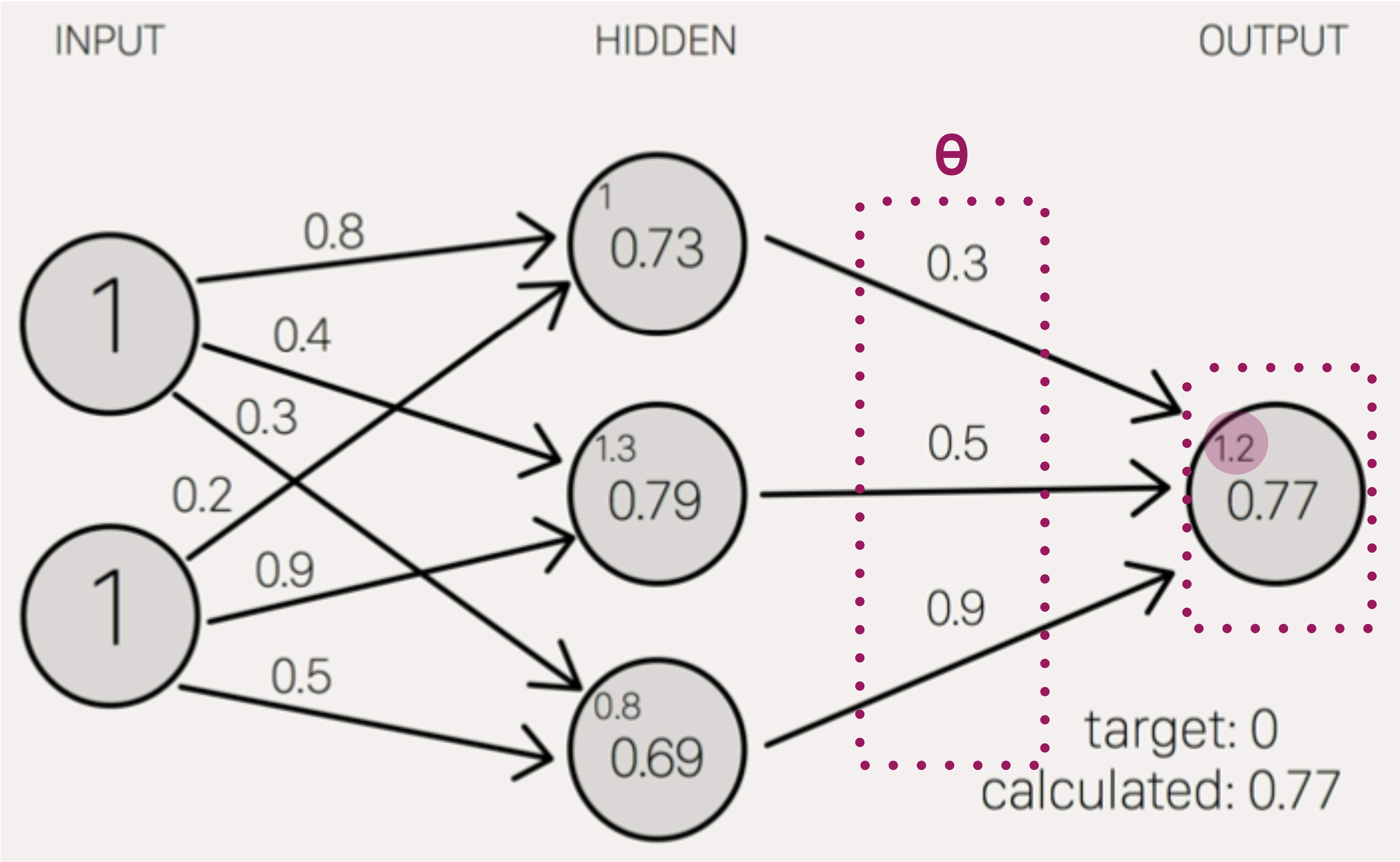
- mathematical function used to transform the inputs;
- The **Rectified Linear Unit** (**ReLU**) computes the function **f(x)=max(0,x)**:
  - greatly accelerate the convergence of stochastic gradient descent compared to other functions;
  - can be implemented by simply thresholding a matrix of activations at zero.



Deep Neural Network - DNN

Training a Neural Network

- series of forward passes:
  - vector of inputs x is multiplied by the weights θ connecting the first hidden layer;
  - all connections to each node in the hidden layer are summed (S);
  - S is passed through the activation function, to calculate the node scores:
    - analogous to an input feature from the input layer;
  - subsequently passed forward identically until reaching the output layer, where the node scores are finally the actual **network output**.

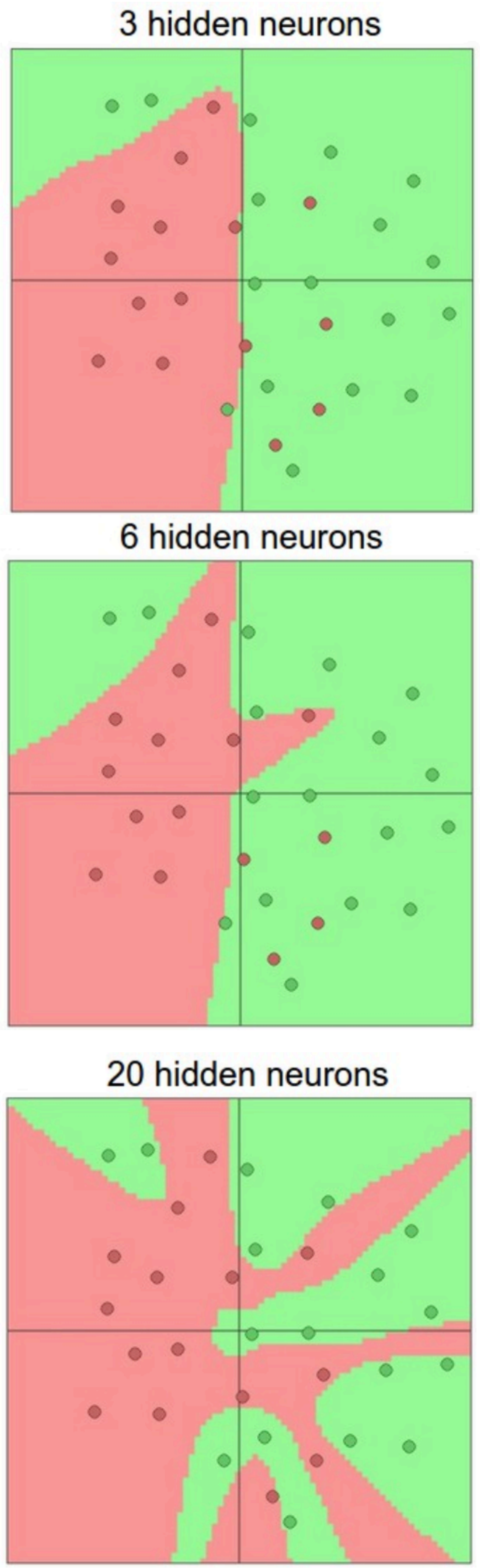


Deep Neural Network - DNN

Setting number of layers and their sizes

- increasing the size and number of layers in a NN, the capacity of the network increases.

- **NN with more neurons can express more complicated functions**;



binary classification problem

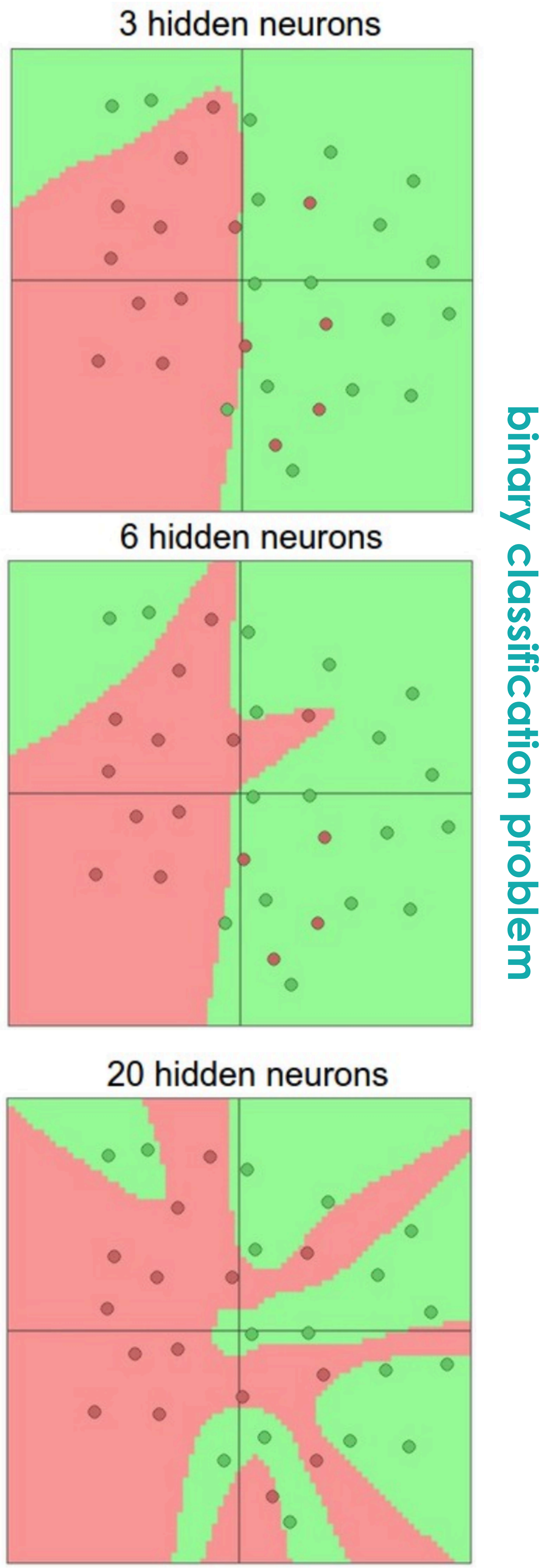
Deep Neural Network - DNN

Setting number of layers and their sizes

- increasing the size and number of layers in a NN, the capacity of the network increases.

- **NN with more neurons can express more complicated functions**;

- **Overfitting** occurs when a model with high capacity fits the noise in the data instead of the (assumed) underlying relationship:
  - model with 20 hidden neurons fits all the training data but at the cost of **segmenting the space into many disjoint red and green decision regions**;



Deep Neural Network - DNN

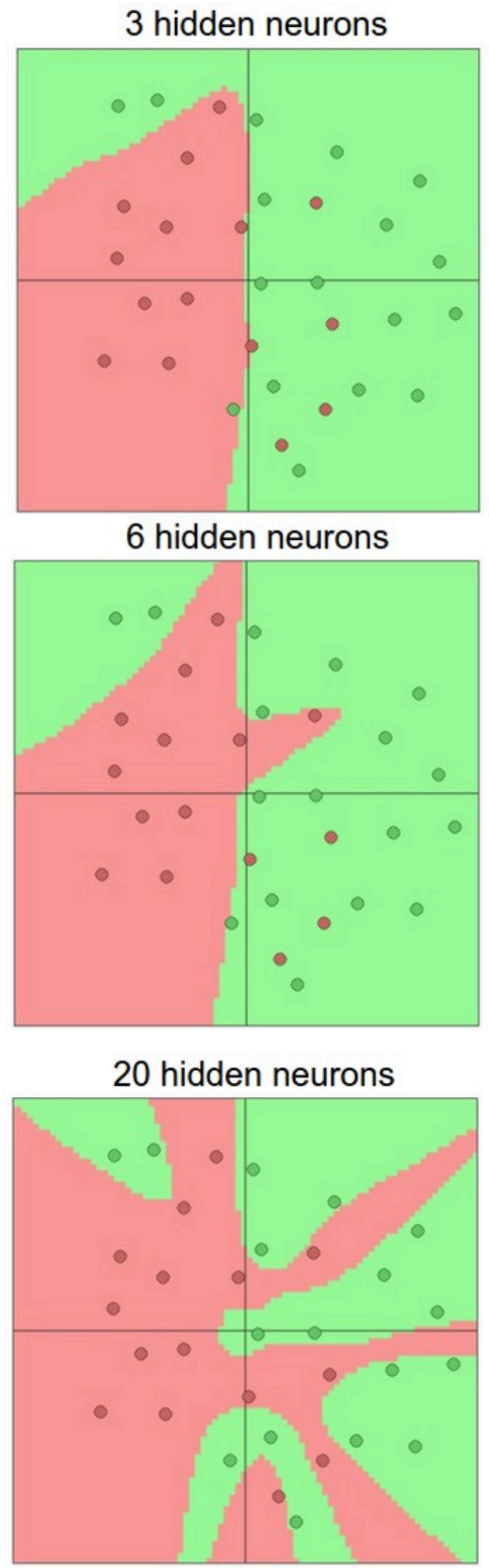
Setting number of layers and their sizes

- increasing the size and number of layers in a NN, the capacity of the network increases.

