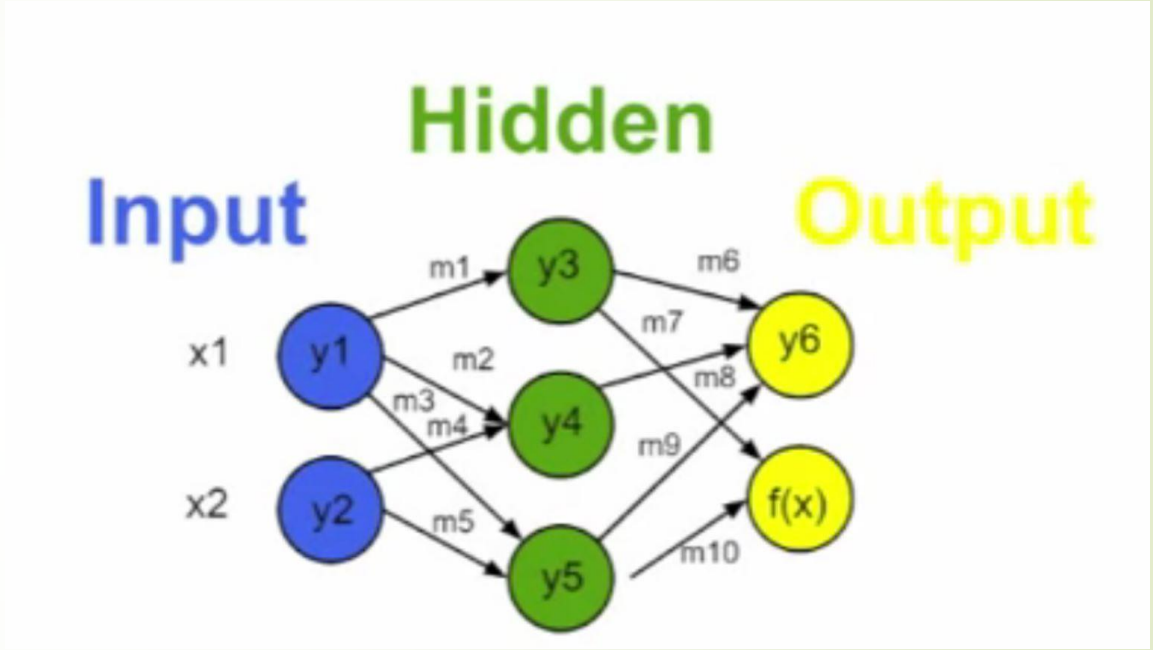
TMVA Methods

By Abdullah Alhag

Artificial Neural Networks
How does it work?

- There are neurons ordered in layers where the first hold the input values and last holds the output values with what called the hidden layers between.
- The connection between a neuron in one layer and the other is a weight which value the importance of an input from the previous layer.
- The first layer (input layer) have as many neuron as many input there is.
- The last layer (output layer) could have either one or many depending on the need (classification vs regression).
- The hidden layers which could be only one have as many neuron as the user choses.



Artificial Neural Networks
How does it work?

