ISAPP Summer Institute 2009

Neural networks in data analysis

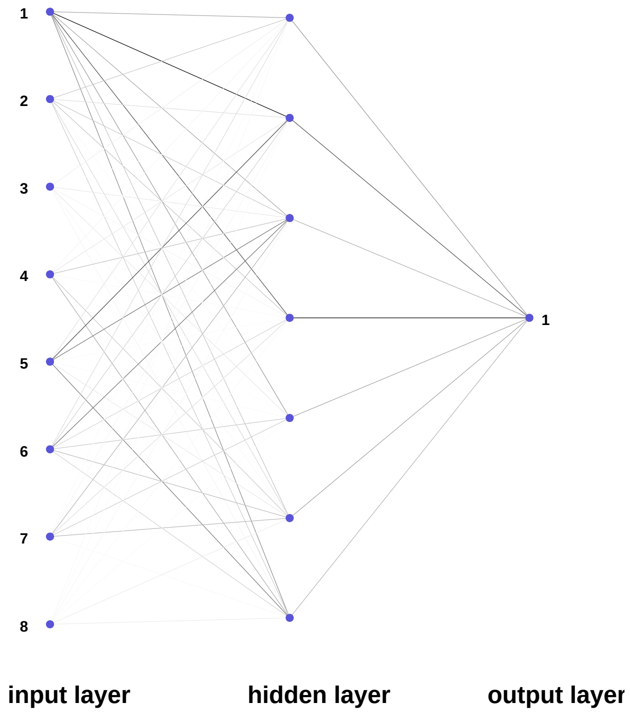
Michal Kreps

KIT
Karlsruhe Institute of Technology

Bundesministerium
für Bildung
und Forschung

Outline

- What are the neural networks
- Basic principles of NN
- For what are they useful
- Training of NN
- Available packages
- Comparison to other methods
- Some examples of NN usage



I'm not expert on neural networks, only advanced user who understands principles.

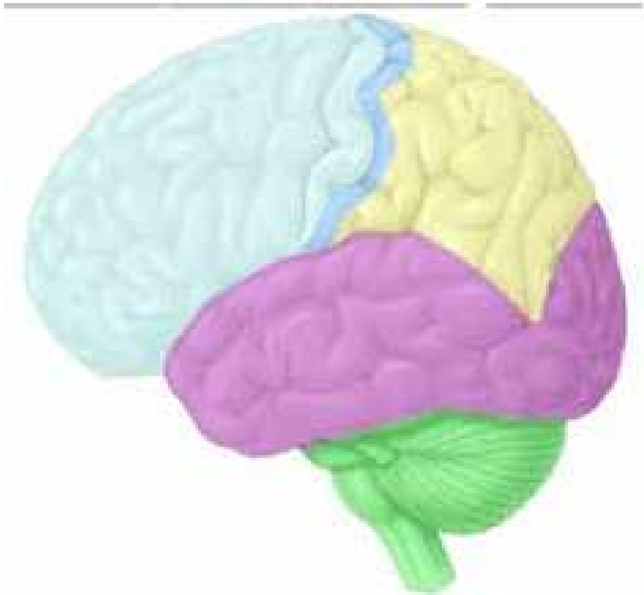
Goal: Help you to understand principles so that:

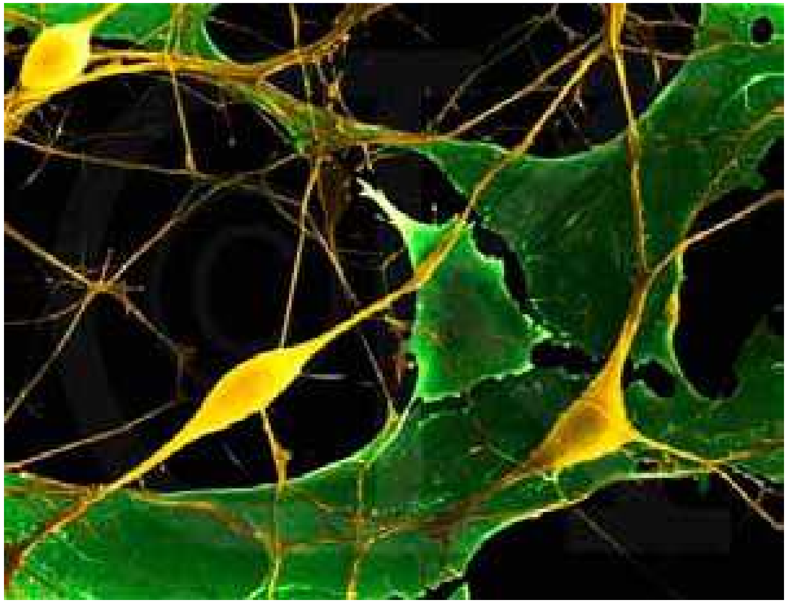
→ neural networks don't look as black box to you

→ you have starting point to use neural networks by yourself

Tasks to solve

- Different tasks in nature
- Some easy to solve algorithmically, some difficult or impossible
- Reading hand written text
  - Lot of personal specifics
  - Recognition often needs to work only with partial information
  - Practically impossible in algorithmic way
  - But our brains can solve this tasks extremely fast
- Use brain as inspiration





Tasks to solve

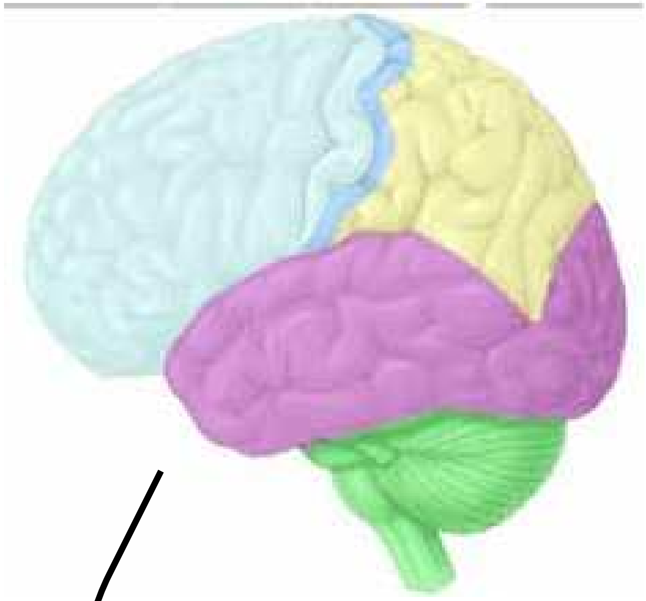
- Why (astro-)particle physicist would care about recognition of hand written text?
- Because it is classification task on not very regular data
  Do we see now letter A or H?

→ Classification tasks are very often in our analysis
Is this event signal or background?

→ Neural networks have some advantages in this task:

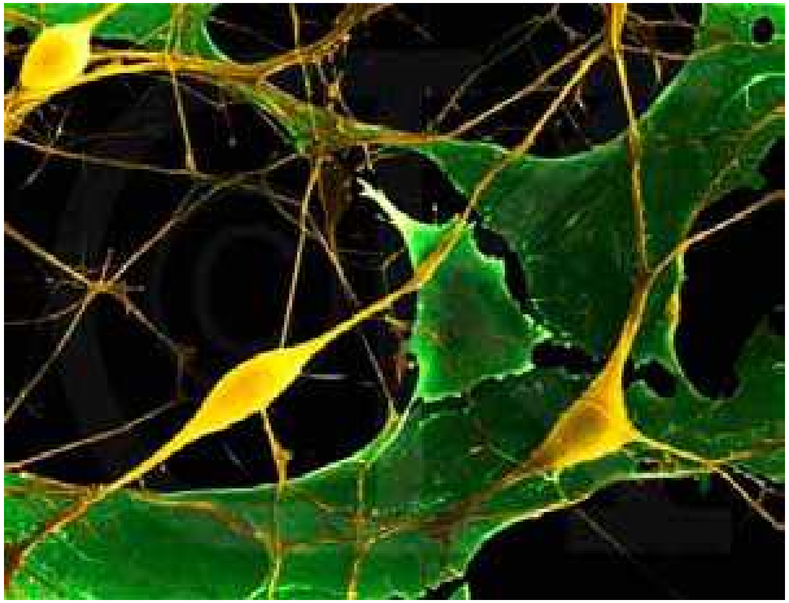
  - Can learn from known examples (data or simulation)
  - Can evolve to complex non-linear models
  - Easily can deal with complicated correlations amongs inputs
  - Going to more inputs does not cost additional examples

Down to principles



- Detailed look shows huge number of neurons
- Neurons are interconnected, each having many inputs from other neurons

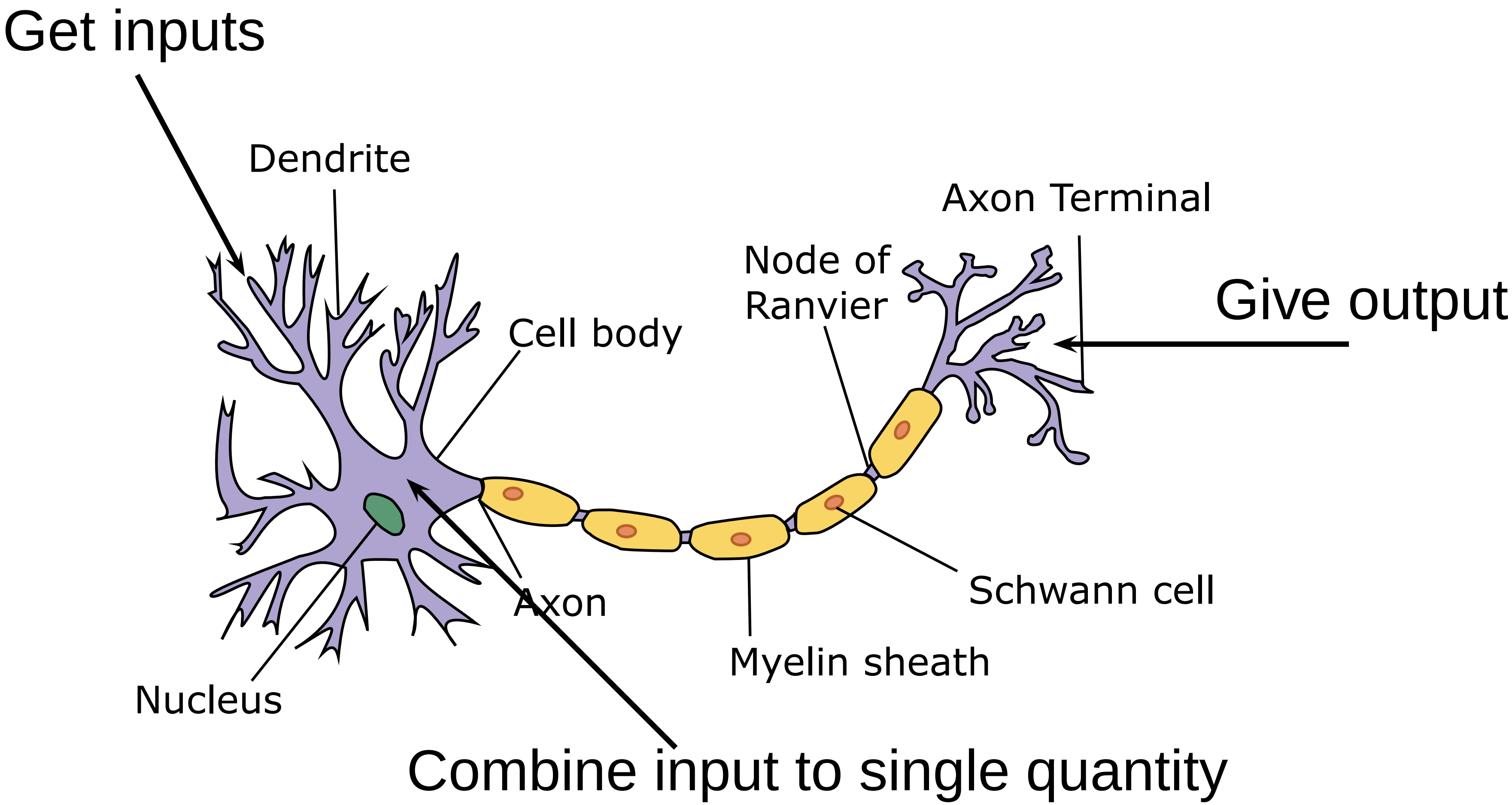
Zoom in



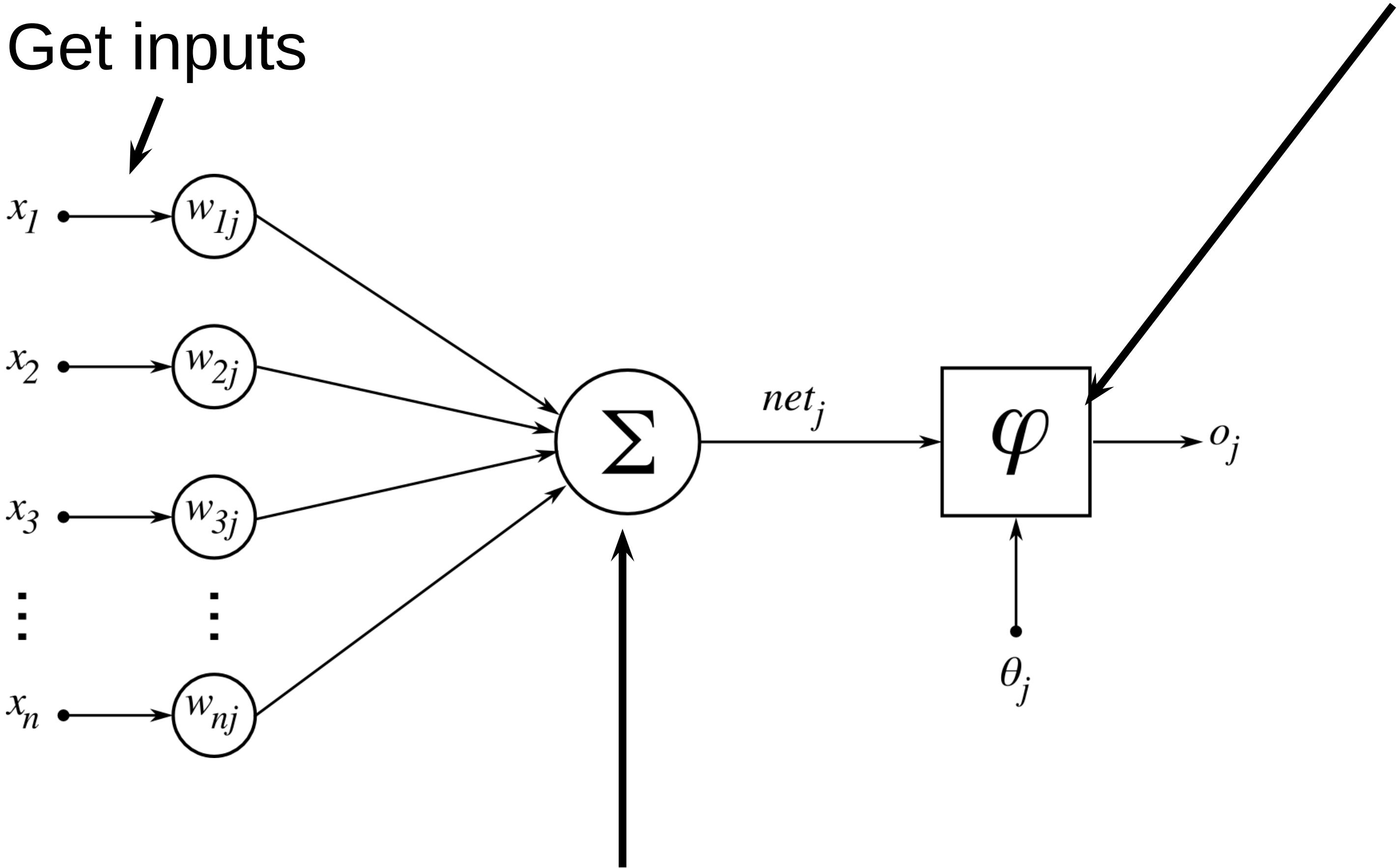
To build artificial neural network:

- Build model of neuron
- Connect many neurons together

Model of neuron



Model of neuron



Get inputs

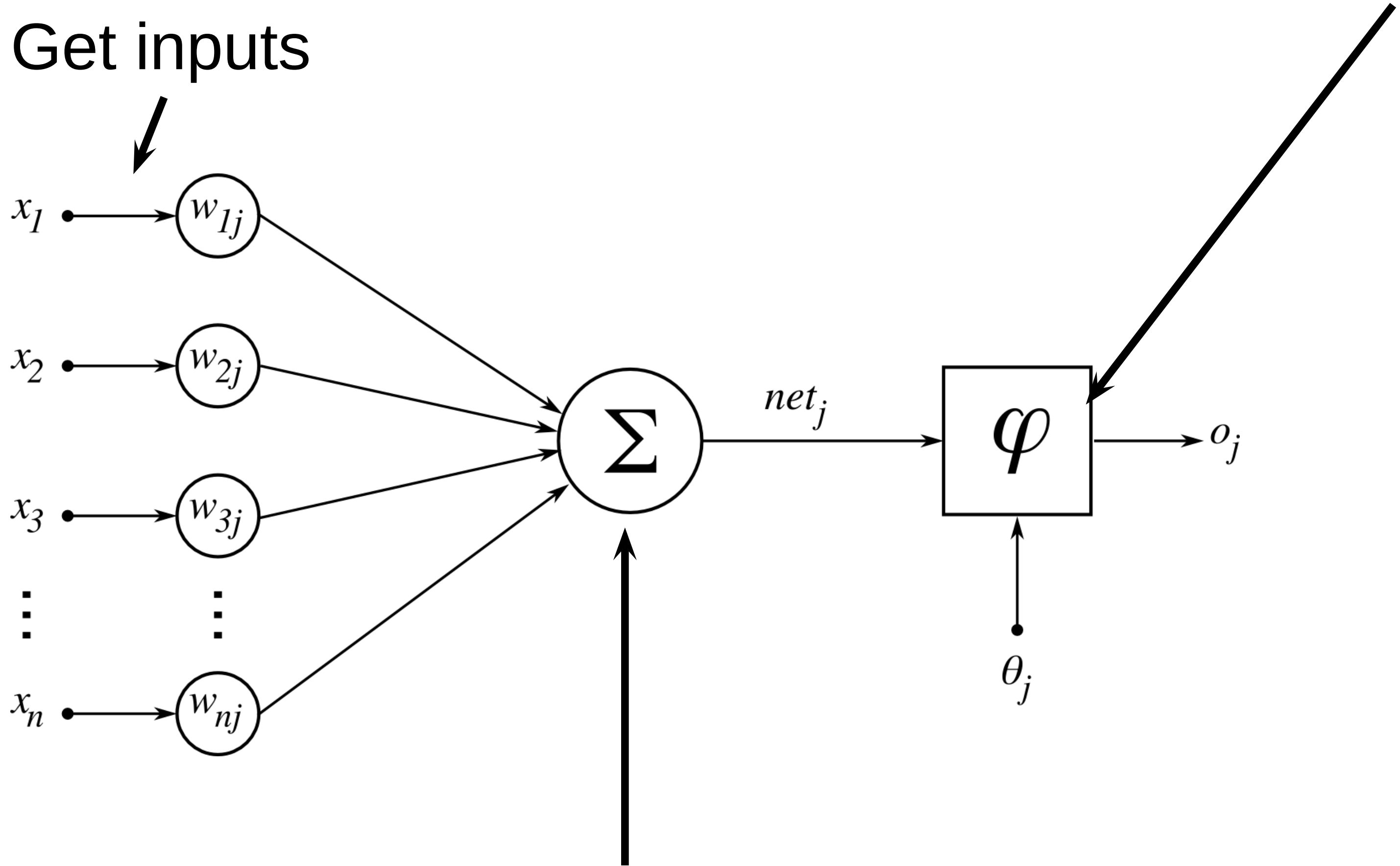
Give output

Combine input to single quantity

Model of neuron

Give output

Get inputs



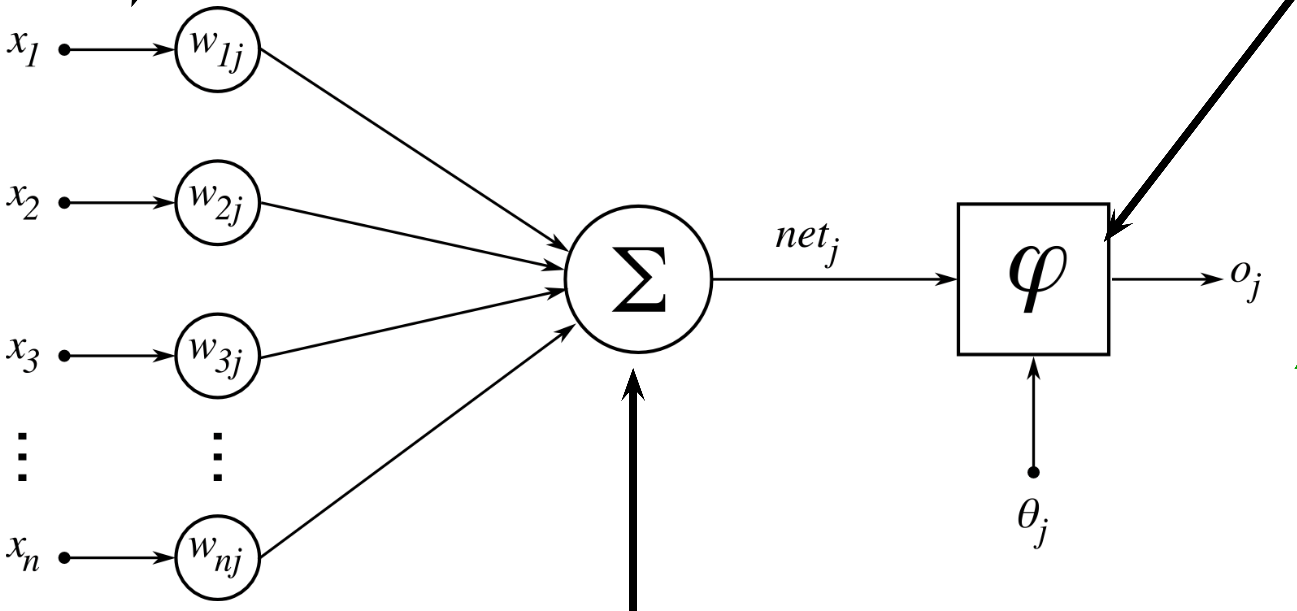
Combine input to single quantity

$S = \sum_i w_i x_i$

Model of neuron

Get inputs

Give output



Pass S through activation function

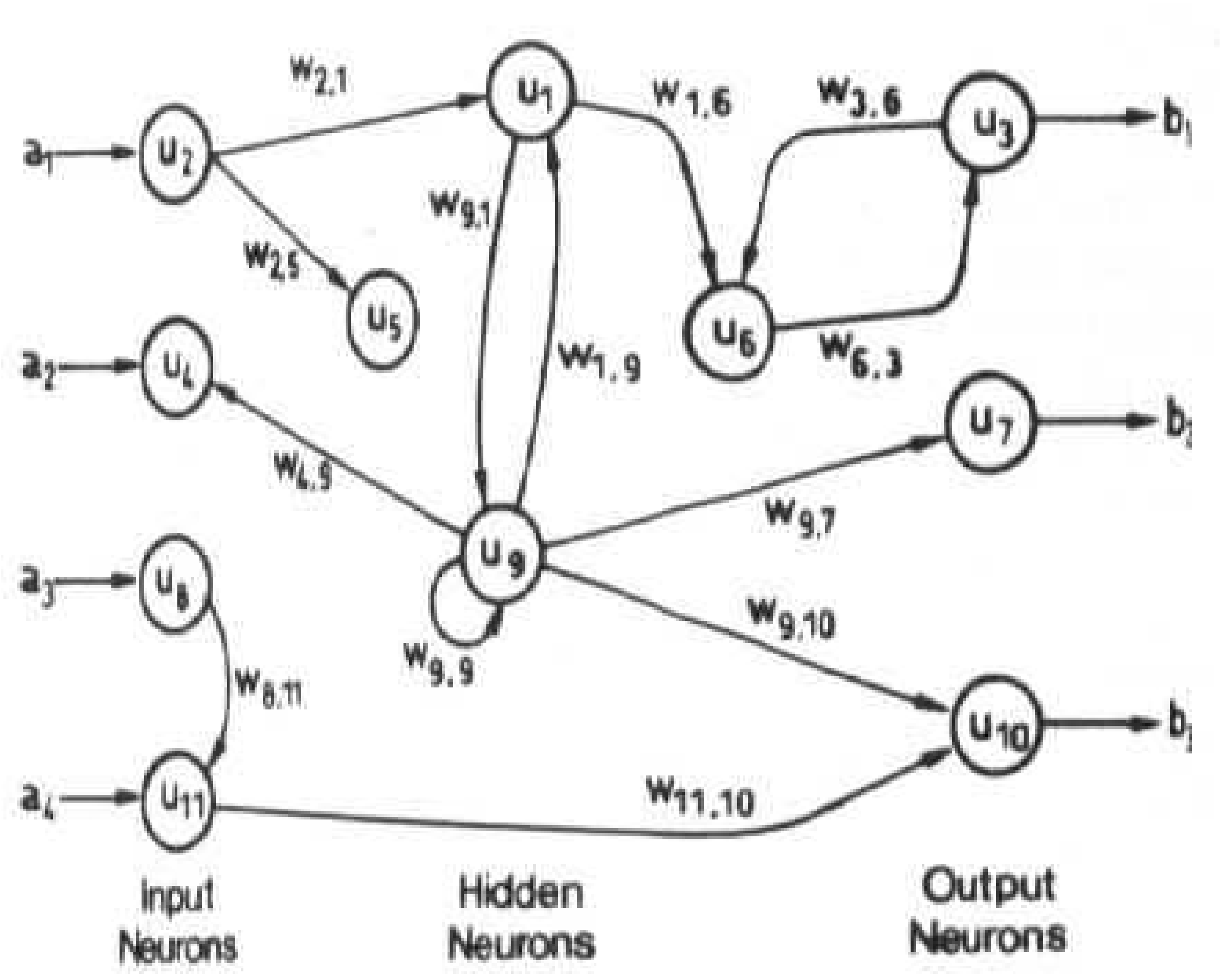
Most common:

$A(x) = \frac{2}{1+e^{-x}} - 1$

Combine input to single quantity

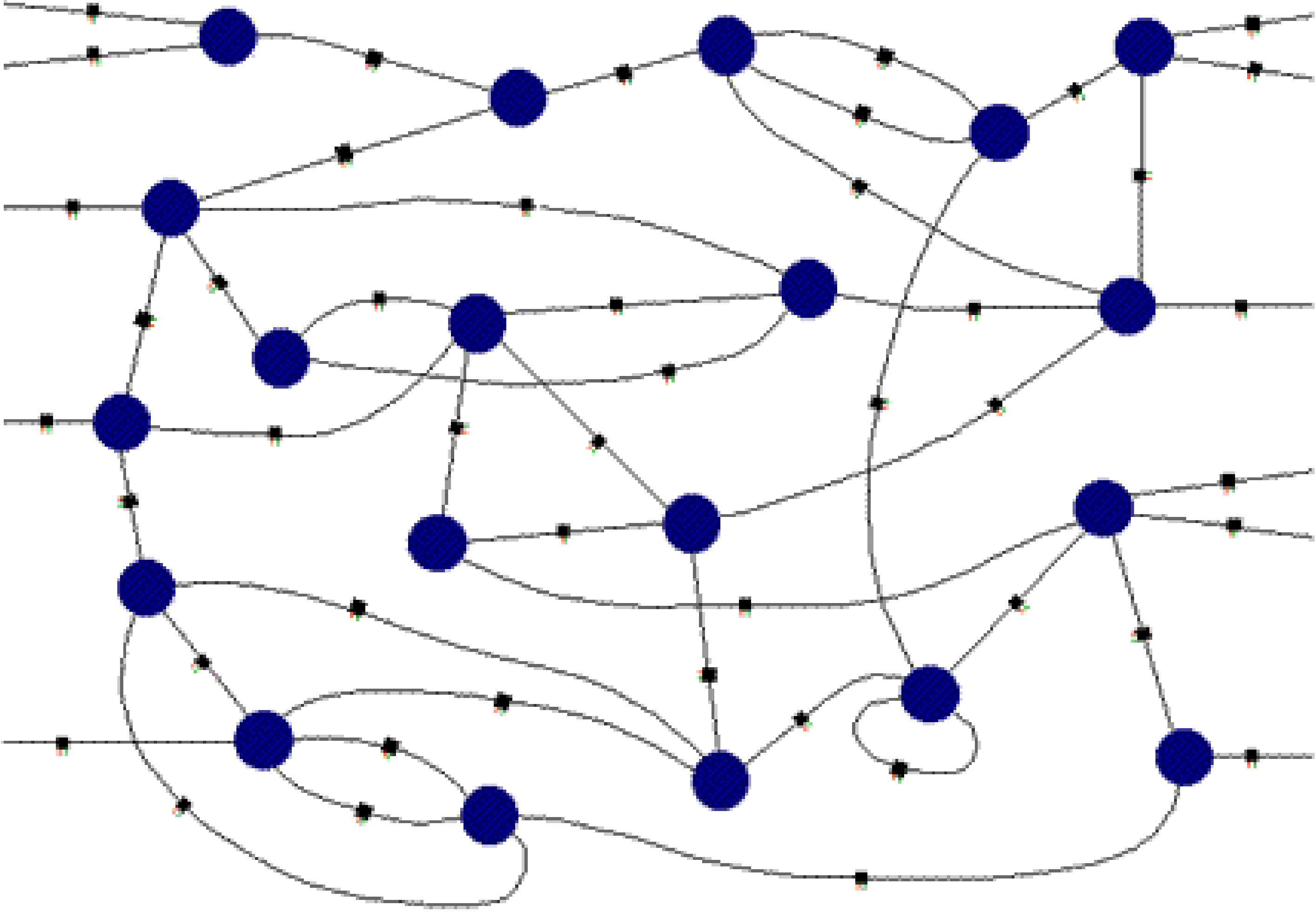
$S = \sum_i w_i x_i$

Building neural network



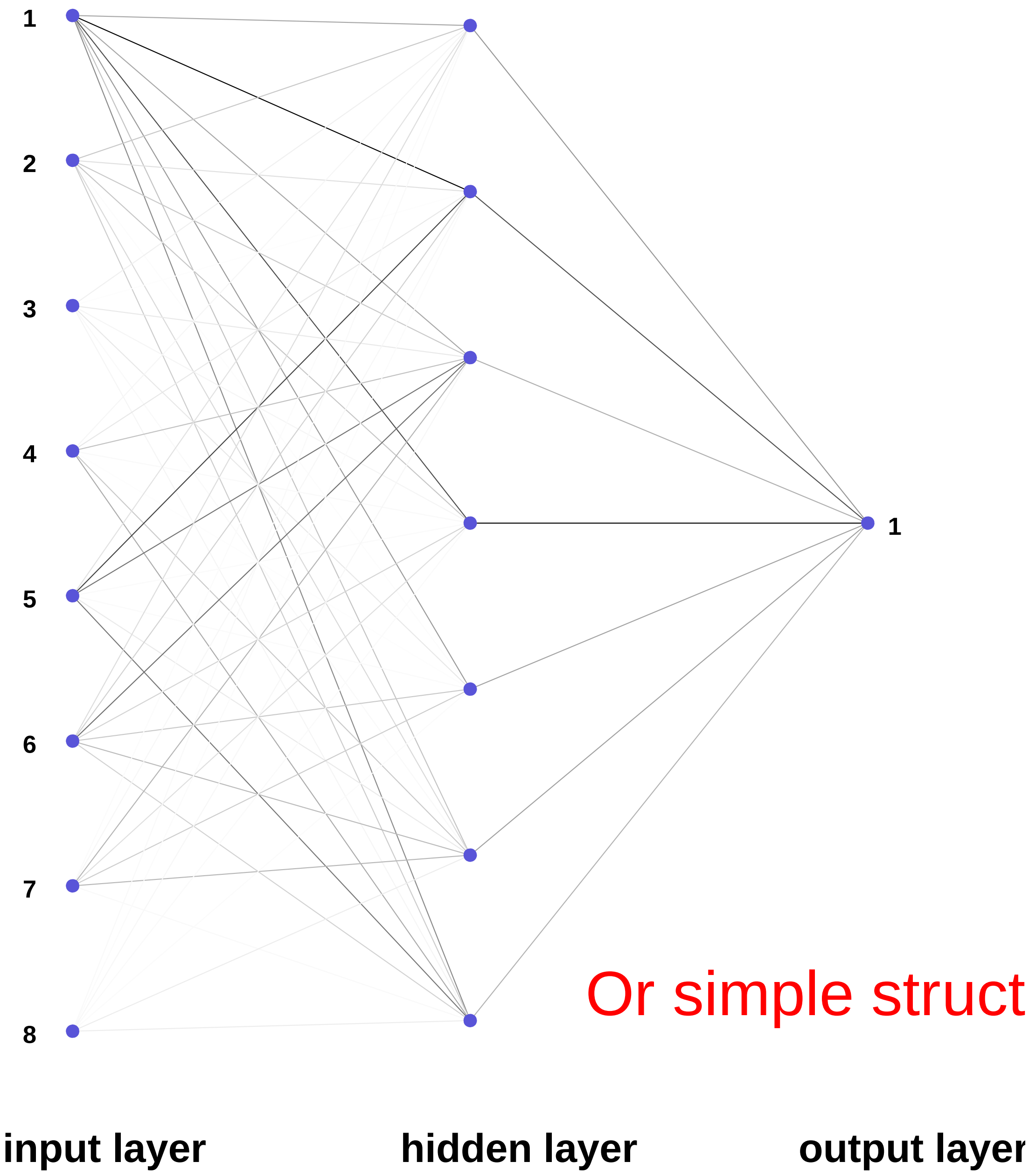
Complicated structure

Building neural network



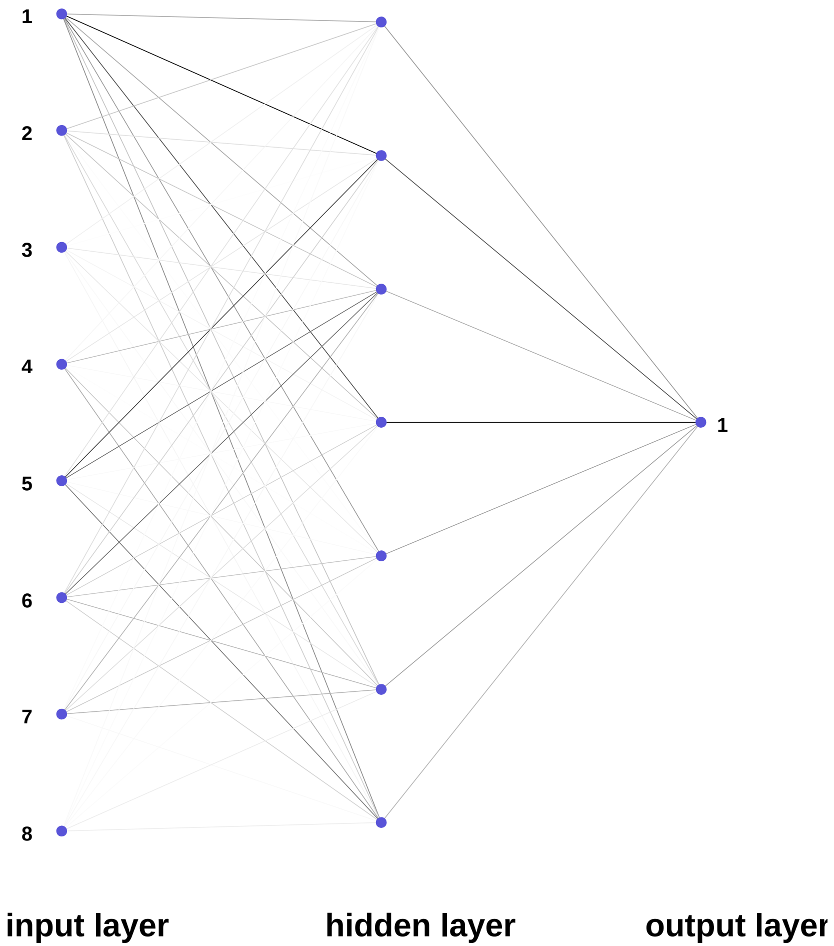
Even more complicated structure

Building neural network



Or simple structures

Output of Feed-forward NN



- Output of node *k* in second layer:
  $o_k = A\left(\sum_i w_{ik}^{1\to 2} x_i\right)$
- Output of node *j* in output layer:
  $o_j = A\left(\sum_k w_{kj}^{2\to 3} x_k\right)$
  $o_j = A\left[\sum_k w_{kj}^{2\to 3} A\left(\sum_i w_{ik}^{1\to 2} x_i\right)\right]$
- More layers usually not needed
- If more layers involved, output follows same logic
- In following, stick to only three layers as here
- Knowledge is stored in weights

Tasks we can solve

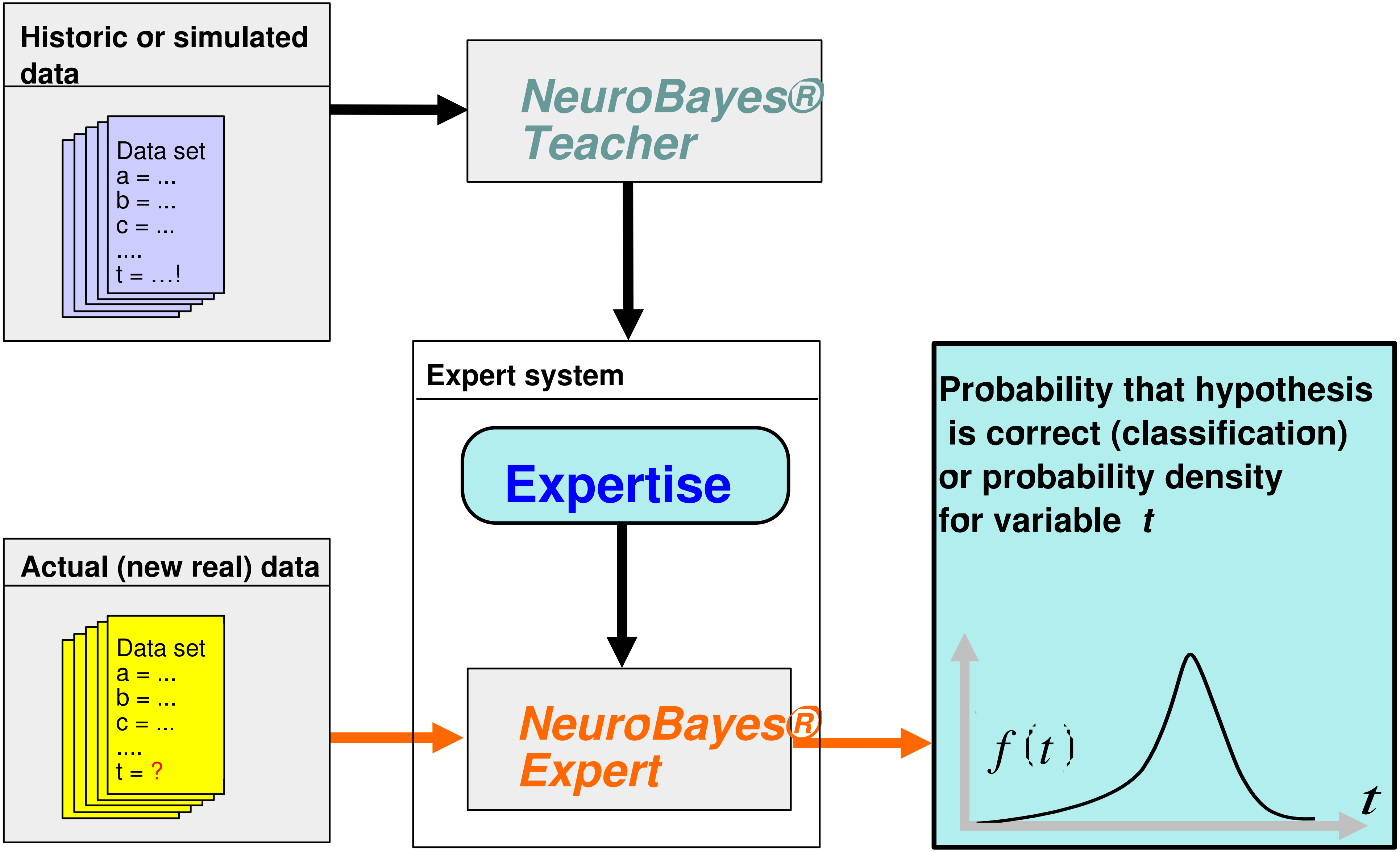
- Classification
  - Binary decision whether event belongs to one of the two classes
  - Is this particle kaon or not?
  - Is this event candidate for Higgs?
  - Will Germany become soccer world champion in 2010?

  → Output layer contains single node

- Conditional probability density
  - Get probability density for each event
  - What is decay time of a decay
  - What is energy of particle given the observed shower
  - What is probability that customer will cause car accident

  → Output layer contains many nodes

NN work flow



Historic or simulated data

Data set
a = ...
b = ...
c = ...
....
t = ...!

NeuroBayes® Teacher

Expert system

Expertise

Actual (new real) data

Data set
a = ...
b = ...
c = ...
....
t = ?