- **NN with more neurons can express more complicated functions**;

- **Overfitting** occurs when a model with high capacity fits the noise in the data instead of the (assumed) underlying relationship:
  - model with 20 hidden neurons fits all the training data but at the cost of **segmenting the space into many disjoint red and green decision regions**;
  - model with 3 hidden neurons only has the representational power to classify the data in broad strokes:
    - models the data as **two blobs** and interprets the **few red points inside the green cluster as outliers (noise)**;
    - could lead to better **generalisation** on the test set.

- **always better to use hyper-parameters to control overfitting instead of the number of neurons**.



binary classification problem

Deep Neural Network - DNN

Setting number of layers and their sizes

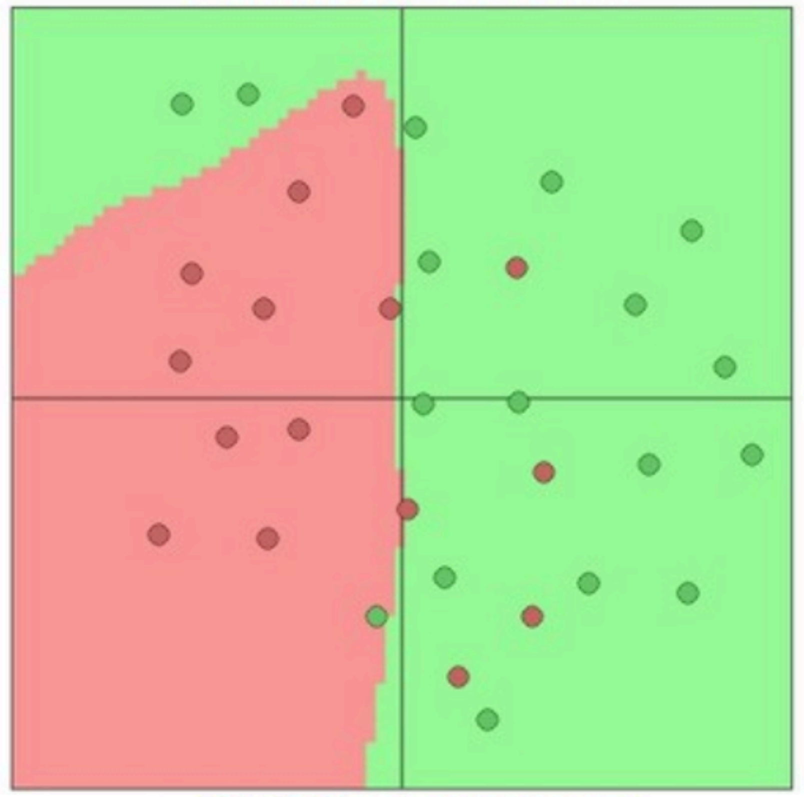
- increasing the size and number of layers in a NN, the capacity of the network increases.
- **NN with more neurons can express more complicated functions**;
- **Overfitting** occurs when a model with high capacity fits the noise in the data instead of the (assumed) underlying relationship:
  - model with 20 hidden neurons fits all the training data but at the cost of **segmenting the space into many disjoint red and green decision regions**;
  - model with 3 hidden neurons only has the representational power to classify the data in broad strokes:
    - models the data as **two blobs** and interprets the **few red points inside the green cluster as outliers (noise)**;
    - could lead to better **generalisation** on the test set.
- **always better to use hyper-parameters to control overfitting instead of the number of neurons**.

Hyper-parameters

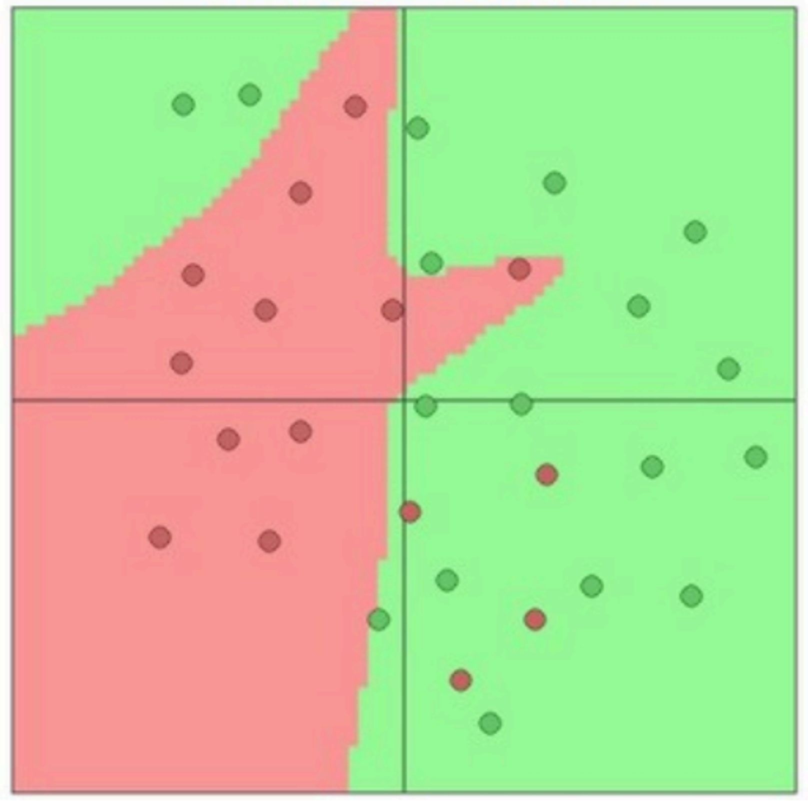
- **batch size**: number of events per update (in the back propagation);
- **epoch**: when all of the data has been passed through the network;
- **dropout**: where connections between nodes are randomly dropped for each batch;

number of layers, number of nodes in each of those layers, amount of dropout and choice of activation functions are all important choices with no a-priori favoured values.

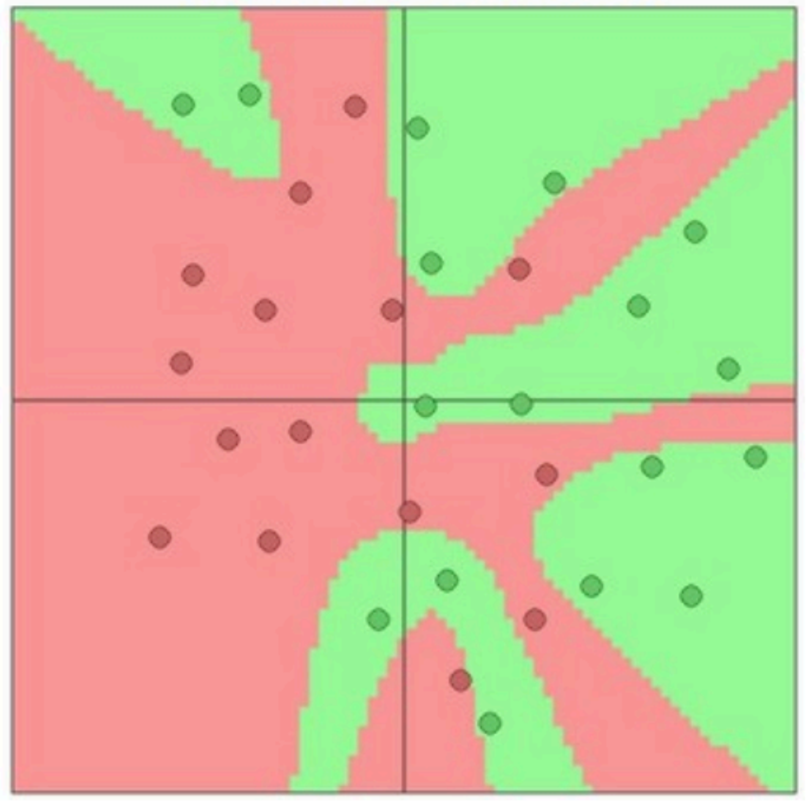
3 hidden neurons



6 hidden neurons



20 hidden neurons



binary classification problem

Large-R jets: tagging technique with ML algorithms

Top and W tagging

Motivation

- Some of the variables used in tagging techniques contain **complementary information**;
- combining these observables by creating a **multivariate classifier provides higher discrimination**.

Goal

- to discriminate **W-boson and top-quark jets from light jets**;
- to provide a **single jet-tagging discriminant** that is widely applicable in place of the single jet moment to augment the discrimination of m_{comb} alone across a broad p_T range.

→ **widely applicable and more powerful tagger!**

Large-R jets: tagging technique with ML algorithms

[Top and W tagging](#)

Motivation

- Some of the variables used in tagging techniques contain **complementary information**;
- combining these observables by creating a **multivariate classifier provides higher discrimination**.

Goal

- to discriminate **W-boson and top-quark jets from light jets**;
- to provide a **single jet-tagging discriminant** that is widely applicable in place of the single jet moment to augment the discrimination of m_{comb} alone across a broad p_T range.

→ **widely applicable and more powerful tagger!**

Strategy

- **BDT** and **DNN** are used to study the performances of the taggers;
- For the design of all multivariate discriminants, **exclusive subsamples of signal and background jets** are derived from the more inclusive sample to be used separately for the training and testing of the discriminant;
- all studies are performed in a wide p_T^{true} bin:
  - **[200,2000] GeV** for W boson tagging;
  - **[350,2000] GeV** for top quark tagging;
- Input variables chosen by comparing the performance when using different sets of input variables **to find the set of observables which gives the largest relative background rejection at a fixed relative signal efficiency**.

Large-R jets: tagging technique with ML algorithms

Top and W tagging

Training

- event selection to isolate ensembles of jets which are representative of those originating from either W bosons or top quarks (**signal**) and gluon or other (non-top) quarks (**background**);
- **anti-k_t R = 1.0 jets**, trimming algorithm with R_{sub} = 0.2 and f_{cut} = 5%;
- reco jets with **200 < p_T < 2000 GeV** (W) and **350 < p_T < 2000 GeV** (top).

- **signal jets** are defined as hadronically-decaying W bosons or top quarks when all partonic decay products are fully contained within fixed ΔR:

  1. reconstructed jets are matched to truth jets;
  2. those truth jets are matched to the truth W-boson and top-quark particles (**W, top**);
  3. their partonic decay products (**q_1, q_2, b**) are matched to the initial reconstructed jet.

Purpose	Train		Test	
Tagger type	W boson	Top Quark	W boson	Top Quark
Truth matching	$dR(jet,x) < 0.75$ x=W,q_1,q_2	$dR(jet,x) < 0.75$ x=top,q_1,q_2,b	$dR(jet,x) < 0.75$ x=W,q_1,q_2	$dR(jet,x) < 0.75$ x=top,q_1,q_2,b
p_T weighting	flat	flat	to QCD	to QCD
Truth p_T [GeV]	[200,2000]	[350,2000]	[200,2000]	[350,2000]
m^{calo} [GeV]	> 40	> 40	[60, 100]	> 120
N^{const}	> 2	> 2	> 2	> 2

Large-R jets: tagging technique with ML algorithms

Top and W tagging

Training

- event selection to isolate ensembles of jets which are representative of those originating from either W bosons or top quarks (**signal**) and gluon or other (non-top) quarks (**background**);
- **anti-k_t R = 1.0 jets**, trimming algorithm with R_{sub} = 0.2 and f_{cut} = 5%;
- reco jets with **200 < p_T < 2000 GeV** (W) and **350 < p_T < 2000 GeV** (top).

- **signal jets** are defined as hadronically-decaying W bosons or top quarks when all partonic decay products are fully contained within fixed ΔR:

  1. reconstructed jets are matched to truth jets;
  2. those truth jets are matched to the truth W-boson and top-quark particles (**W, top**);
  3. their partonic decay products (**q_1, q_2, b**) are matched to the initial reconstructed jet.