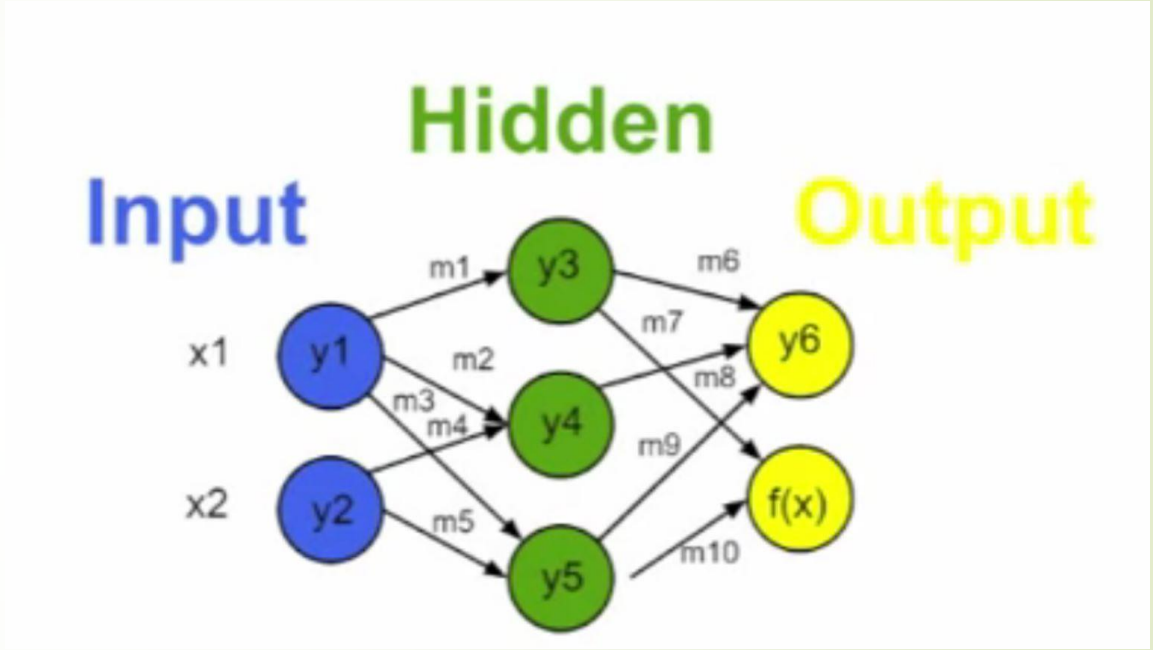
- Each neuron in the hidden layer has what is called neuron respond function which is composed of neuron activation function and synapse function .
- The available neuron activation function for use in TMVA are:
- x *Linear*
- $\frac{1}{1+e^{-kx}}$ *Sigmoid*
- $\frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}$ *Tanh*
- $e^{\frac{-x^2}{2}}$ *Radial*
- x is what is called synapse function

Artificial Neural Networks
How does it work?

- Thesynapse function could be either one of the following:
- $\sum_{i=0}^{n} x_i w_{ij}$ sum
- $\sum_{i=0}^{n} (x_i w_{ij})^2$ sum of squares
- $\sum_{i=0}^{n} \left| x_i w_{ij} \right|$ sum of absolutes
- Where n is the number of inputs, x_i is the input, and w_{ij} is the weight between input-layer neuron i and hidden-layer neuron j.
- The synapse function is what transfer the function from multi space (multiple inputs) in to one space (one output).

Artificial Neural Networks: Application

- How does that apply to the picture in the right?
- The neuron response function with choosing Tanh as the neuron activation function and sum as synapse function for y_4 is
- $y_4 = \tanh \sum_{i=0}^{n} x_i w_{ij}$
- $y_4 = \tanh \sum_{i=0}^{2} x_i m_{ij} = \tanh((y_1 m_2) + (y_2 m_4))$
- Where y_1 and y_2 is x_1 and x_2 respectively.
- For y_6, the only difference is that you are summing the result from all neuron in the last hidden layers (n_h).
- $y_6 = \sum_{j=1}^{n_h} \tanh \sum_{i=0}^{n} x_i w_{ij}$
- $= \sum_{j=1}^{3} \tanh \sum_{i=0}^{2} y_i m_{ij} = (\tanh((y_1 m_2) + (y_2 m_4)) m_8) + (y_3 m_6) + (y_5 m_9)$



Artificial Neural Networks
How does the weighting works?