NeuroBayes® Expert

Probability that hypothesis is correct (classification) or probability density for variable *t*

Neural network training

- After assembling neural networks, weights $w_{ik}^{1\to 2}$ and $w_{kj}^{2\to 3}$ unknown
  - Initialised to default value
  - Random value
- Before using neural network for planned task, need to find best set of weights
- Need some measure, which tells which set of weights performs best $\Rightarrow$ loss function
- Training is process, where based on known examples of events we search for set of weights which minimise loss function
  - Most important part in using neural networks
  - Bad training can result in bad or even wrong solutions
  - Many traps in the process, so need to be careful

Loss function

- Practically any function which allows you to decide, which neural network is better

→ In particle physics it can be one which results in best measurement

- In practical applications, two commonly used functions (T_{ji} is true value with -1 for one class and 1 for other, o_{ji} is actual neural network output)
  - Quadratic loss function
    $\chi^2 = \sum_j w_j \chi_j^2 = \sum_j \frac{1}{2} \sum_i (T_{ji} - o_{ji})^2$
    → $\chi^2 = 0$ for correct decisions and $\chi^2 = 2$ for wrong
  - Entropy loss function
    $E_D = \sum_j w_j E_D^j = \sum_j \sum_i -\log(\frac{1}{2}(1 + T_{ji} o_{ji}))$
    → $E_D = 0$ for correct decisions and $E_D = \infty$ for wrong

Conditional probability density

- Classification is easy case, we need single output node and if output is above chosen value, it is class 1 (signal) otherwise class 2 (background)
- Now question is how to obtain probability density?
- Use many nodes in output layer, each answering question, whether probability is larger than some value
- As example, lets have $n = 10$ nodes and true value of 0.63
  - Vector of true values for output nodes: $(1, 1, 1, 1, 1, 1, -1, -1, -1, -1)$
  - Node j answers question whether probability is larger than $j/10$
- The output vector that represents integrated probability density
- After differentiation we obtain desired probability density

Interpretation of NN output

- In classification tasks in general higher NN output means more signal like event

→ NN output rescaled to interval [0; 1] can be interpreted as probability

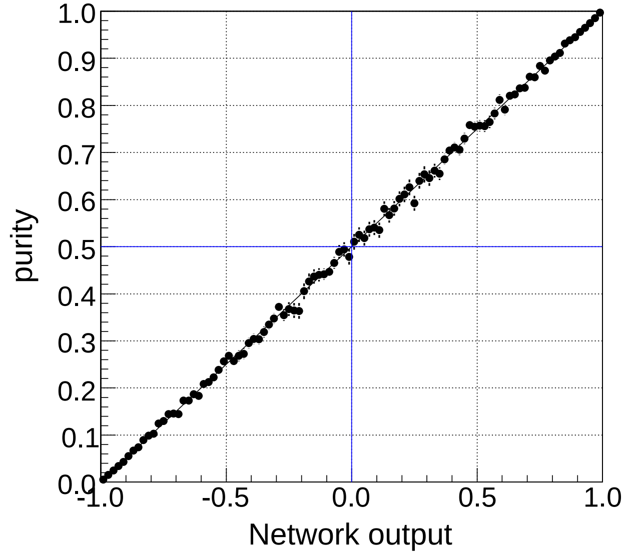
  - Quadratic/entropy loss function is used
  - NN is well trained ⇔ loss function minimised

  → Purity $P(o) = N_{\text{selected signal}}(o)/N_{\text{selected}}(o)$ should be linear function of NN output

- Looking to expression for output

$$o_j = A\left[\sum_k w_{kj}^{2\to3} A\left(\sum_i w_{ik}^{1\to2} x_i\right)\right]$$

→ NN is function, which maps n-dimensional space to 1-dimensional space



How to check quality of training

Define quantities:

- Signal efficiency:

  $$\epsilon_S = \frac{\text{N(selected signal)}}{\text{N(all signal)}}$$

- Efficiency:

  $$\epsilon = \frac{\text{N(selected)}}{\text{N(all)}}$$

- Purity:

  $$P = \frac{\text{N(selected signal)}}{\text{N(selected)}}$$

How to check quality of training

Define quantities:

- Signal efficiency:

$$\epsilon_S = \frac{\text{N(selected signal)}}{\text{N(all signal)}}$$

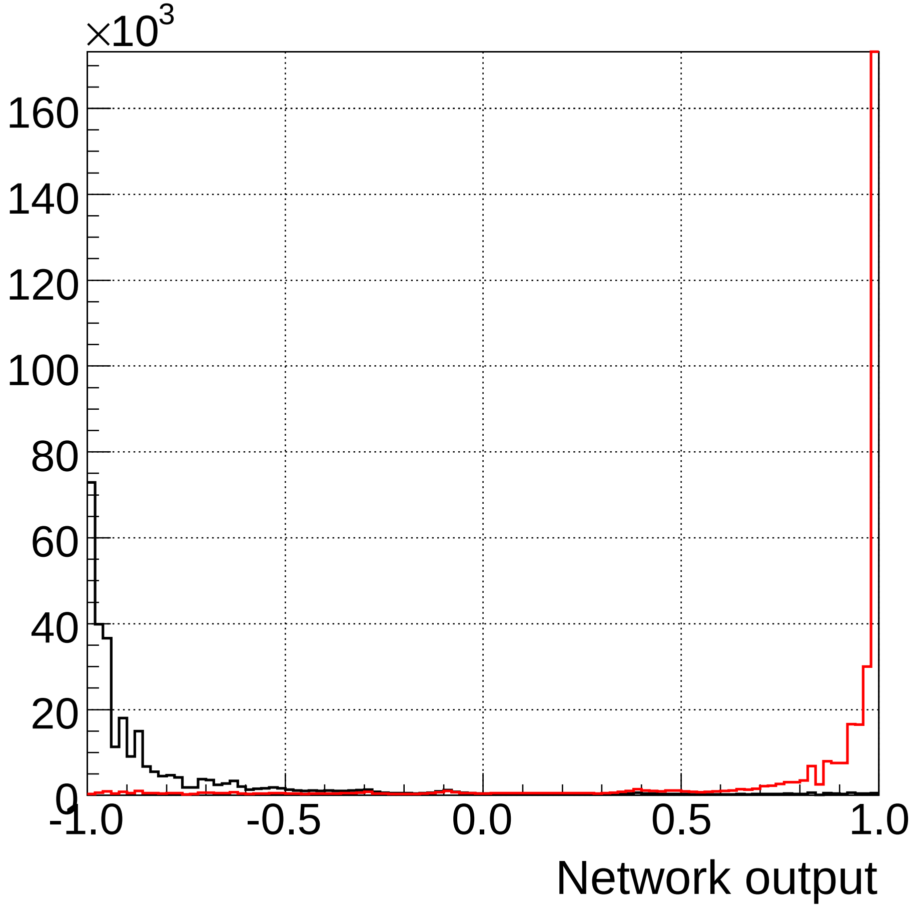
- Efficiency:

$$\epsilon = \frac{\text{N(selected)}}{\text{N(all)}}$$

- Purity:

$$P = \frac{\text{N(selected signal)}}{\text{N(selected)}}$$

Distribution of NN output for two classes



How good is separation?

How to check quality of training

Define quantities:

- Signal efficiency:

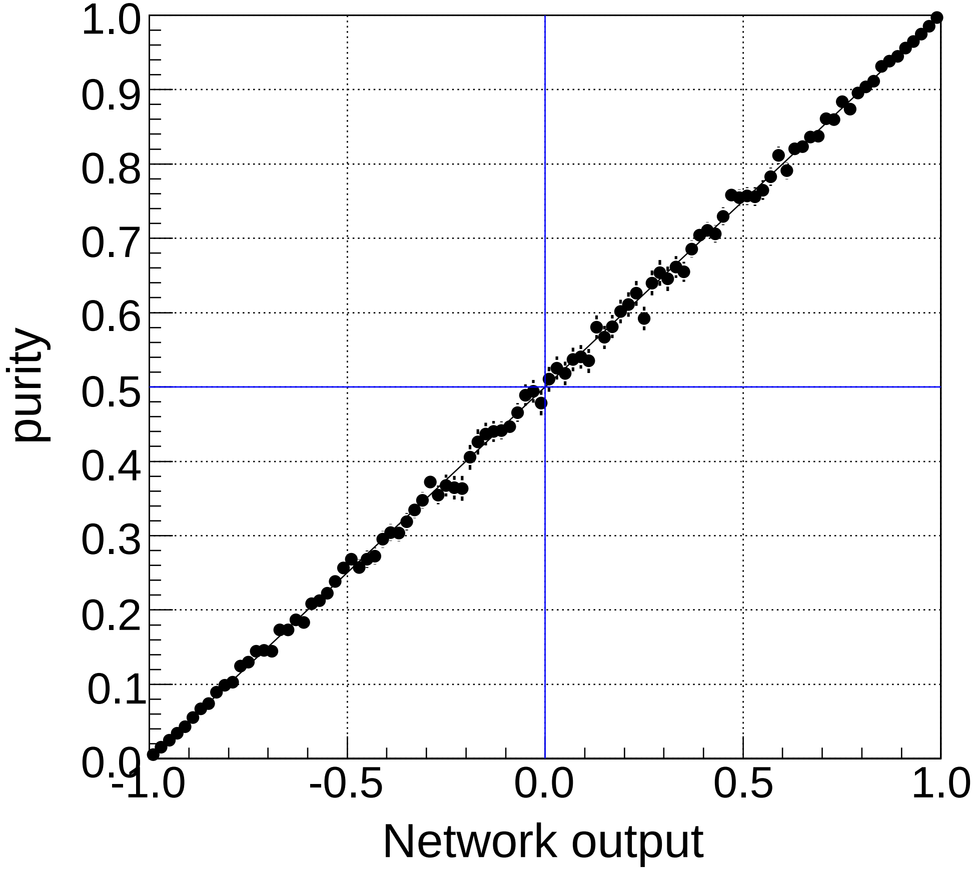
$$\epsilon_S = \frac{\text{N(selected signal)}}{\text{N(all signal)}}$$

- Efficiency:

$$\epsilon = \frac{\text{N(selected)}}{\text{N(all)}}$$

- Purity:

$$P = \frac{\text{N(selected signal)}}{\text{N(selected)}}$$



Are we at minimum of the loss function?

How to check quality of training

Define quantities:

- Signal efficiency:

$$\epsilon_S = \frac{\text{N(selected signal)}}{\text{N(all signal)}}$$

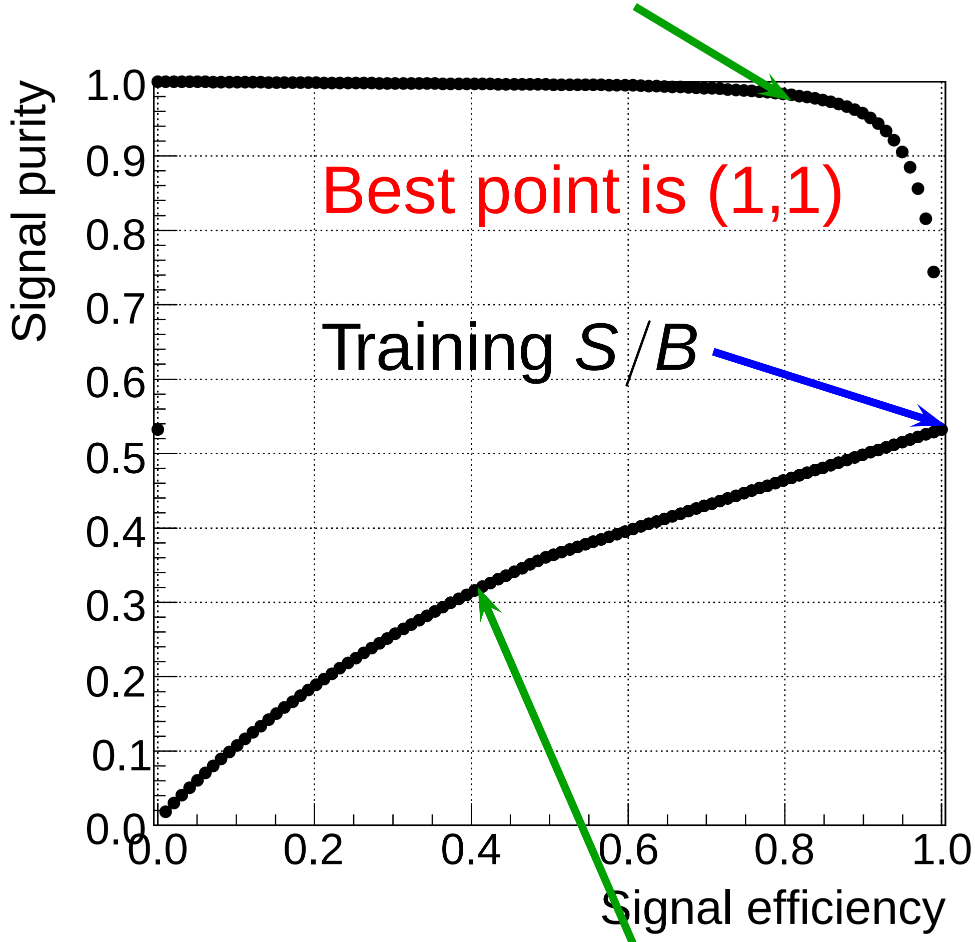
- Efficiency:

$$\epsilon = \frac{\text{N(selected)}}{\text{N(all)}}$$

- Purity:

$$P = \frac{\text{N(selected signal)}}{\text{N(selected)}}$$

Each point has requirement on *out* $>$ *cut*



Best point is (1,1)