Purpose	Train		Test	
Tagger type	W boson	Top Quark	W boson	Top Quark
Truth matching	$dR(jet, x) < 0.75$ x=W,q_1,q_2	$dR(jet, x) < 0.75$ x=top,q_1,q_2,b	$dR(jet, x) < 0.75$ x=W,q_1,q_2	$dR(jet, x) < 0.75$ x=top,q_1,q_2,b
p_T weighting	flat	flat	to QCD	to QCD
Truth p_T [GeV]	[200,2000]	[350,2000]	[200,2000]	[350,2000]
m^{calo} [GeV]	> 40	> 40	[60, 100]	> 120
N^{const}	> 2	> 2	> 2	> 2

- **30% of signal and bkg samples** combined and weighted such that the p_T distribution of signal jets matches the **dijets bkg distribution**;
- **to remove any bias on the performance** due to difference in p_T spectrum of signal and bkg jet samples.

Large-R jets: tagging technique with ML algorithms

Top and W tagging

Training

- event selection to isolate ensembles of jets which are representative of those originating from either W bosons or top quarks (**signal**) and gluon or other (non-top) quarks (**background**);
- **anti-k_t R = 1.0 jets**, trimming algorithm with R_{sub} = 0.2 and f_{cut} = 5%;
- reco jets with **200 < p_T < 2000 GeV** (W) and **350 < p_T < 2000 GeV** (top).

- **signal jets** are defined as hadronically-decaying W bosons or top quarks when all partonic decay products are fully contained within fixed ΔR:

  1. reconstructed jets are matched to truth jets;
  2. those truth jets are matched to the truth W-boson and top-quark particles (**W, top**);
  3. their partonic decay products (**q_1, q_2, b**) are matched to the initial reconstructed jet.

Purpose	Train		Test	
Tagger type	W boson	Top Quark	W boson	Top Quark
Truth matching	$dR(jet,x) < 0.75$ x=W,q_1,q_2	$dR(jet,x) < 0.75$ x=top,q_1,q_2,b	$dR(jet,x) < 0.75$ x=W,q_1,q_2	$dR(jet,x) < 0.75$ x=top,q_1,q_2,b
p_T weighting	flat	flat	to QCD	to QCD
Truth p_T [GeV]	[200,2000]	[350,2000]	[200,2000]	[350,2000]
m^{calo} [GeV]	> 40	> 40	[60, 100]	> 120
N^{const}	> 2	> 2	> 2	> 2

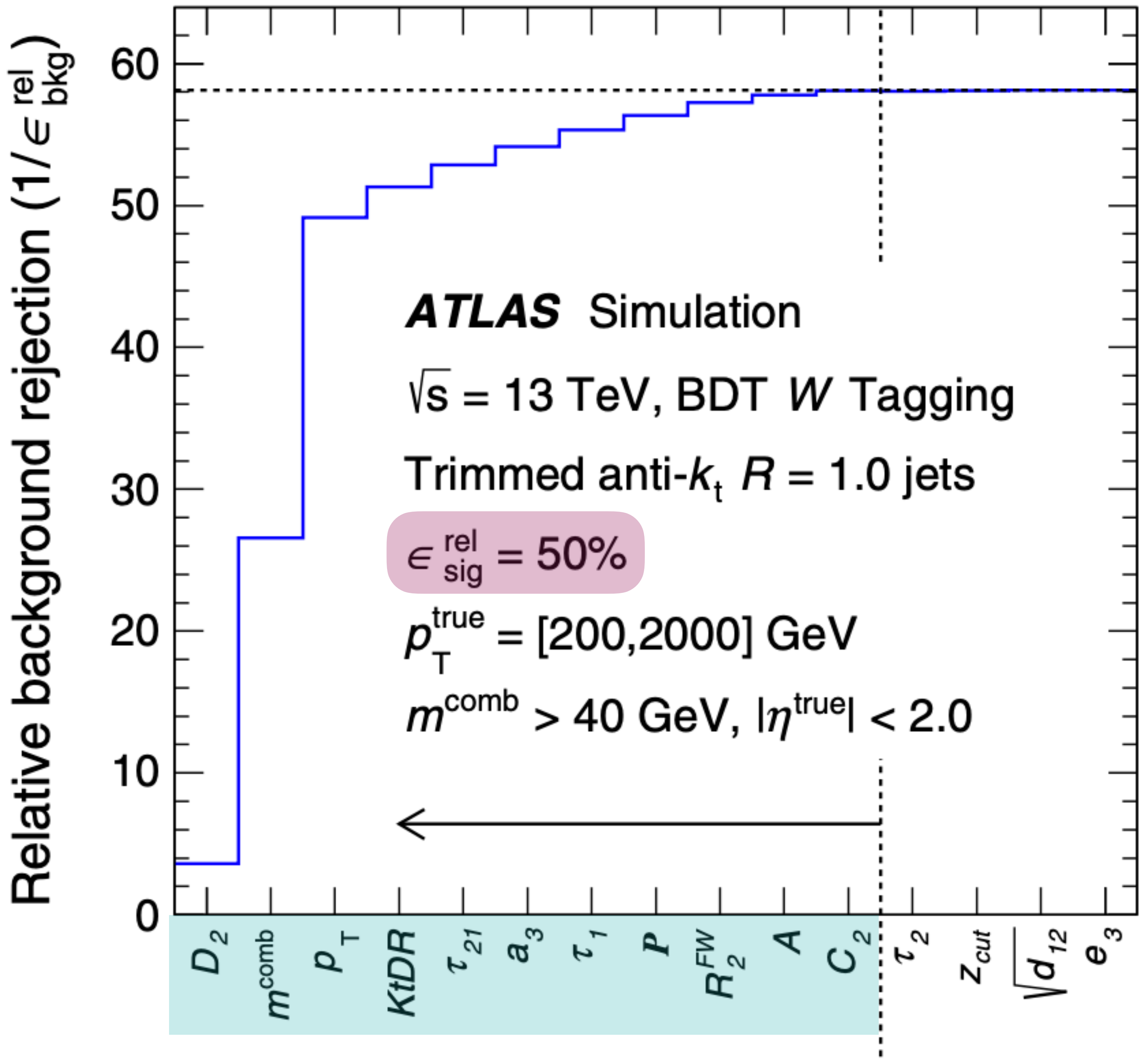
Purpose	Training (Top quark)		Training (W boson)		Testing		
Sample	Signal	Background	Signal	Background	Top Signal	W Signal	Background
Number of jets	10^6	10^6	7×10^5	7×10^5	4×10^5	3×10^5	1×10^6

Large-R jets: tagging technique with ML algorithms

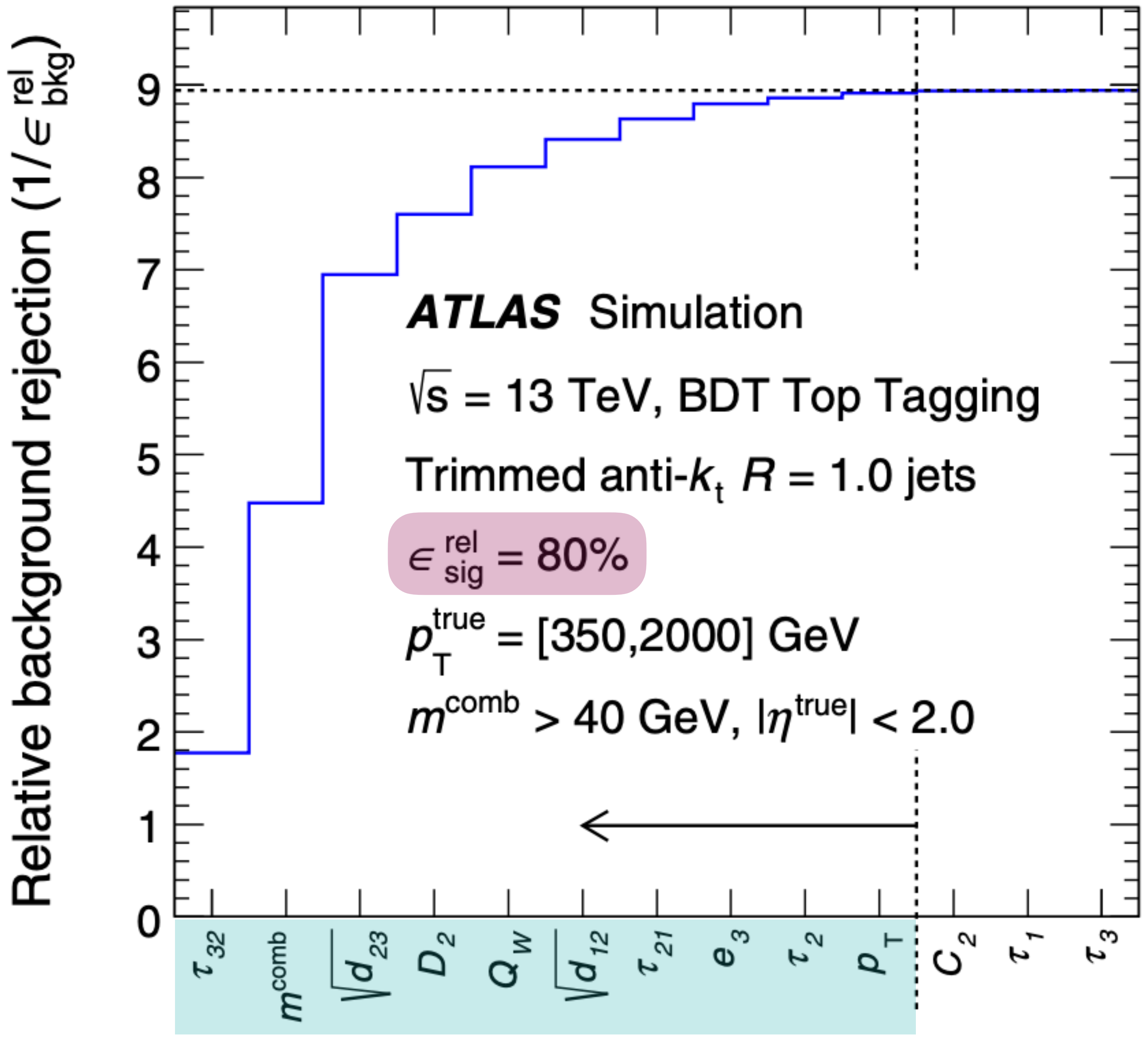
Top and W tagging

Choice of variables for BDT

- Fixed relative signal efficiency of **50%** (W-boson tagging) and **80%** (top-quark tagging);
- observables which give the largest increase in relative performance are **sequentially added** to the network;
- the smallest set of variables which reaches the **highest relative background rejection within statistical uncertainties** is selected.