- The weight are first chosen randomly than adjusted
- In TMVA, the weight is adjusted by the function $w^{(p+1)} = w^p - \mu \nabla_w E$, where μ is the learning rate selected by the user.
- Where E is the error function which is defined by $\sum_{a=1}^{N} \frac{1}{2}\left(y_{j,a} - y_a\right)$, $y_{j,a}$ is the function output, y_a is the desired output, and N is the number of training events.
- Provided that the neuron response function is differentiable with respect to the input weights, Using the same Tanh as the neuron activation function and sum as synapse function

Artificial Neural Networks
How does the weighting works?

- Understanding that the weighting is done on multiple events $\sum_{a=1}^{N}$ where each has it is own input parameter $x_{i,a}$.
- Taking $y_{j,a} = \sum_{j=1}^{n_h} \tanh \sum_{i=0}^{n} x_{i,a} w_{ij} \,.\, w_{j1}$ where as before n is the number of inputs, $x_{i,a}$ is the input, w_{ij} is the weight between input-layer neuron i and hidden-layer neuron j, n_h is the number of neurons in the hidden layer, and $w_{j1}$is the weight between the hidden-layer neuron j and the output neuron.
- For a neuron in the last layer, the weight between the output neuron and one of the neuron from previous layer is $\Delta\, w_{j1} = -\mu \sum_{a=1}^{N} \frac{\delta E_a}{\delta w_{j1}} = -\mu \sum_{a=1}^{N} (\, y_{j,a} - y_a)\, y_{j,a}$, j is just the neuron in the hidden layer which it is output being weighted.
- For a neuron in any of the hidden layers, the weight between one neuron in the hidden layer and the other connected to is $\Delta\, w_{ij} = -\mu \sum_{a=1}^{N} \frac{\delta E_a}{\delta w_{ij}} = -\mu \sum_{a=1}^{N} (\, y_{j,a} - y)\, y_{j,a}\, (1 - y_{j,a})\, w_{j1}\, x_{i,a}$ where $x_{i,a}$ is the input, and $w_{j1}$is the weight between the hidden-layer neuron j and the output neuron.
- Notice the $(1 - y_{j,a})$ coming from taking the derivative of tanh().

Artificial Neural Networks
Notes On using Artificial Neural Networks in TMVA

- Enabling the Bayesian extension will result in avoiding over training while allowing for the modeling of complex functions.
- It is better to increase the number of neurons in the hidden layer than to increase the number of hidden layers as the time spent on reweighting is decreased.

Boosted Decision Trees
How does it work?

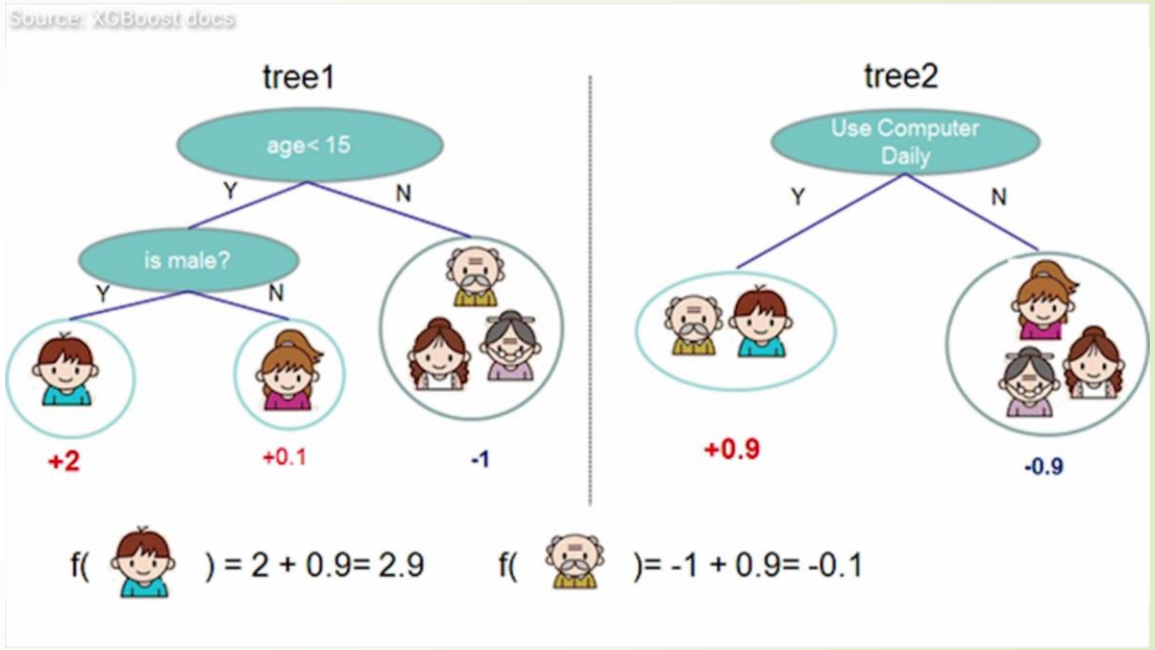
- At first a training events is giving to the tree.
- In the root a variable and a cut value which are at first randomly selected (true for all nodes) determine the splitting criteria of the events (Signal or Background for us).
- The variable and the cut value selected during the training procedure based on how much they optimisms the increase in separation index between the parents node and the sum of the indices of the two daughter nodes weighted by their relative fraction of events.
- The cut values are optimized by scanning over the variable range with a granularity that is set via the option nCuts.
- The training events are then split into two subsets where the same procedure is repeated in the two daughter nodes.