Training S/B

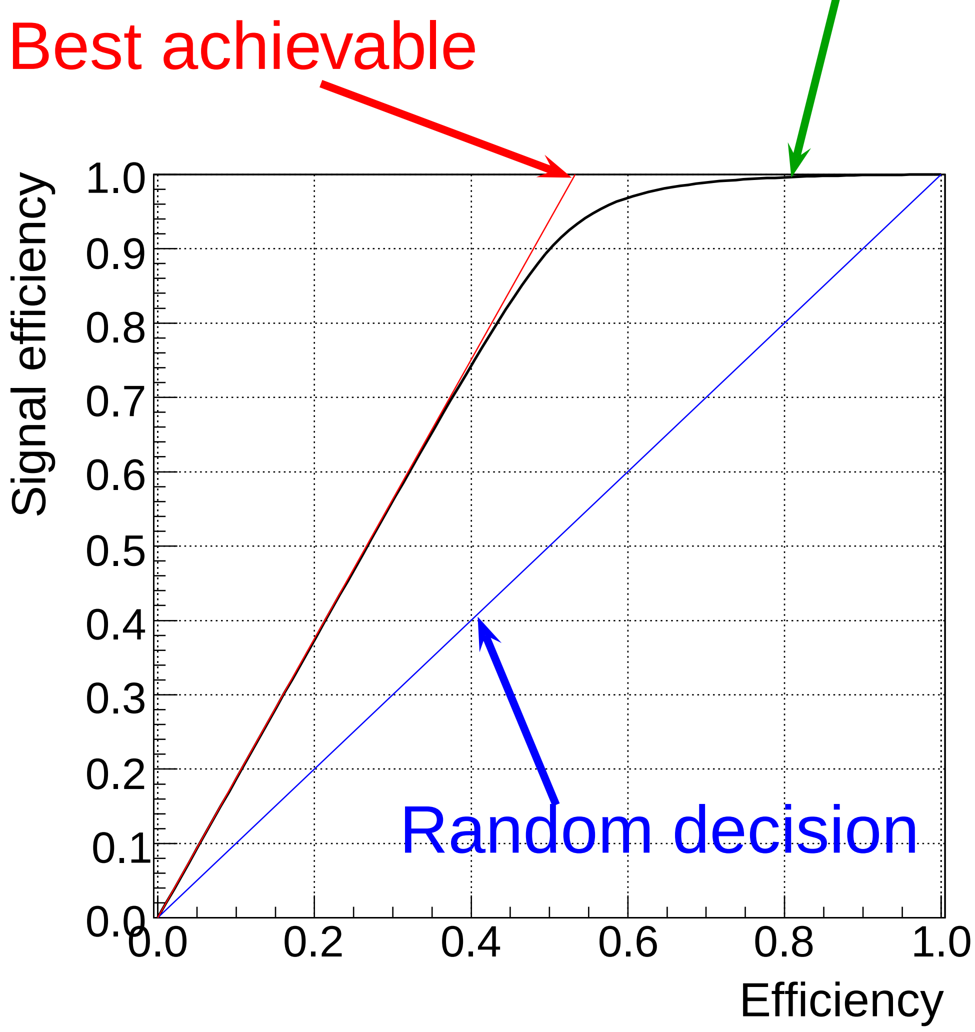
Each point has requirement on *out* $<$ *cut*

How to check quality of training

Define quantities:

- Signal efficiency:
  $$\epsilon_S = \frac{\text{N(selected signal)}}{\text{N(all signal)}}$$
- Efficiency:
  $$\epsilon = \frac{\text{N(selected)}}{\text{N(all)}}$$
- Purity:
  $$P = \frac{\text{N(selected signal)}}{\text{N(selected)}}$$

Each point has requirement on *out* $>$ *cut*



Best achievable

Random decision

Issues in training

Main issue: Finding minimum in high dimensional space

5 input nodes with 5 nodes in hidden layer and one output node gives $5x5 + 5 = 30$ weights

Imagine task to find deepest valley in the Alps (only 2 dimensions)

Being put to some random place, easy to find some minimum

Issues in training

Main issue: Finding minimum in high dimensional space

5 input nodes with 5 nodes in hidden layer and one output node gives $5x5 + 5 = 30$ weights

Imagine task to find deepest valley in the Alps (only 2 dimensions)

Being put to some random place, easy to find some minimum

But what is in next valley?

Issues in training

- In training, we start with:
  - Large number of weights which are far from optimal value
  - Input values in unknown ranges

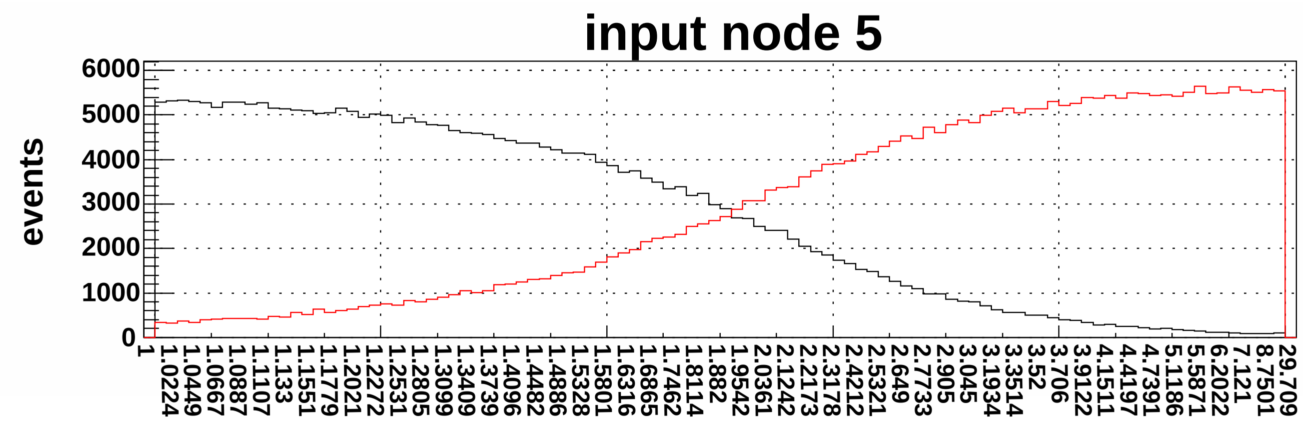
→ Large chance for numerical issues in calculation of loss function

- Statistical issues coming from low statistics in training
  - In many places in physics not that much issue as training statistics can be large
  - In industry can be quite big issue to obtain large enough training dataset
  - But also in physics it can be sometimes very costly to obtain large enough training sample
- Learning by heart, when NN has enough weights to do that

How to treat issues

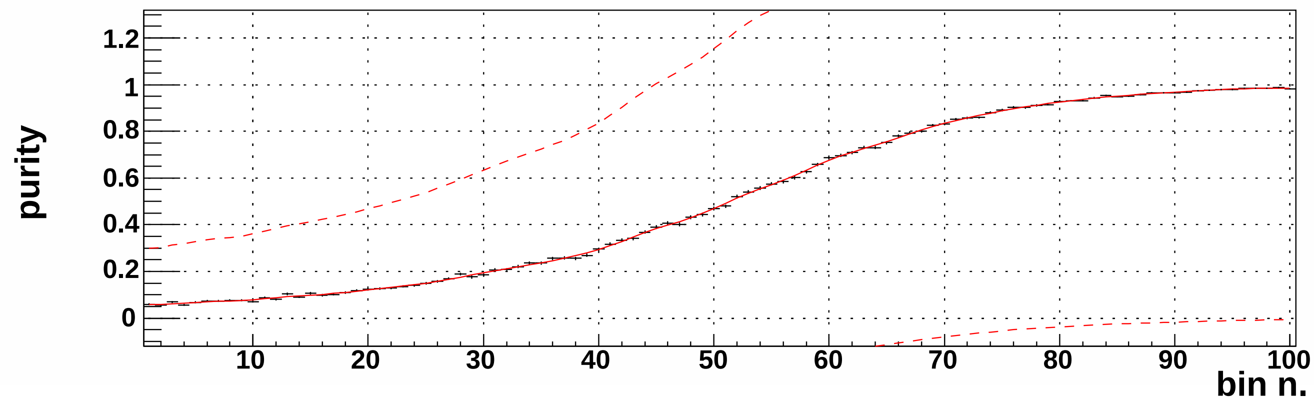
- All issues mentioned can be treated by neural network package in principle
- Regularisation for numerical issues at the beginning of training
- Weight decay - forgetting
  - Helps in overtraining on statistical fluctuations
  - Idea is that effects which are not significant are not shown often enough to keep weight to remember them
- Pruning of connections - remove connections, which don't carry significant information
- Preprocessing
  - numerical issues
  - search for good minimum by good starting point and decorrelation

Neurobayes preprocessing



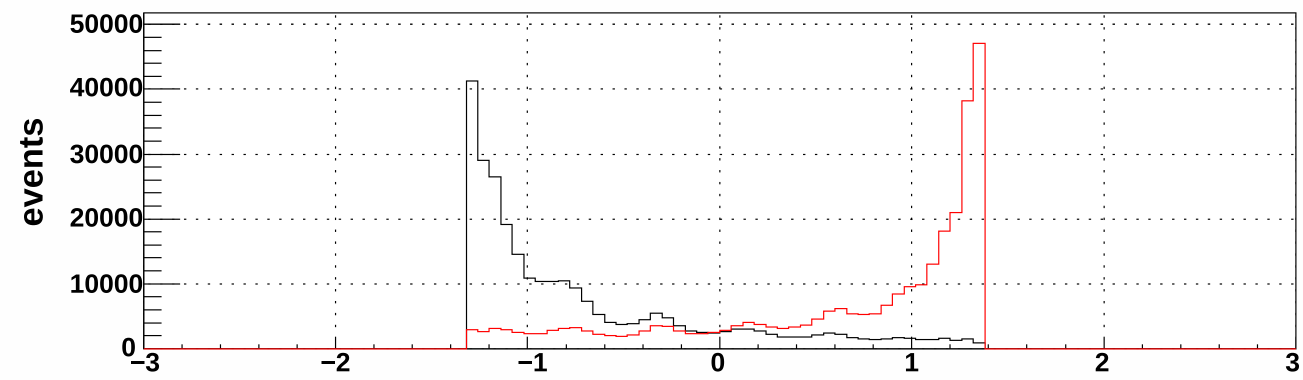
flat

Bin to bins of equal statistics



spline fit

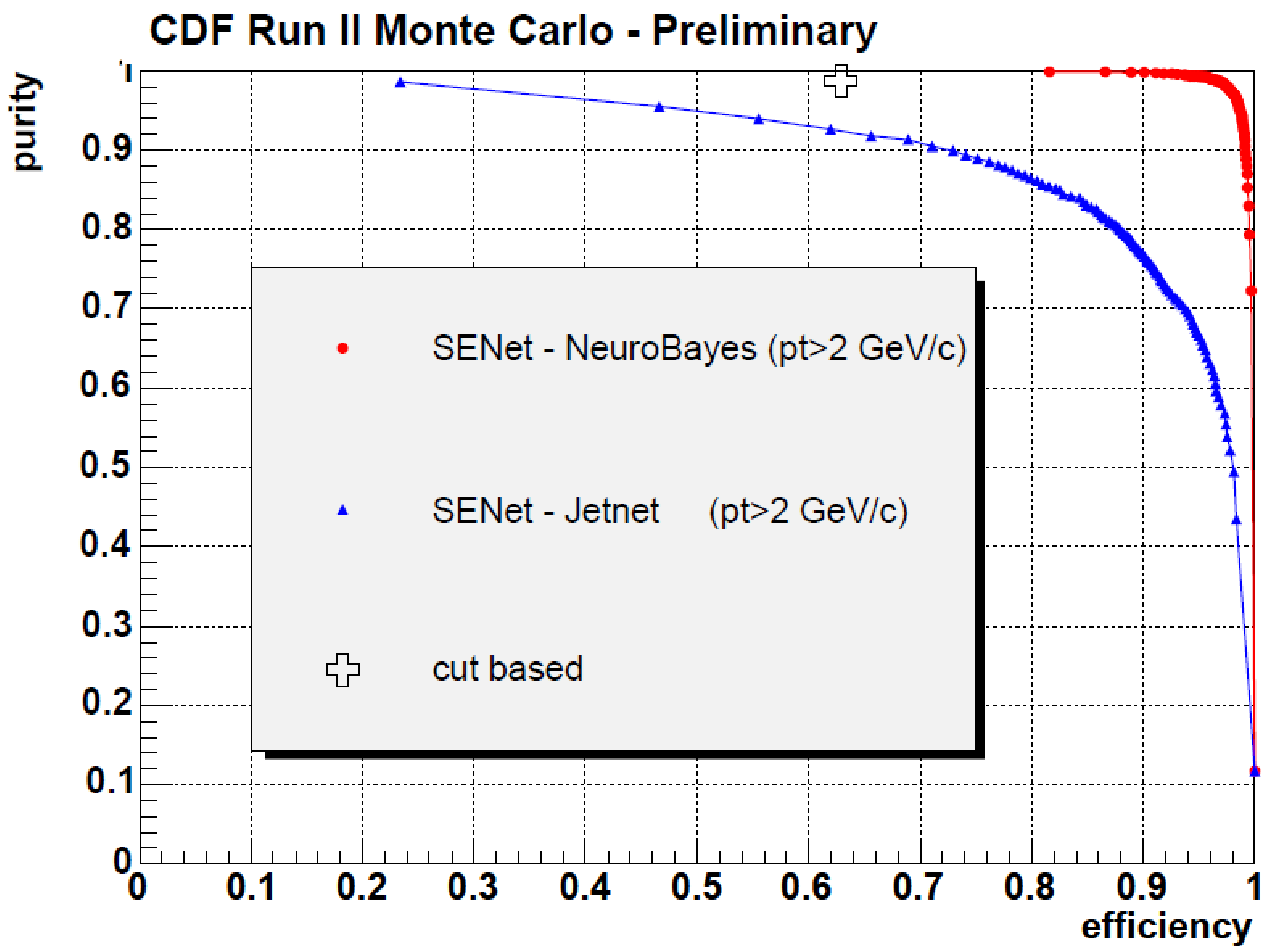
Calculate purity for each bin and do spline fit



final

Use purity instead of original value, plus transform to $\mu = 0$ and $RMS = 1$

Electron ID at CDF



Comparing two different NN packages with exactly same inputs

Preprocessing is important and can improve result significantly

Available packages

- Not exhaustive list
- NeuroBayes
  - http://www.phi-t.de
  - Best I know
  - Commercial, but fee is small for scientific use
  - Developed originally in high energy physics experiment
- ROOT/TMVA
  - Part of the ROOT package
  - Can be used for playing, but I would advise to use something else for serious applications
- SNNS: Stuttgart Neural Network Simulator
  - http://www.ra.cs.uni-tuebingen.de/SNNS/
  - Written in C, but ROOT interface exist
  - Best open source package I know about

NN vs. other multivariate techniques

- Several times I saw statements of type, my favourite method is better than other multivariate techniques
  - Usually lot of time is spend to understand and properly train favourite method
  - Other methods are tried just for quick comparison without understanding or tuning
  - Often you will even not learn any details of how other methods were setup
- With TMVA it is now easy to compare different methods - often I saw that Fischer discriminant won in those
  - From some tests we found that NN implemented in TMVA is relativily bad
  - No wonder that it is usually worst than other methods
- Usually it is useful to understand method you are using, none of them is black box