11



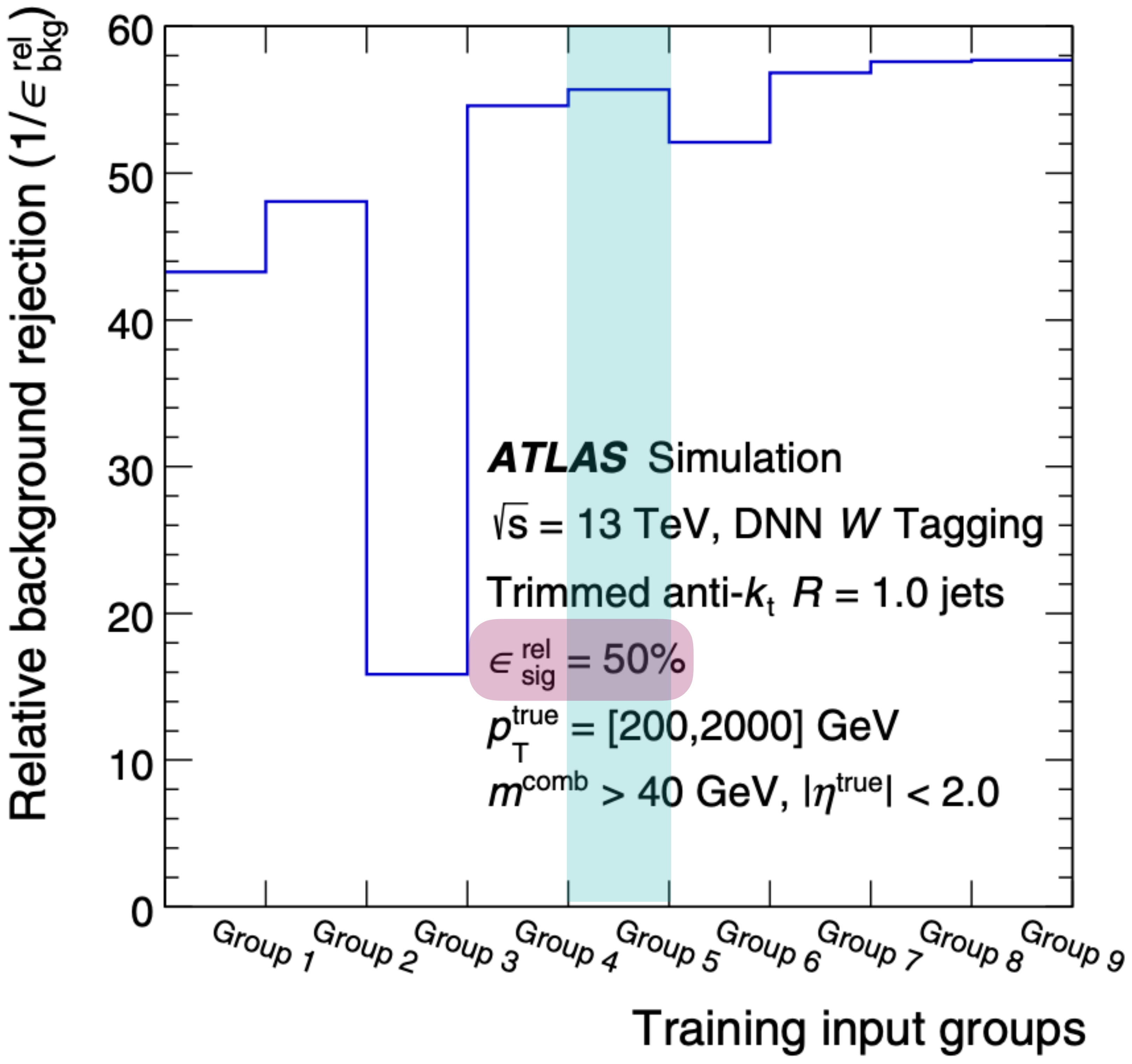
10

Large-R jets: tagging technique with ML algorithms

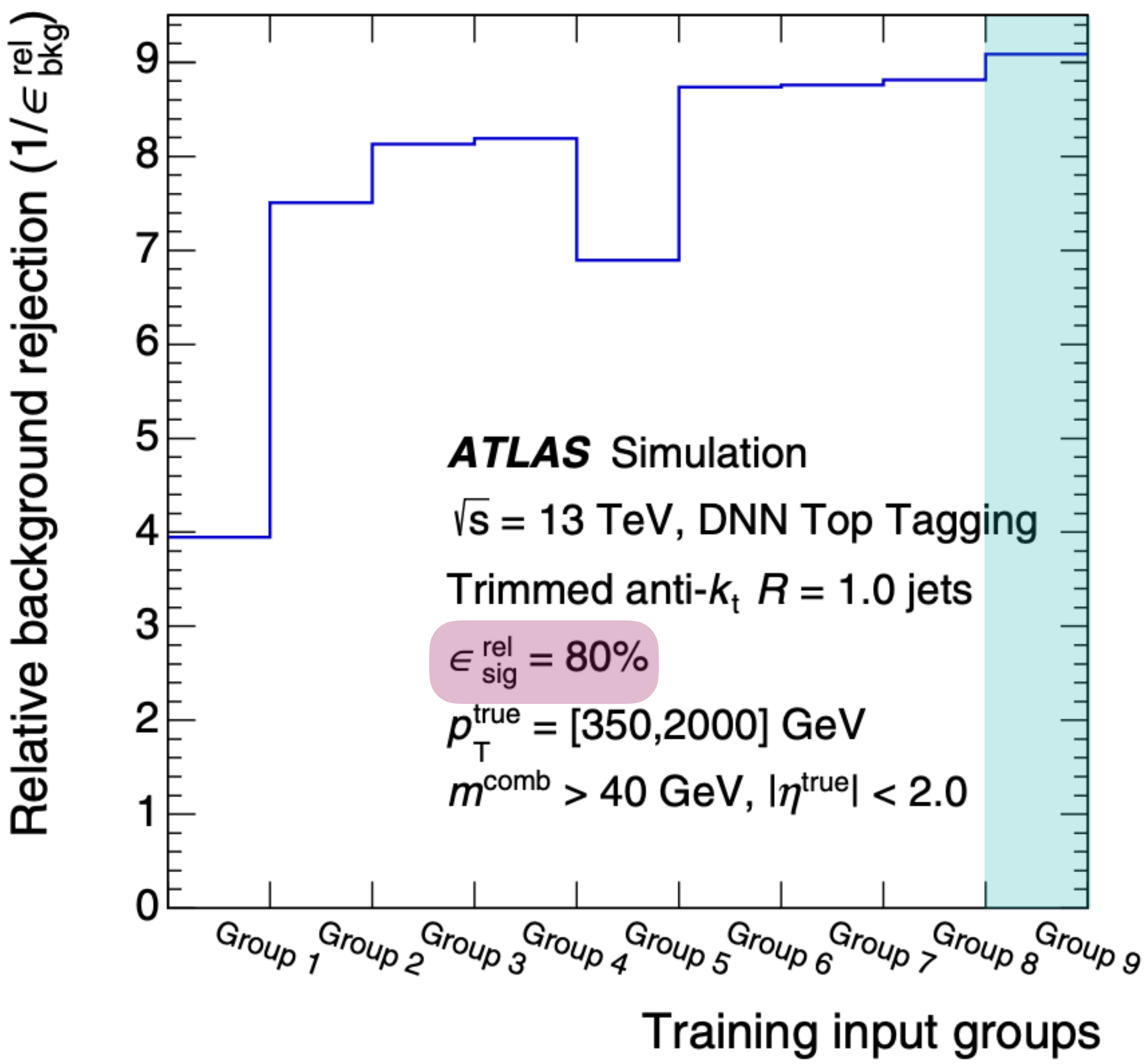
Top and W tagging

Choice of variables for DNN

- Fixed relative signal efficiency of **50%** (W-boson tagging) and **80%** (top-quark tagging);
- variables are **not added in succession** due to the time requirements to train the large number of networks;
- chosen by selecting variables according to their dependence on the **momentum scale of the jet substructure objects**, what features of the **substructure** they describe and their **dependence on other substructure variables**.



12



13

Large-R jets: tagging technique with ML algorithms

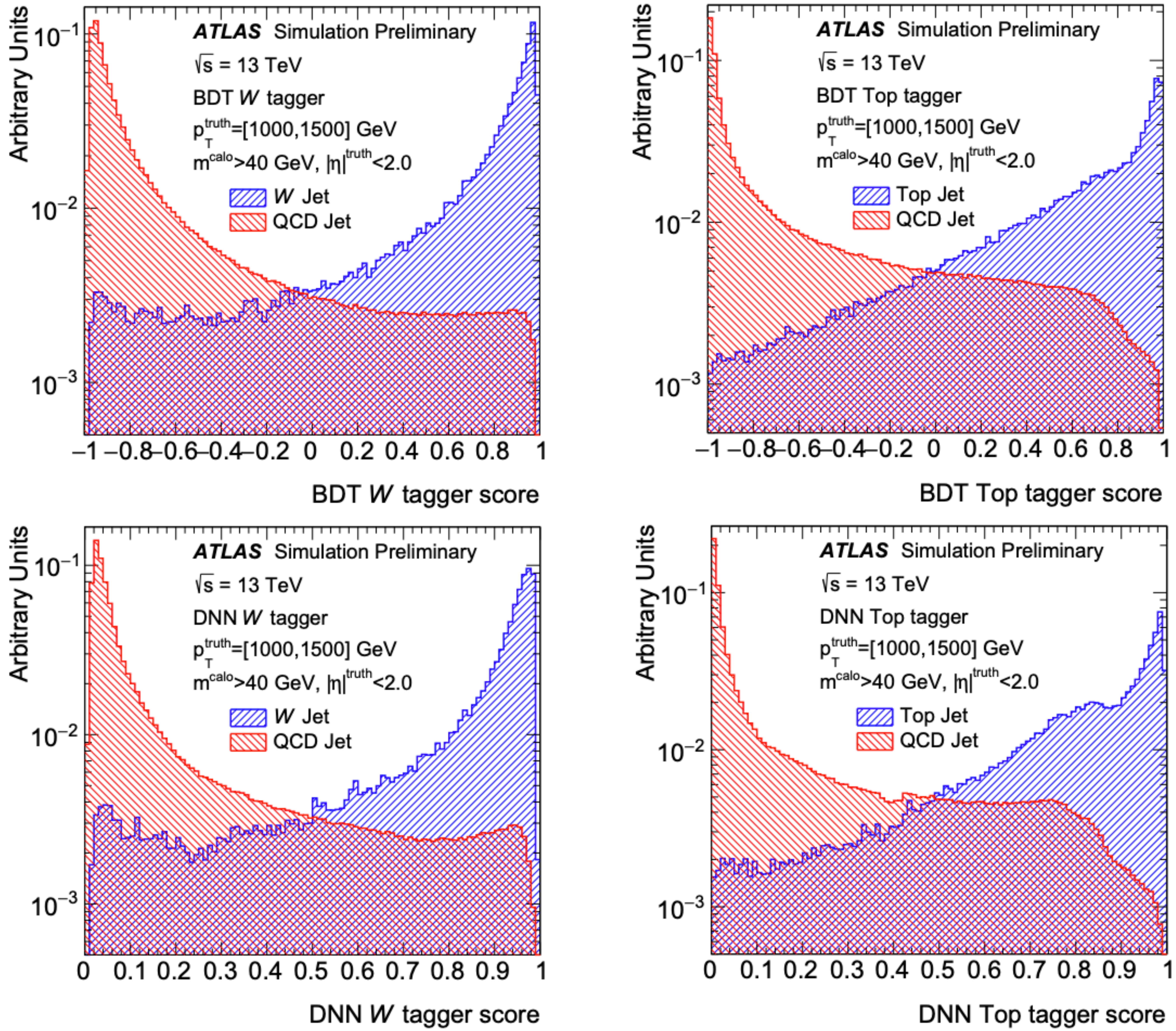
Results and final choice

- BDT and DNN algorithms result in a **single discriminant that allows for the classification of a jet** as either a top-quark or gluon/other (non-top) quark jet and a W-boson or gluon/other (non-top) quark jet;

- **performance of the two new taggers** is studied more quantitatively in two ways:

  1. discrimination power of the BDTs and DNNS are **compared with respect to the simple reference taggers** to estimate the gain expected by adding more variables and taking advantage of non-linear correlations;

  2. performance of BDTs and DNNs are **compared with each other** to determine if one algorithm is able to extract more information than the other.