Boosted Decision Trees
How does it work?

- The process continues until the leaf node only consist of either signal or background if the UseYesLeaf option is set to true. Else, once the minimum number of events set by the option nEventsMin is reached, the decision is based on the purity of the leaf node.
- The purity (p) of the node is given by the ratio of signal to all events in the node.
- Each variable is assessed by separation criteria which can be chosen from the option SeperationType.
- All options are similar performance wise. All the following are separation criteria options available in MTVA:
- Gini Index $p(1-p)$, Cross entropy $-p\ \ln(p) - (1-p)\ln(1-p)$, Misclassification error $1\ - max\ (p, 1-p)$, Statistical significance $\frac{S}{\sqrt{S+B}}$, Average squared error $\frac{1}{N\ \sum^{N}(y-y')\text{^}2}$.

Boosted Decision Trees: Application

- How does that apply to the picture in the right?
- In the example on the right side, the goal is to figure out if someone plays video games or not.
- The variable selected to split the training events are (is male? , Age < 15 , and Uses computer daily)
- Each splitting criteria (variable) is assessed by a cut value that determined wither the variable optimisms the increase in separation index between the parents node and the sum of the indices of the two daughter nodes.
- The cut value for (is male?) is closer to 1 than it is for (Uses computer daily) because it is statistically being a boy means greater chance of playing game as computers are not the only mean of gaming, consoles are another one.
- The values here do not really mean much, think of it as not just yes or no but more of a number associated with yes and no. The number varies depending on the cut value.



Boosted Decision Trees
Advantages over non-boosted decision tree

- Boosting increases the statistical stability of the classifier and is able to drastically improve the separation performance compared to a single decision tree when input variables exhibit similar separation power.
- Limiting the tree depth during the tree building process (training), the tendency of over training for simple decision trees which are typically grown to a large depth and then pruned, is almost completely eliminated.

Boosted Decision Trees
Notes on using Boosted Decision Trees in TMVA

- A truly optimal cut is determined by setting nCuts=-1. This invokes an algorithm that tests all possible cuts on the training sample and finds the best one.
- Decision trees are sometimes referred to as the best “out of the box” classifiers. This is because little tuning is required in order to obtain reasonably good results due to the simplicity of the method where each training step (node splitting) involves only a one-dimensional cut optimization.

Reverences

- TMVA user guide
- Introduction to Artificial Neural Networks