Practical tips

- No need for several hidden layers, one should be sufficient for you
- Number of nodes in hidden layer should not be very large

→ Risk of learning by heart

- Number of nodes in hidden layer should not be very small

→ Not enough capacity to learn your problem

→ Usually $\mathcal{O}$(number of inputs) in hidden layer

- Understand your problem, NN gets numbers and can dig information out of them, but you know meaning of those numbers

Practical tips

- Don’t throw away knowledge you acquired about problem before starting using NN

→ NN can do lot, but you don’t need that it discovers what you already know

  - If you already know that given event is background, throw it away
  - If you know about good inputs, use them rather than raw data from which they are calculated

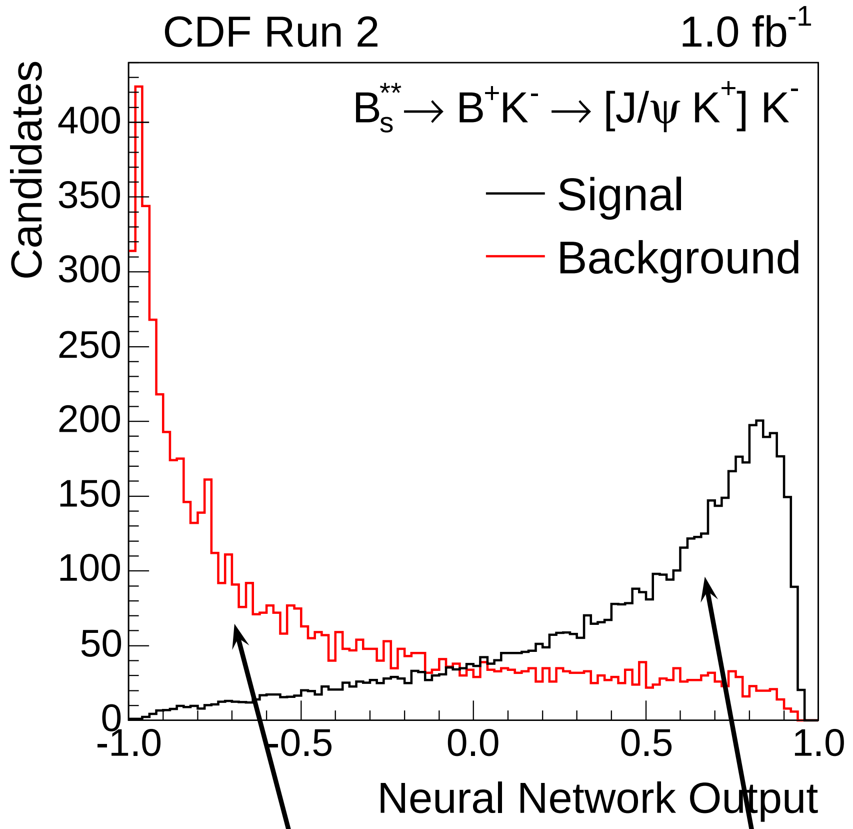
- Use neural network to learn what you don’t know yet

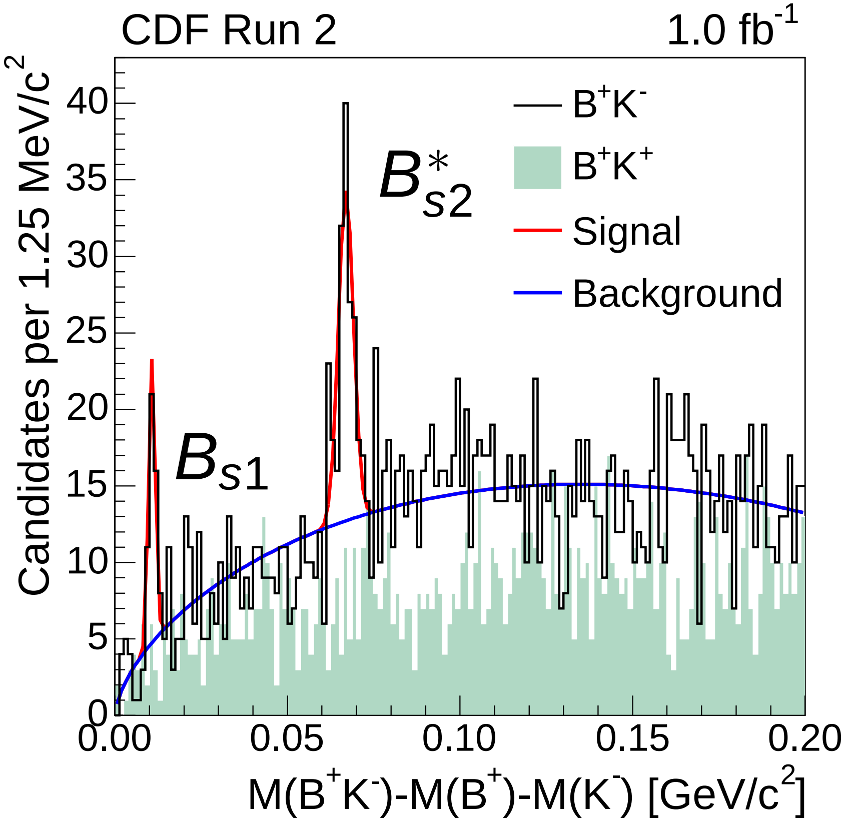
Physics examples

Only mentioning work done by Karlsruhe HEP group

- "Stable" particle identification at Delphi, CDF and now Belle
- Properties (E, ϕ, θ, Q-value) of inclusively reconstructed B mesons
- Decay time in semileptonic B_s decays
- Heavy quark resonances studies: $X(3872)$, B^{**}, B_s^{**}, Λ_c^*, Σ_c
- B flavour tagging and B_s mixing (Delphi, CDF, Belle)
- CPV in $B_s \to J/\psi\phi$ decays (CDF)
- B tagging for high p_T physics (CDF, CMS)
- Single top and Higgs search (CDF)
- B hadron full reconstruction (Belle)
- Optimisation of KEKB accelerator