Large-R jets: tagging technique with ML algorithms

Top and W tagging

Performances of the taggers

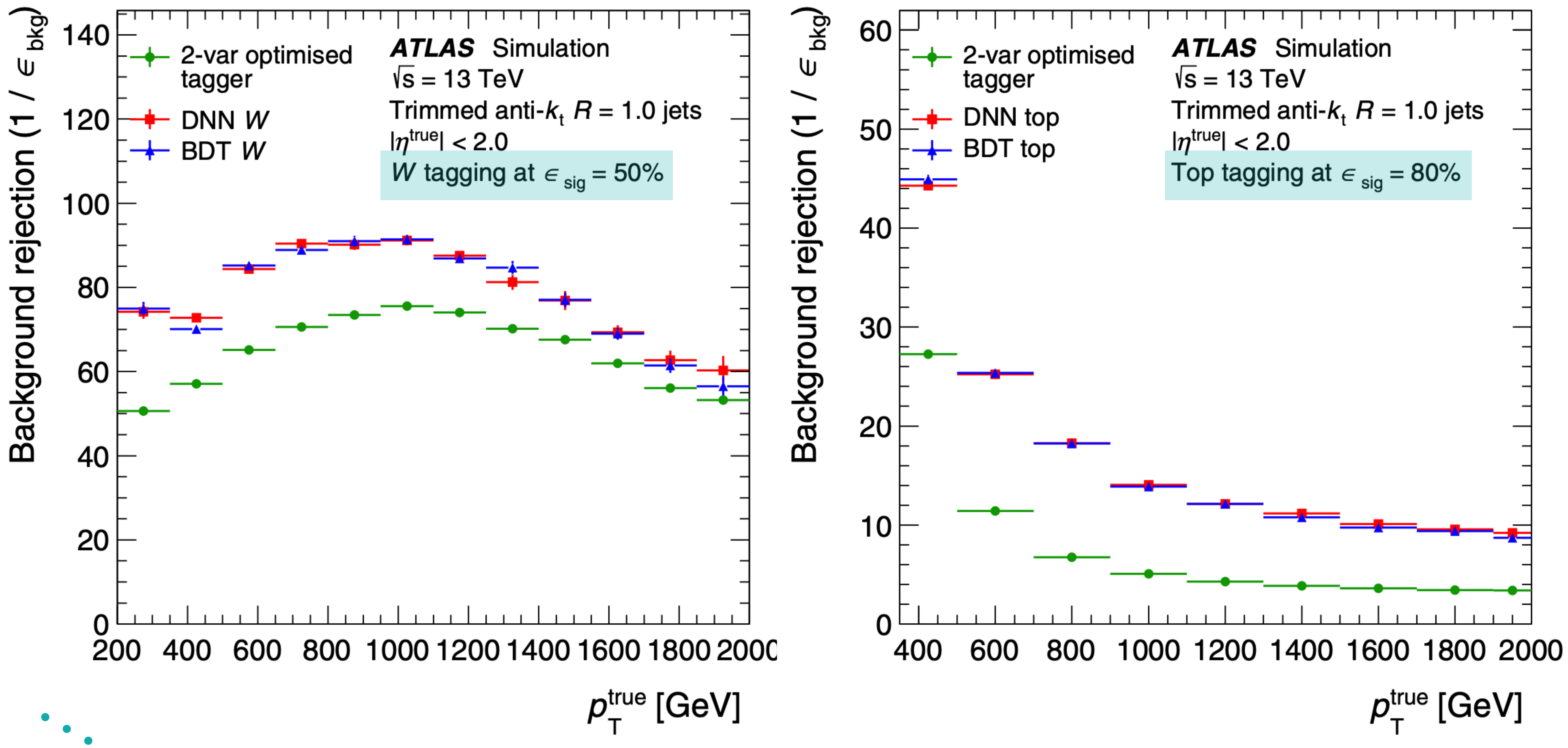
- characterised by the background rejection, evaluated **as a function of jet p_T^{true}**, for a fixed signal efficiency of 50% (W-boson tagging) and 80% (top-quark tagging).

Results for W-tagging:

- performance **improvements** beyond the cut-based taggers are **highest at low jet p_T and decrease at higher p_T^{true}**;
- due to the merging of calorimeter energy depositions and subsequent **loss of granularity** in discerning substructure information;

Results for top-tagging:

- **improvements** in performance are more sizeable, showing increases in background rejection of roughly **a factor of two over the entire kinematic range studied**;
- due to the **greater complexity of the top decay** in contrast to that of the isolated W.



among the observables studied here, no single observable adequately captures alone the full set of features that provide ability to discriminate signal from background!

Supporting material

Large-R jets: tagging technique with ML algorithms

	W boson tagging											Top quark tagging										
	DNN Test groups									Chosen inputs		DNN test groups									Chosen inputs	
Observable	1	2	3	4	5	6	7	8	9	BDT	DNN	1	2	3	4	5	6	7	8	9	BDT	DNN
m^{comb}	○	○		○	○	○	○	○	○	○	○		○	○	○		○	○	○	○	○	○
p_{T}	○	○			○	○		○	○	○	○			○	○			○	○	○	○	○
e_3	○	○				○			○						○			○		○	○	○
C_2			○	○	○		○	○	○		○	○	○	○		○	○		○	○		○
D_2			○	○	○		○	○	○	○	○	○	○	○		○	○		○	○	○	○
τ_1	○	○				○			○	○					○			○		○		○
τ_2	○	○				○			○						○			○		○	○	○
τ_3															○			○		○		○
τ_{21}			○	○	○		○	○	○	○	○	○	○	○		○	○		○	○	○	○
τ_{32}												○	○	○		○	○		○	○	○	○
R_2^{FW}			○	○	○	○	○	○	○	○	○											
$\mathcal{P}$			○	○	○	○	○	○	○	○	○											
a_3			○	○	○	○	○	○	○	○	○											
A			○	○	○	○	○	○	○	○	○											
z_{cut}			○	○	○		○	○	○		○											
$\sqrt{d_{12}}$		○				○	○	○	○	○	○					○	○	○	○	○	○	○
$\sqrt{d_{23}}$																○	○	○	○	○	○	○
$KtDR$		○				○	○	○	○	○	○											
Q_w																○	○	○	○	○	○	○

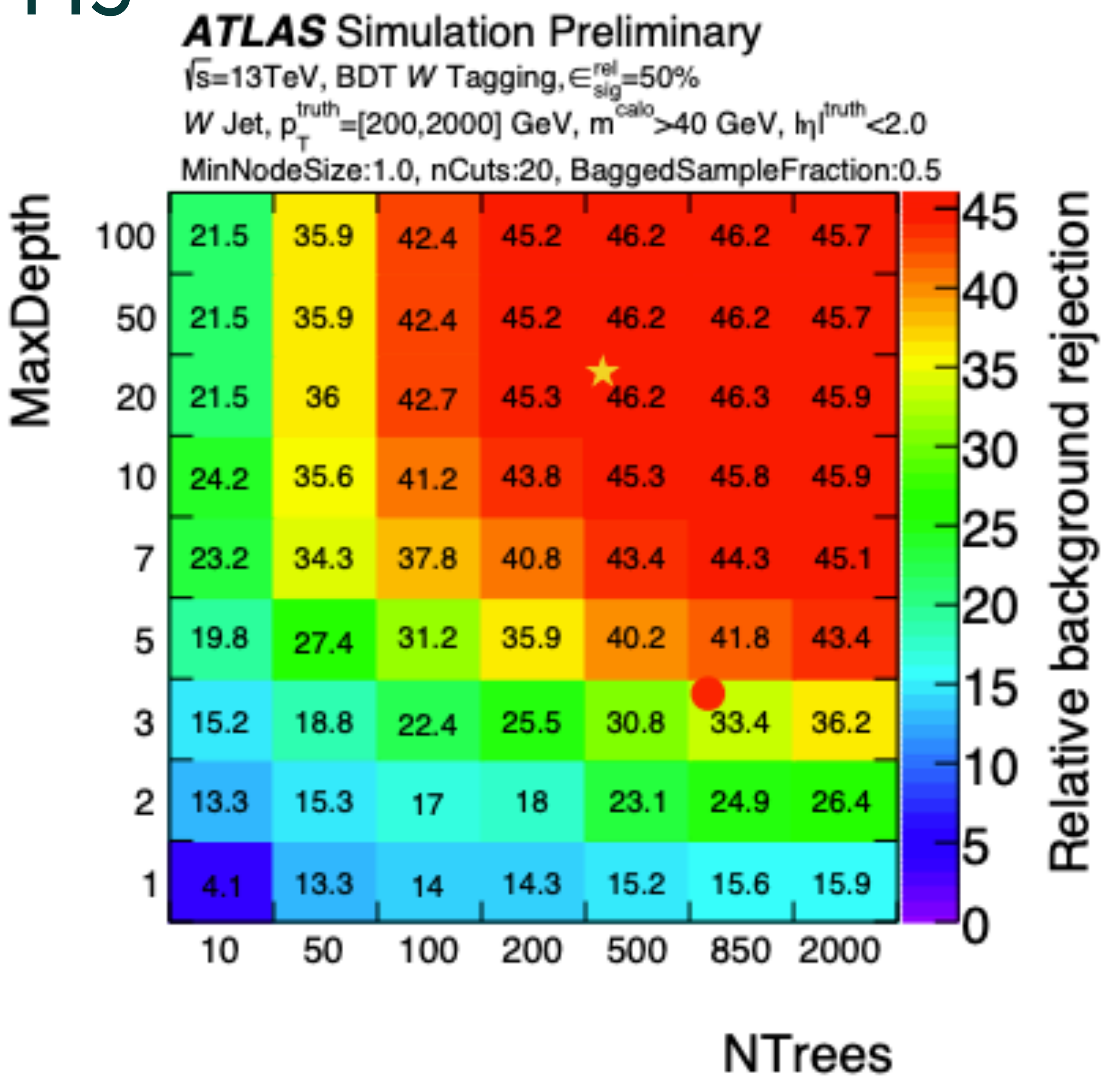
Large-R jets: tagging technique with ML algorithms

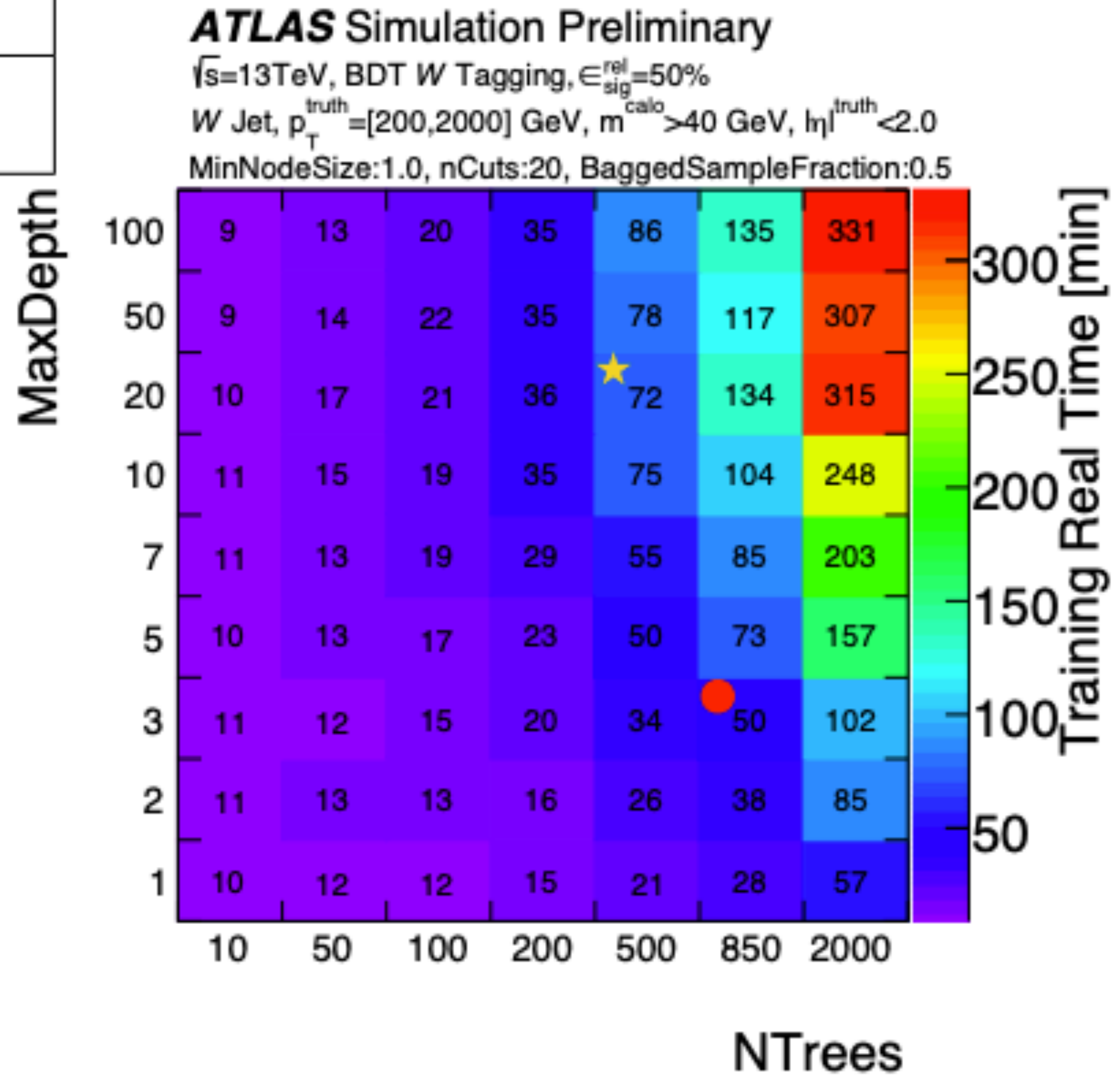
	W-Boson Tagging		Top-Quark Tagging	
Observable	BDT	DNN	BDT	DNN
ECF_1		$\circ$	$\circ$	$\circ$
ECF_2	$\circ$	$\circ$		$\circ$
ECF_3	$\circ$	$\circ$	$\circ$	$\circ$
C_2		$\circ$	$\circ$	$\circ$
D_2	$\circ$	$\circ$	$\circ$	$\circ$
τ_1	$\circ$	$\circ$		$\circ$
τ_2		$\circ$	$\circ$	$\circ$
τ_3				$\circ$
τ_{21}	$\circ$	$\circ$	$\circ$	$\circ$
τ_{32}			$\circ$	$\circ$
R_2^{FW}	$\circ$	$\circ$		
S	$\circ$	$\circ$		
$\mathcal{P}$	$\circ$	$\circ$		
$\mathcal{D}$		$\circ$		
a_3	$\circ$	$\circ$		
A	$\circ$	$\circ$		
T_{MIN}				
T_{MAJ}				
Z_{CUT}		$\circ$		
μ_{12}		$\circ$		
$\sqrt{d_{12}}$		$\circ$	$\circ$	$\circ$
$\sqrt{d_{23}}$			$\circ$	$\circ$
$KtDR$	$\circ$	$\circ$		
Q_w			$\circ$	$\circ$

Large-R jets: tagging technique with ML algorithms

Setting Name	Description	Chosen Value
BoostType	Type of boosting technique	GradientBoost
NTrees	Number of trees in the forest	500
MaxDepth	Max depth of the decision tree allowed	20
MinimumNodeSize	Minimum fraction of training events required in a leaf node	1.0%
Shrinkage	Learning rate for GradientBoost algorithm	0.5
UseBaggedBoost	Use only a random (bagged) subsample of all events for growing the trees in each iteration	True
BaggedSampleFraction	Relative size of bagged event sample to original size of the data sample	0.5
SeparationType	Separation criterion for node splitting	GiniIndex
nCuts	Number of grid points in variable range used in finding optimal cut in node splitting	500

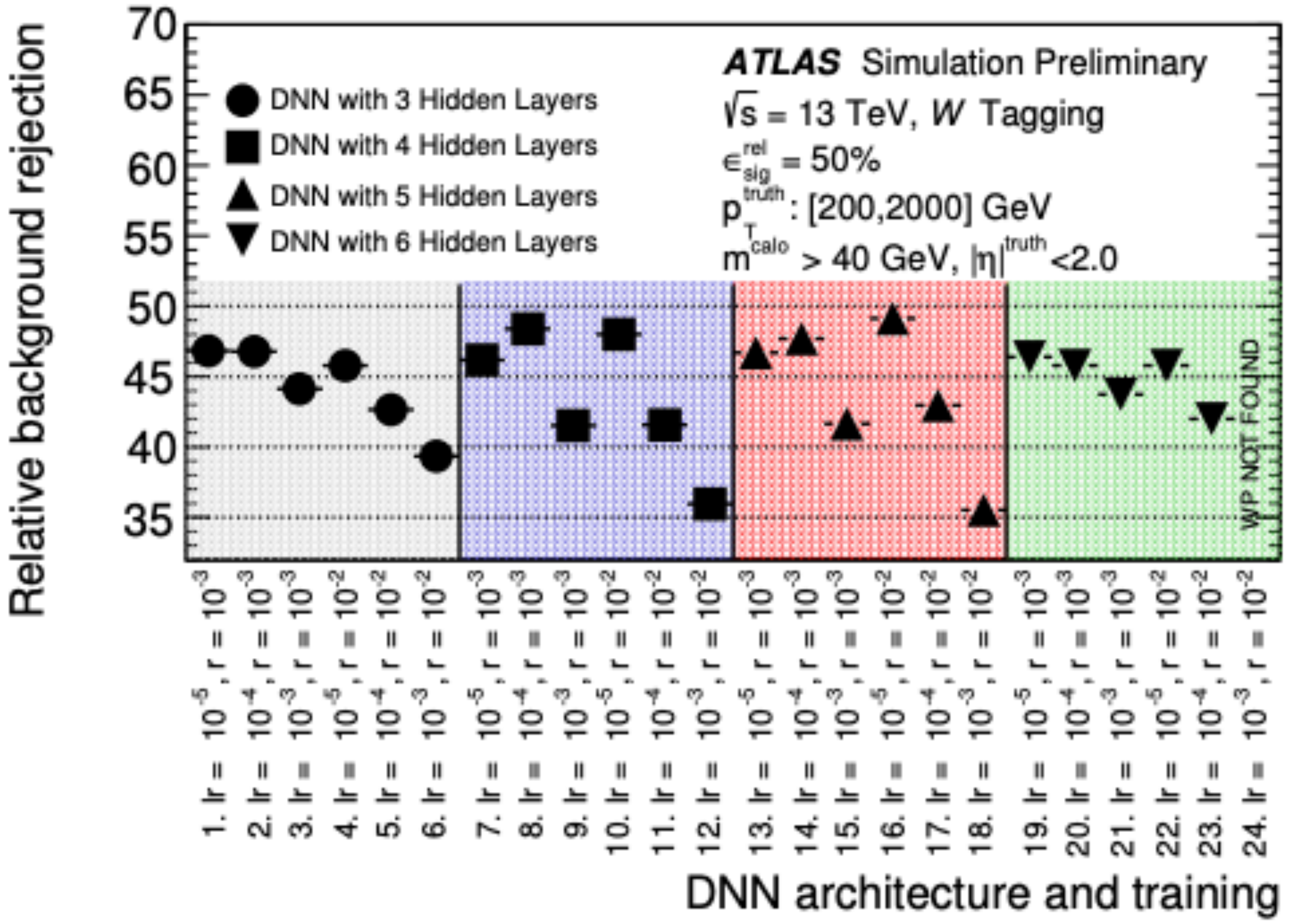
BDT parameters optimisation

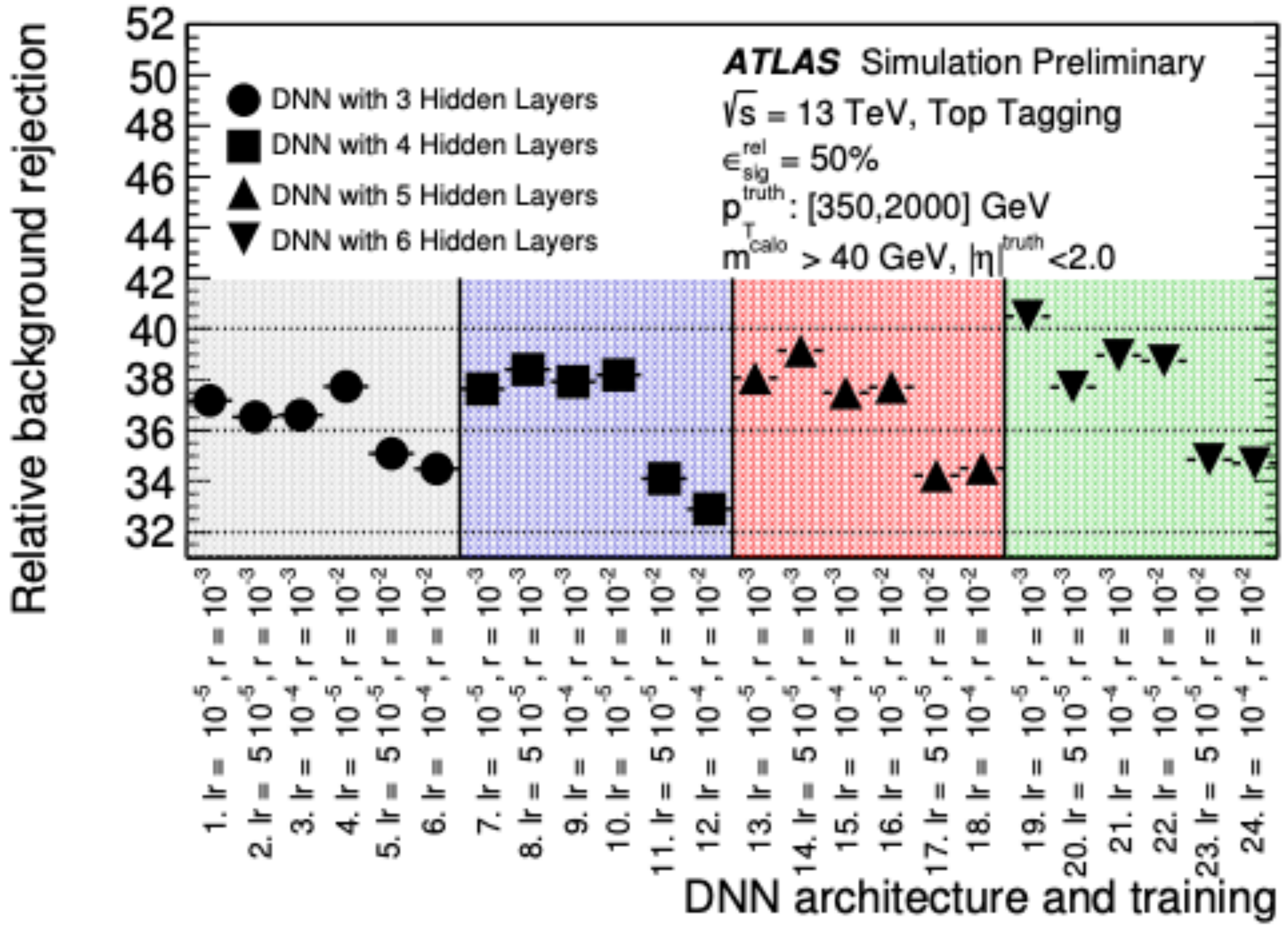




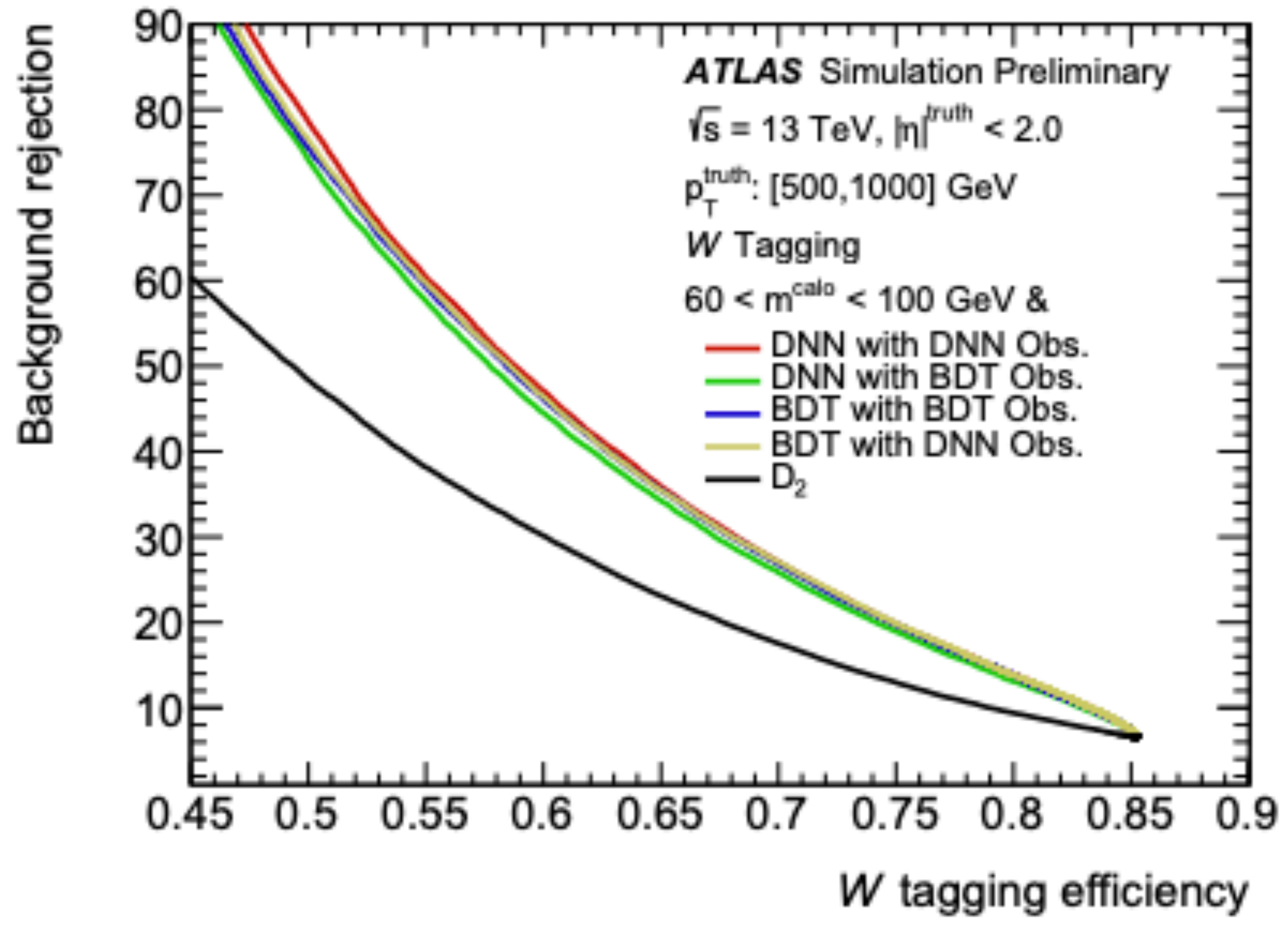
Large-R jets: tagging technique with ML algorithms