B_s^{**} at CDF





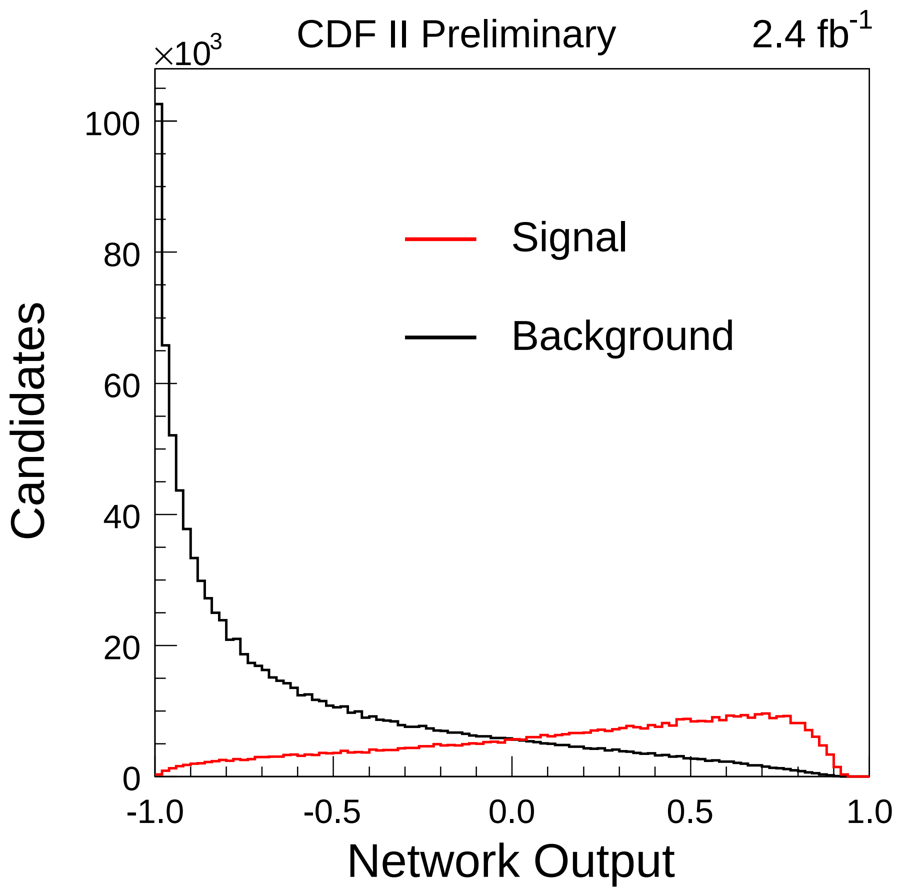
Background from data

Signal from MC

First observation of B_{s1}

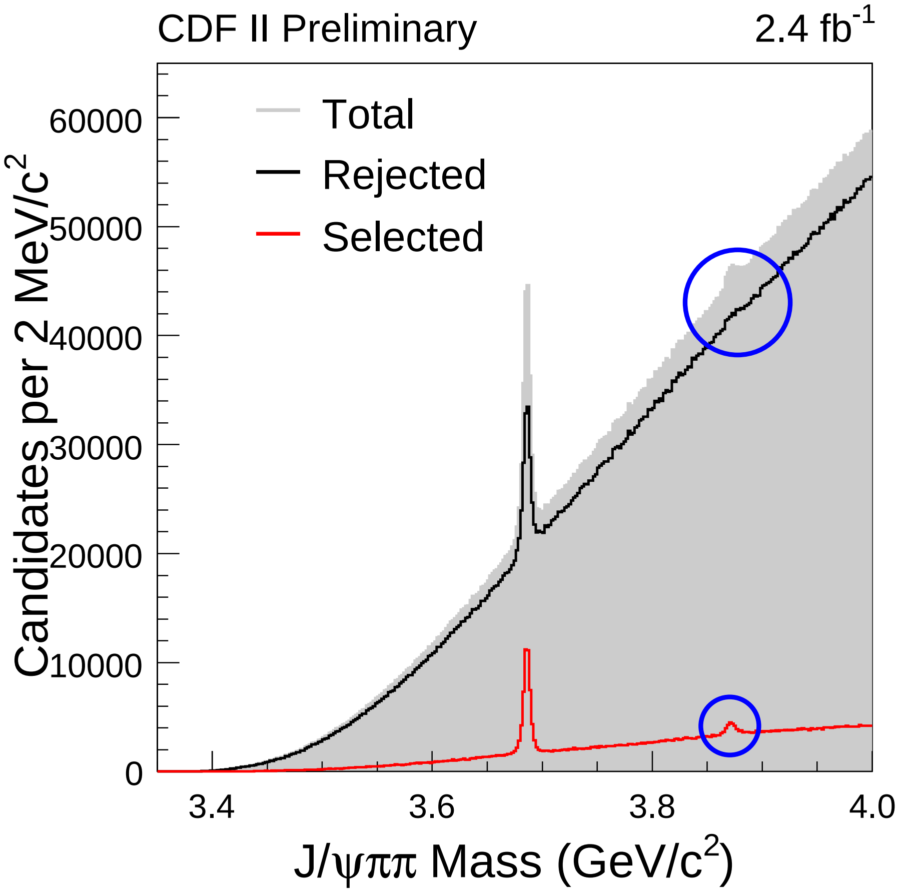
Most precise masses

X(3872) mass at CDF



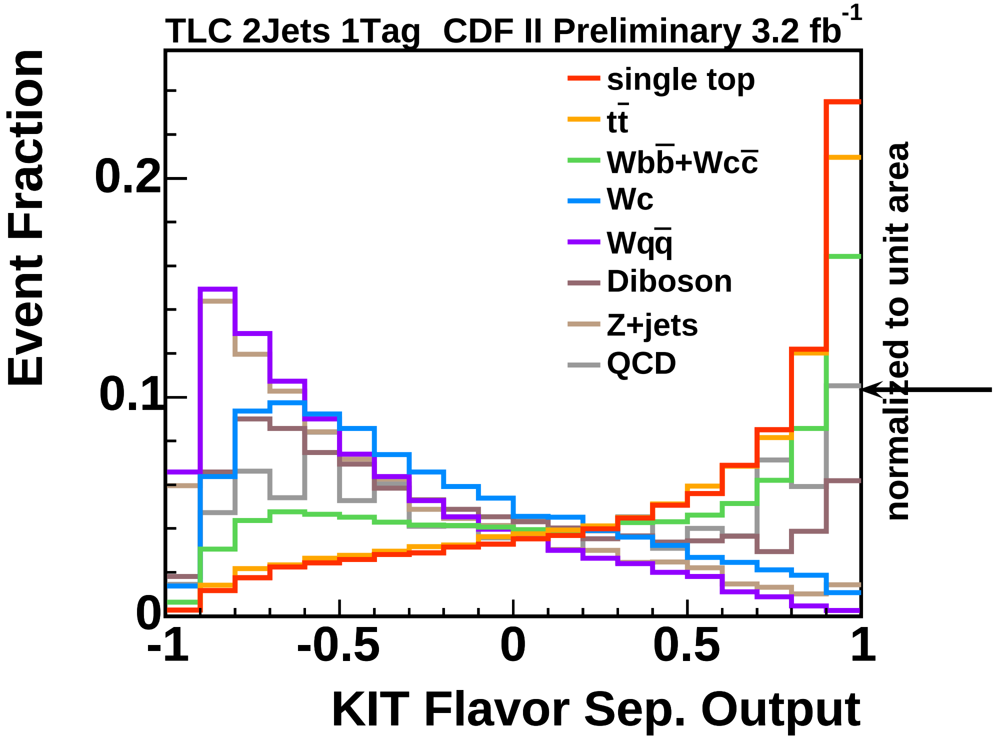
X(3872) first of charmonium like states

Lot of effort to understand its nature



Powerful selection at CDF of largest sample leading to best mass measurement

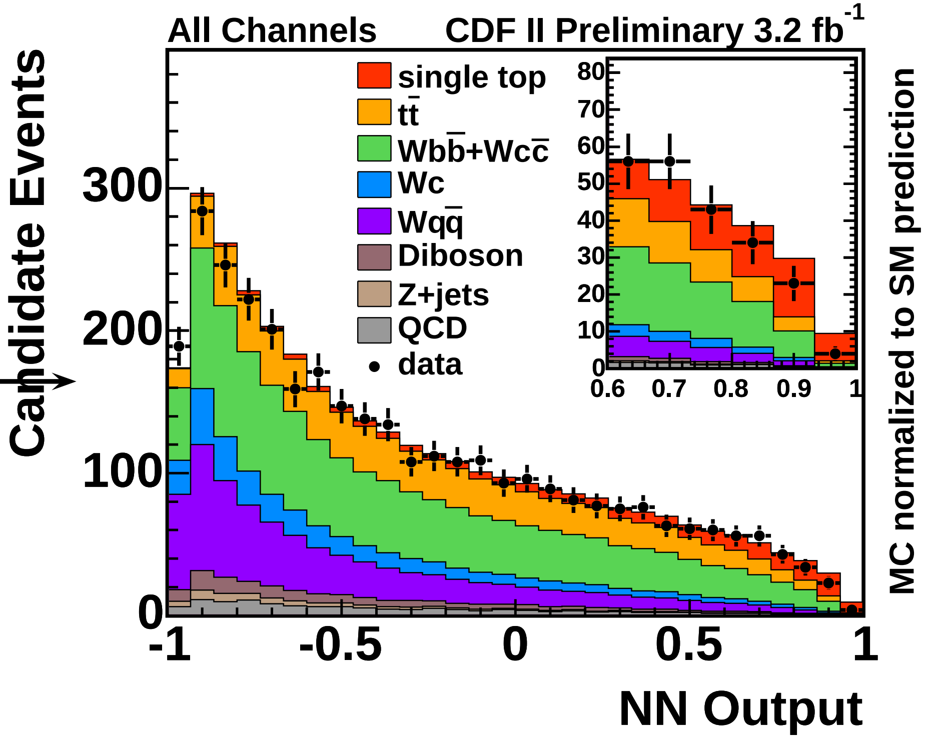
Single top search



Complicated analysis with many different multivariate techniques

Tagging whether jet is *b*-quark jet or not using NN

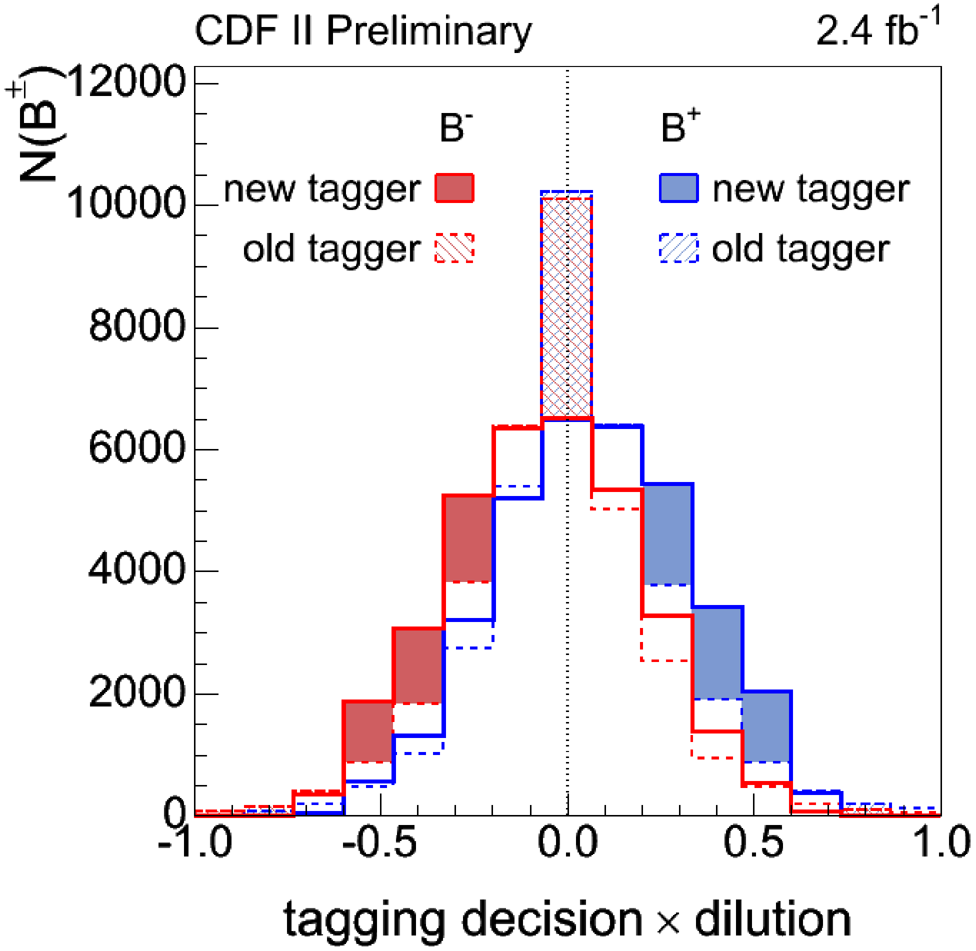
Final data plot



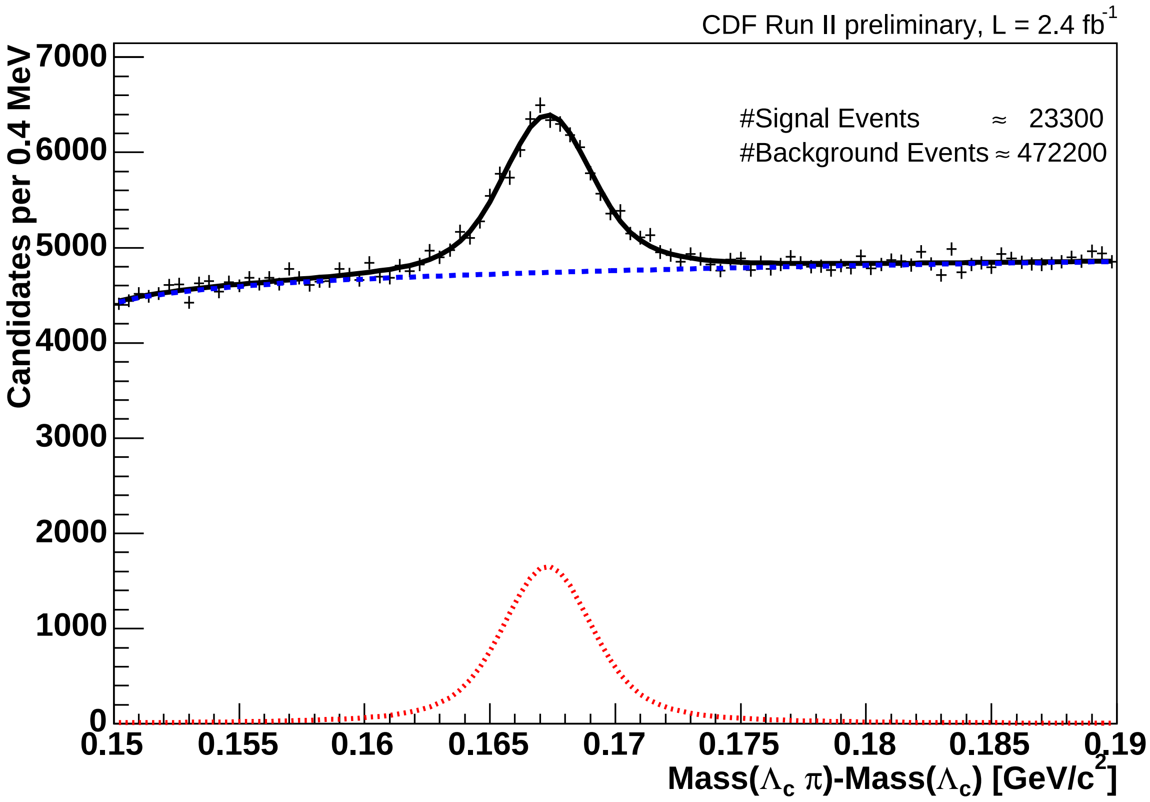
Part of big program leading to discovery of single top production

B flavour tagging - CDF

- Flavour tagging at hadron colliders is difficult task
- Lot of potential for improvements
- In plot, concentrate on continuous lines
- Example of not that good separation

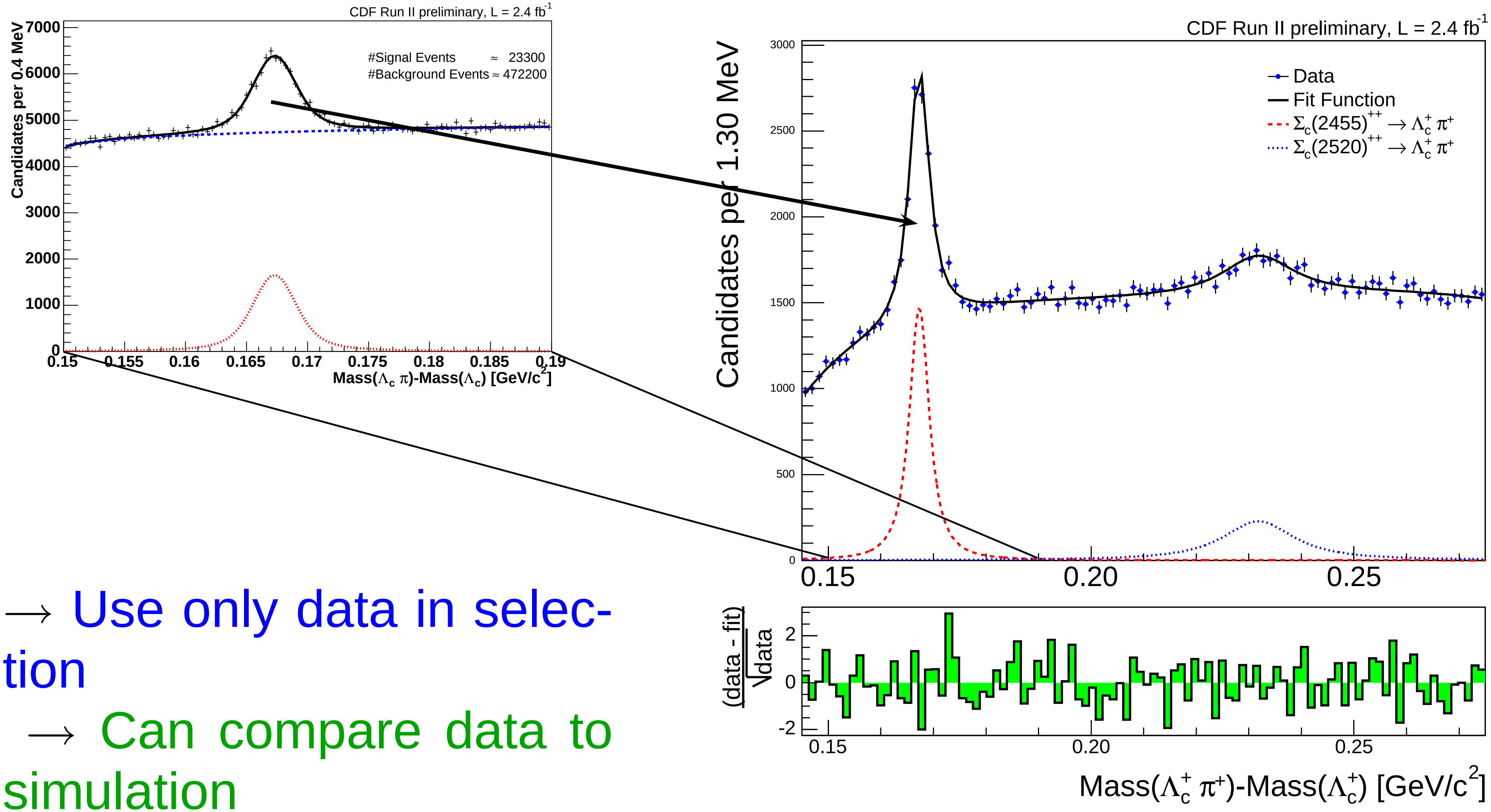


Training with weights



- One knows statistically whether event is signal or background
- Or has sample of mixture of signal and background and pure sample for one class
- Can use data only as example of application