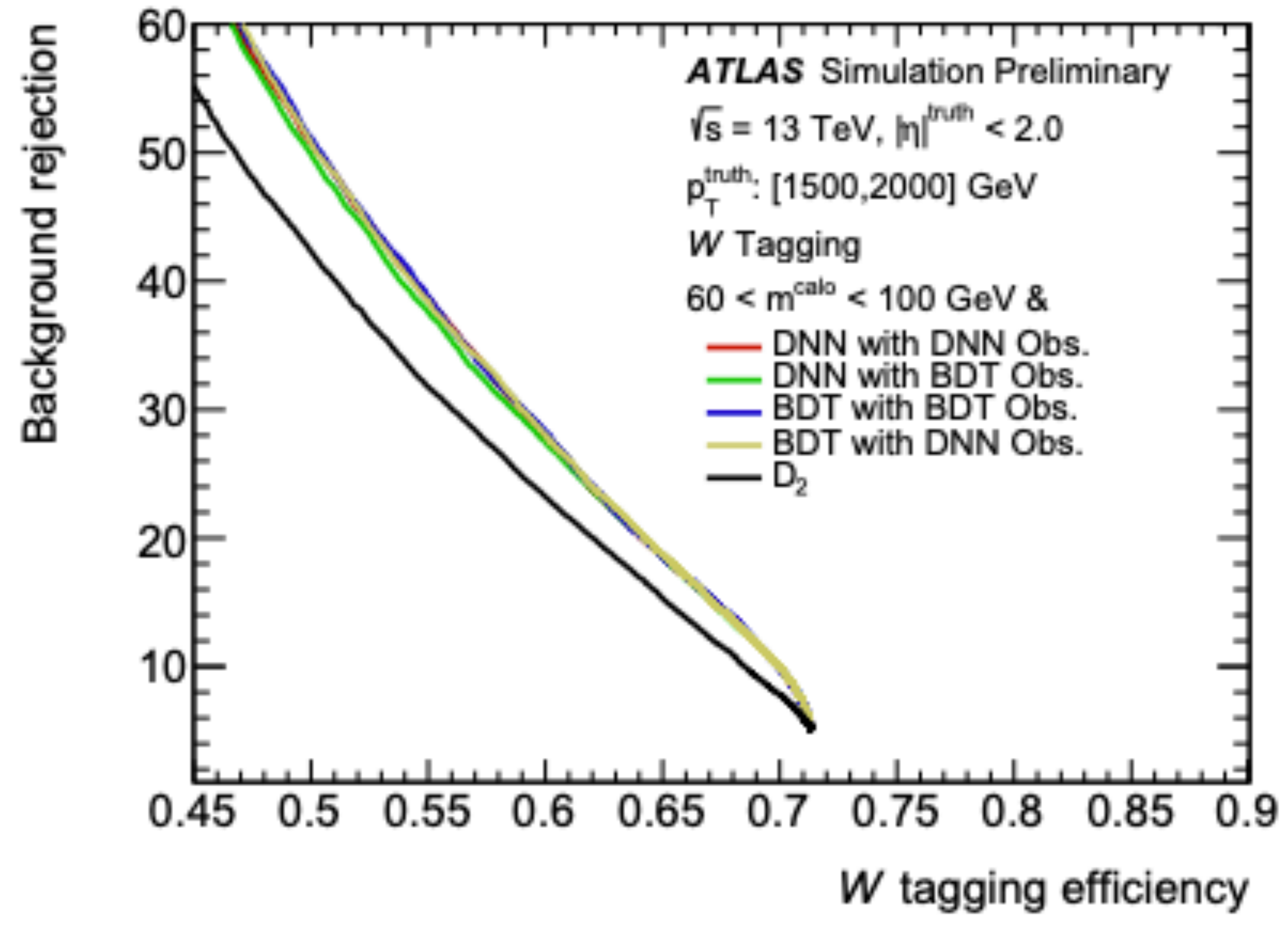
DNN hyperparameters optimisation





Large-R jets: tagging technique with ML algorithms

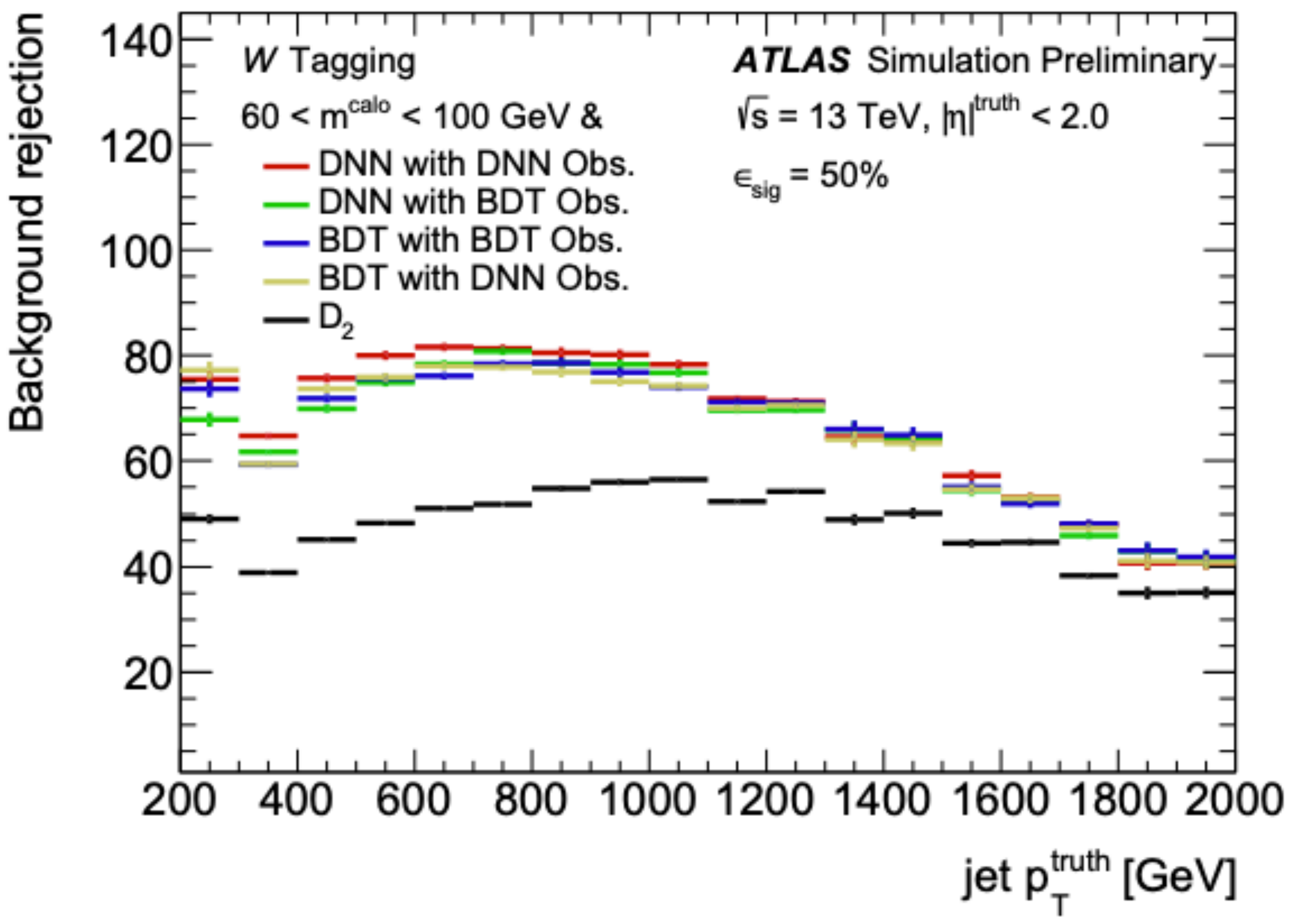




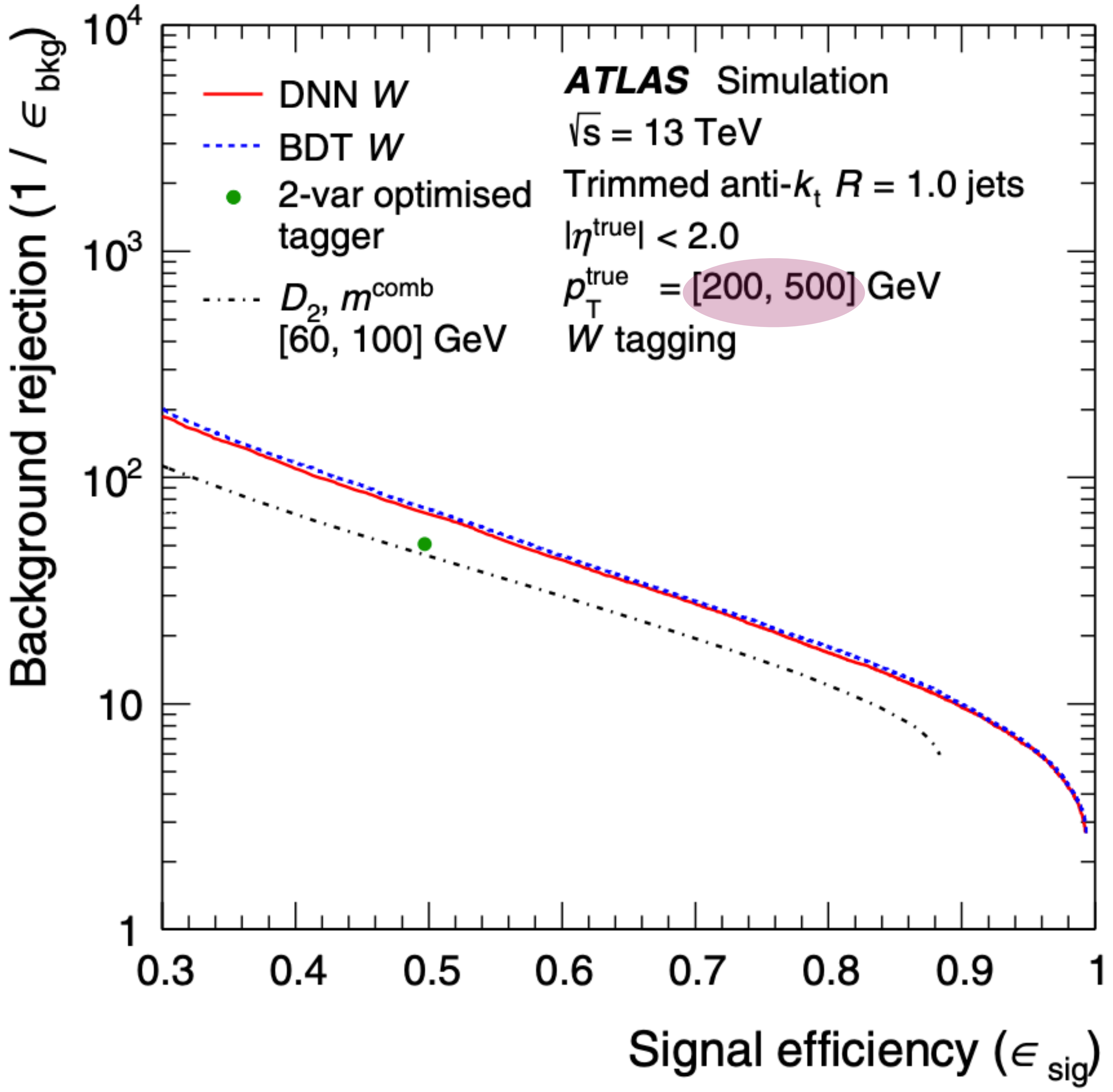
Performances of the taggers

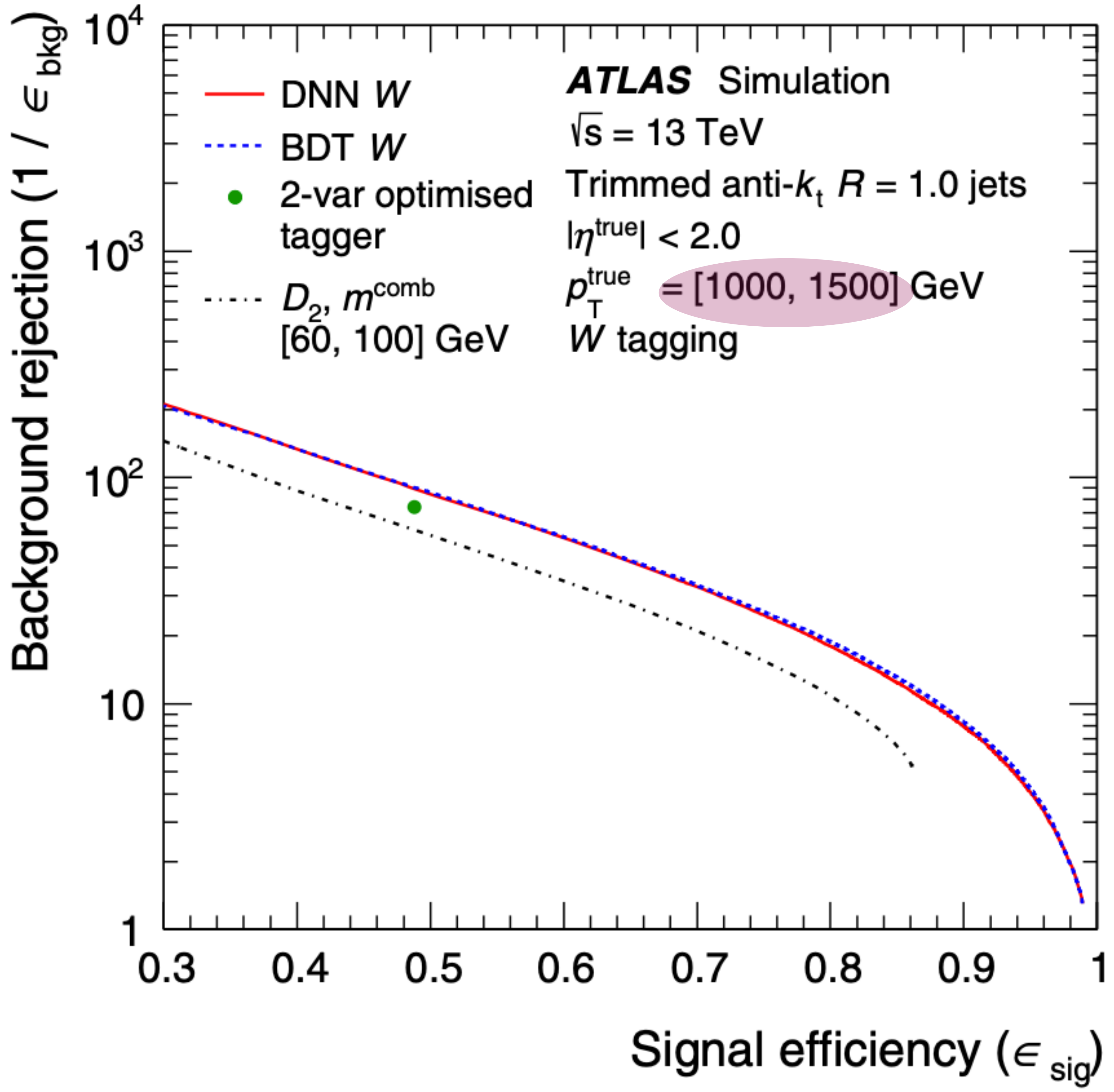
- Distributions showing comparison of the BDT and DNN taggers performance to the simple W-boson tagger based on a simple fixed cut on the mass and a selection on the D_2 observable in a low-p_T true(top left) and high-p_T true (top right) bin.

- The background rejection for a fixed 50% signal efficiency working point is also presented (bottom), where a selection defined by a requirement on the jet mass is followed by a D_2, BDT or DNN requirement optimised as a function of the jet p_T true.

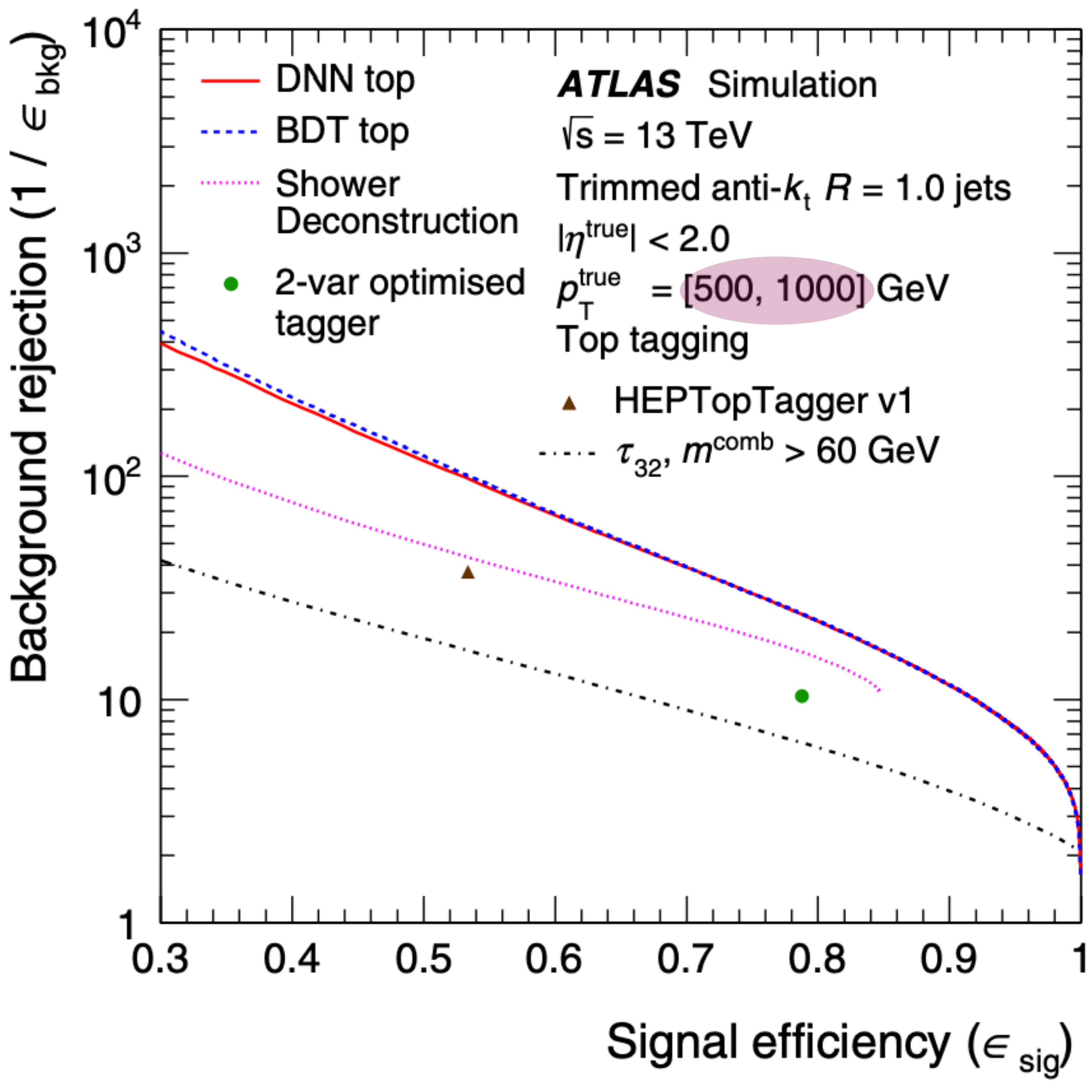


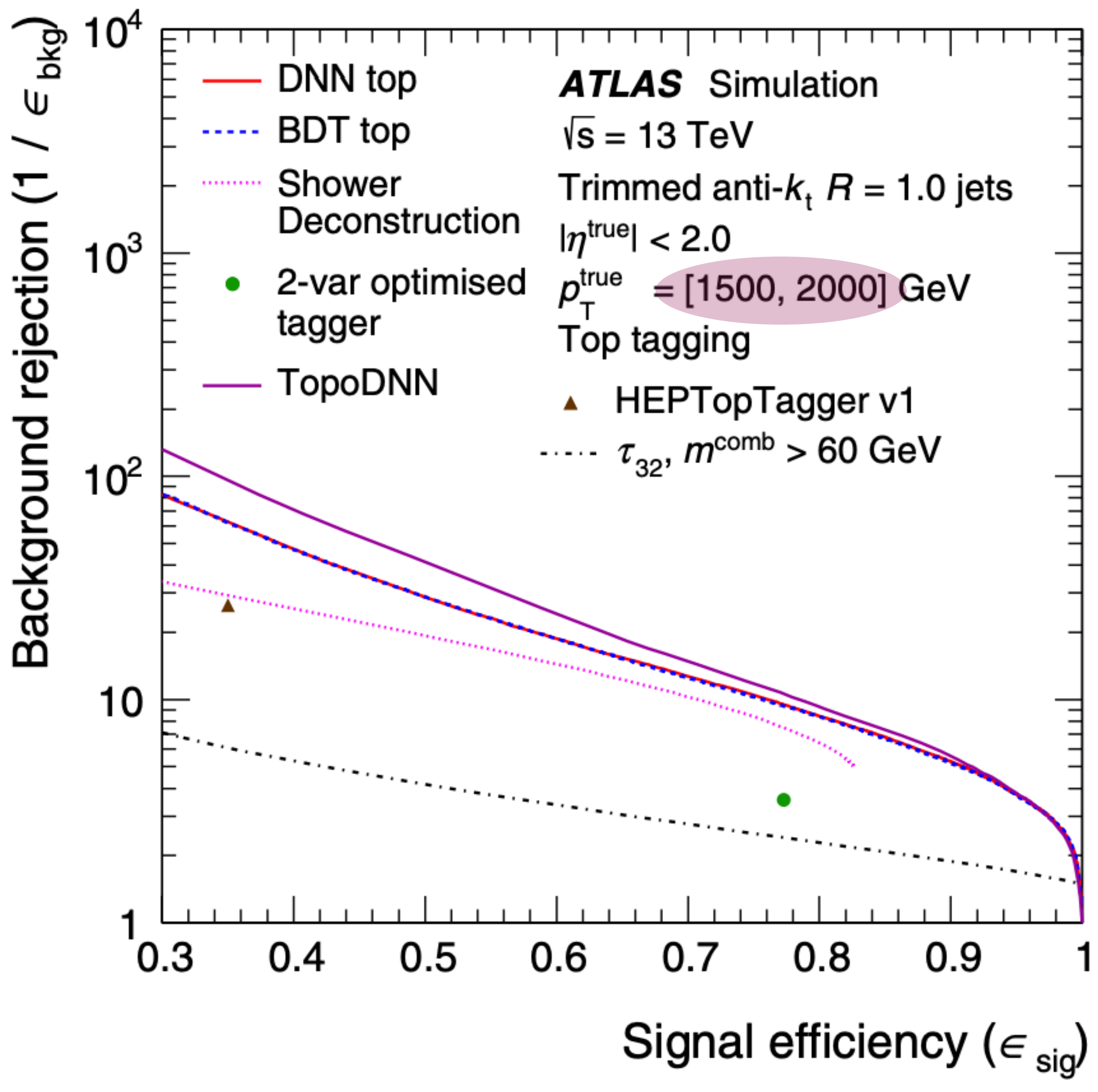
Large-R jets: tagging technique with ML algorithms





Large-R jets: tagging technique with ML algorithms





Large-R jets: tagging technique with ML algorithms