Training with weights



→ Use only data in selection

→ Can compare data to simulation

→ Can reweight simulation to match data

Examples outside physics

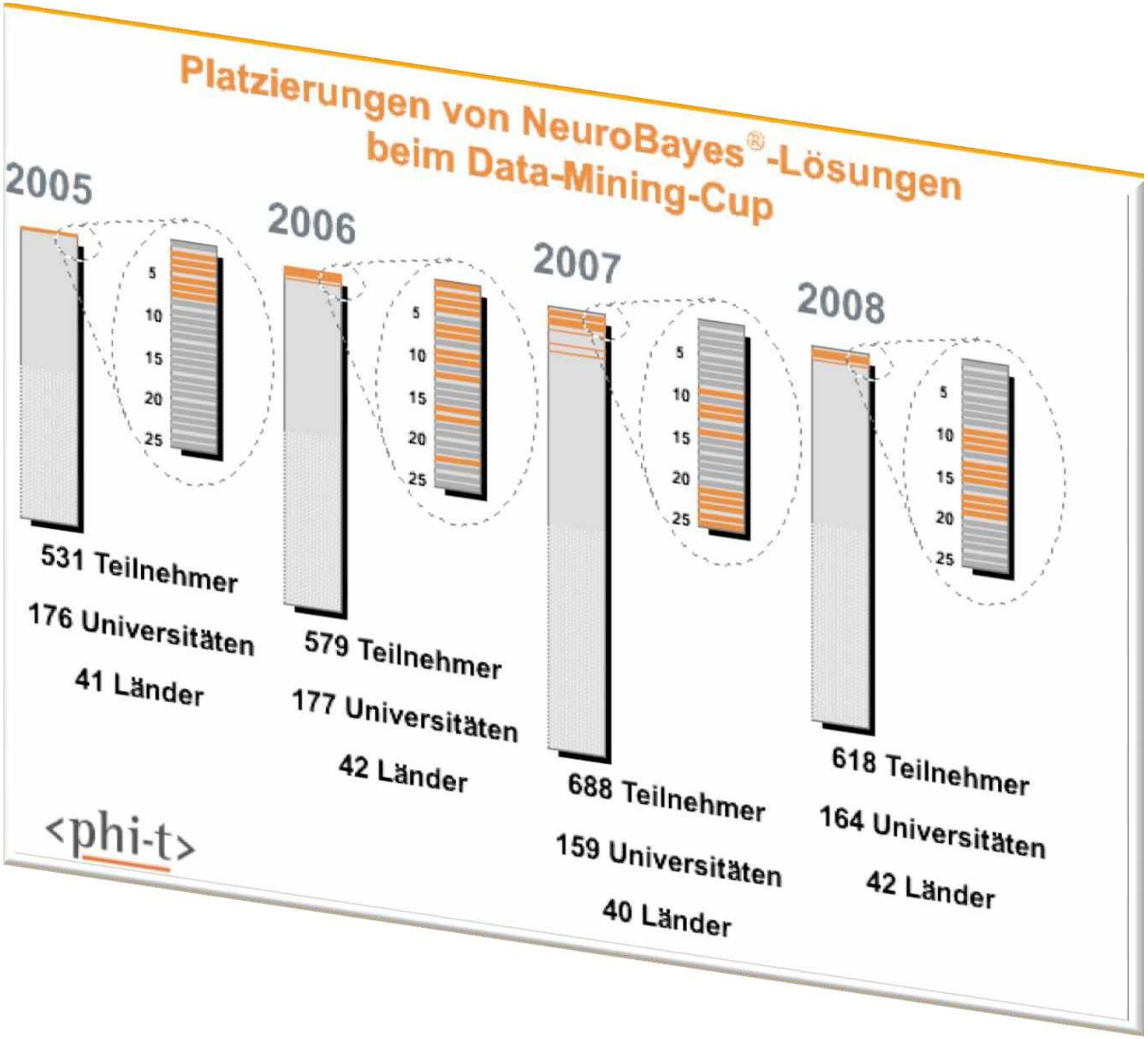
- Medicine and Pharma research
  e.g. effects and undesirable effects of drugs early tumor recognition
- Banks
  e.g. credit-scoring (Basel II), finance time series prediction, valuation of derivates, risk minimised trading strategies, client valuation
- Insurances
  e.g. risk and cost prediction for individual clients, probability of contract cancellation, fraud recognition, justice in tariffs
- Trading chain stores:
  turnover prognosis for individual articles/stores

Data Mining Cup

World‘s largest students competition: Data-Mining-Cup

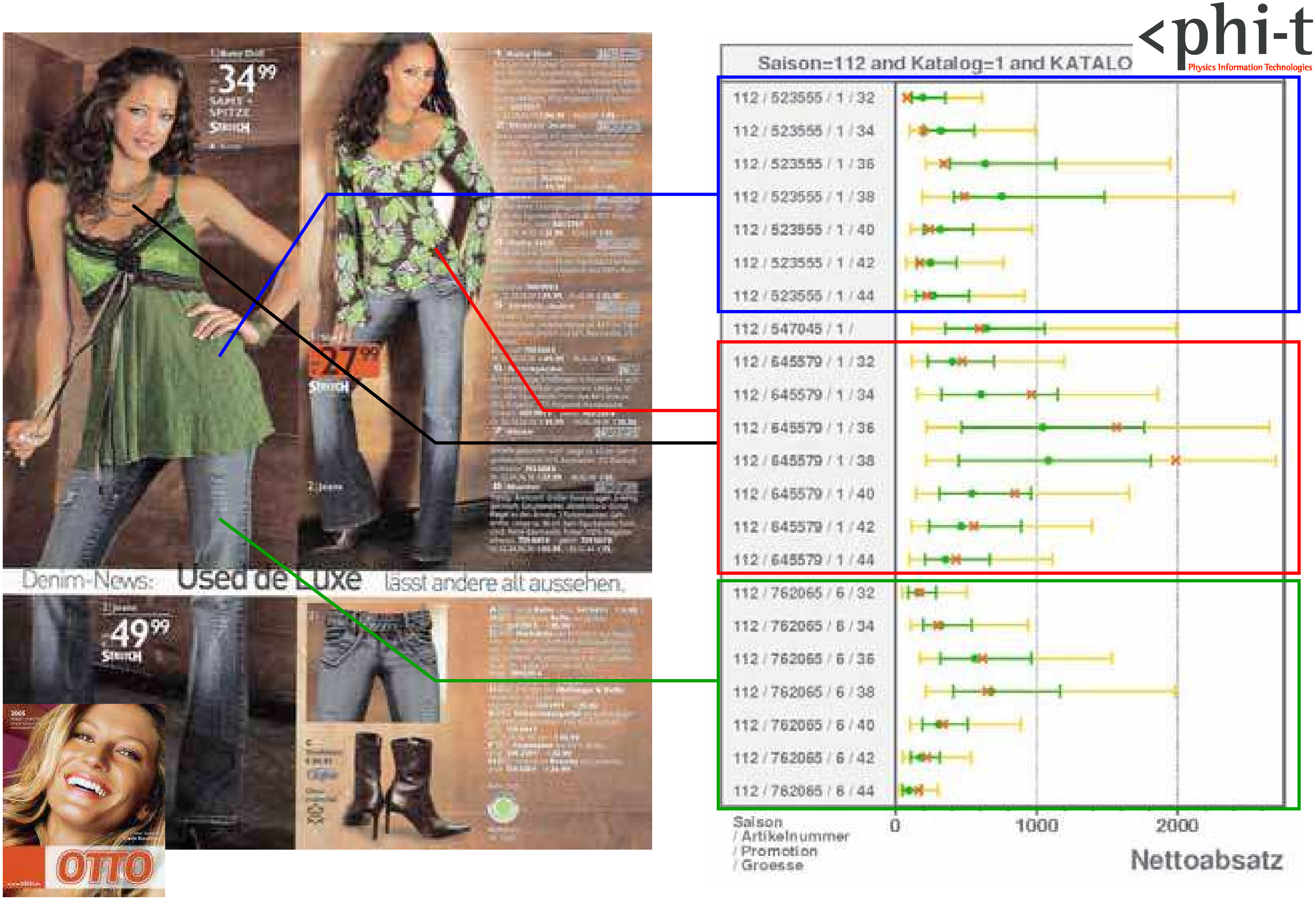
Very successful in competition with other data-mining methods

2005: Fraud detection in internet trading
2006: price prediction in ebay auctions
2007: coupon redemption prediction
2008: lottery customer behaviour prediction



NeuroBayes Turnover prognosis

<phi-t>®
Physics Information Technologies



Individual health costs

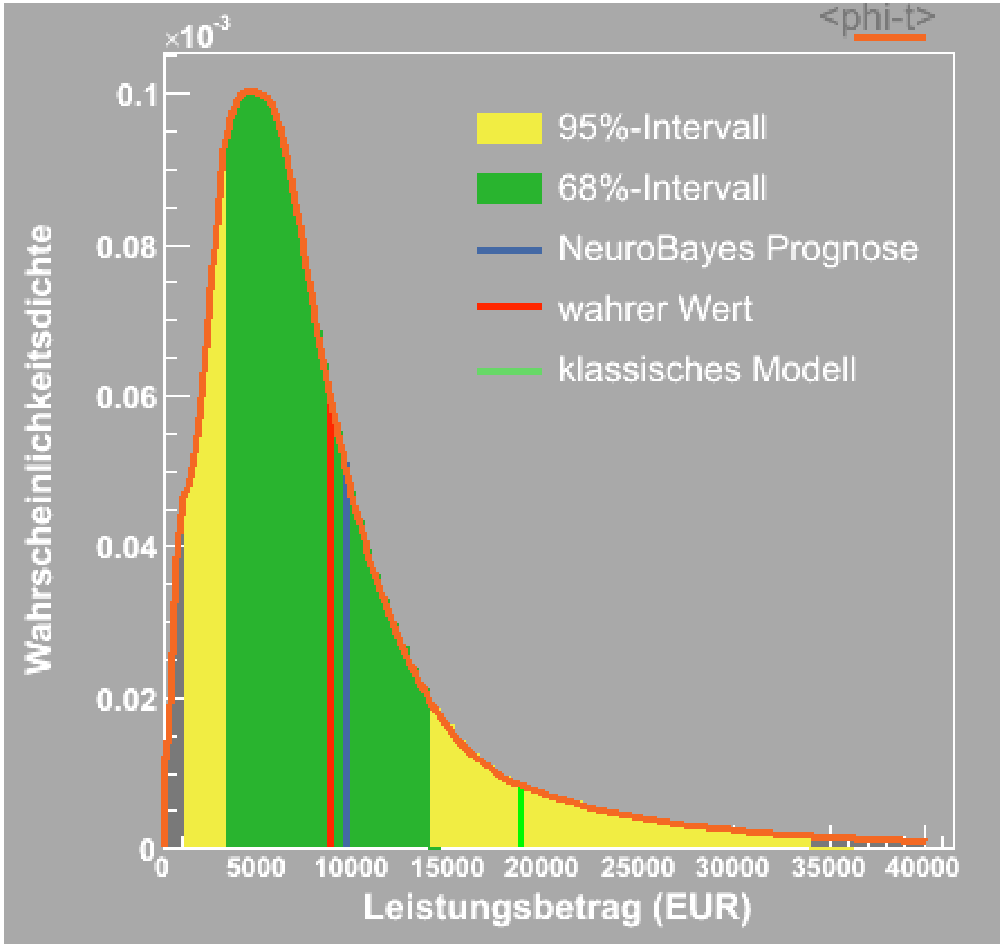
Pilot project for a large private health insurance

Prognosis of costs in following year for each person insured with confidence intervals

4 years of training, test on following year

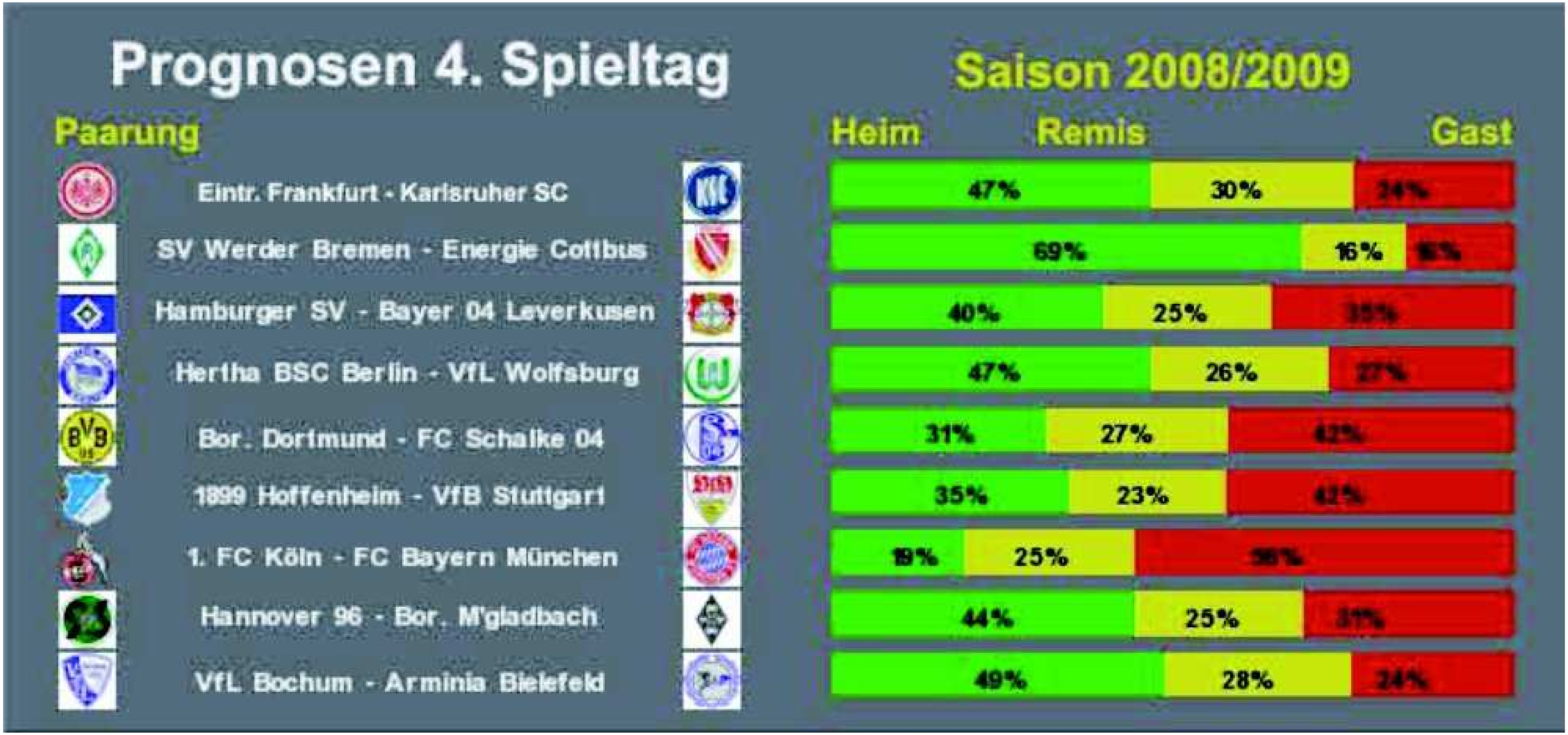
Results: Probability density for each customer/tariff combination

Very good test results!



Potential for an objective cost reduction in health management

Prognosis of sport events



Prognosen 4. Spieltag

Saison 2008/2009

Paarung	Heim	Remis	Gast
Eintr. Frankfurt - Karlsruher SC	47%	30%	24%
SV Werder Bremen - Energie Cottbus	69%	16%	15%
Hamburger SV - Bayer 04 Leverkusen	40%	25%	35%
Hertha BSC Berlin - VfL Wolfsburg	47%	26%	27%
Bor. Dortmund - FC Schalke 04	31%	27%	42%
1899 Hoffenheim - VfB Stuttgart	35%	23%	42%
1. FC Köln - FC Bayern München	19%	25%	56%
Hannover 96 - Bor. M'gladbach	44%	25%	31%
VfL Bochum - Arminia Bielefeld	49%	28%	24%

Results: Probabilities for home - tie - guest

Conclusions

- Neural network is powerful tool for your analysis
- They are not genius, but can do marvellous things for genius
- Lecture of this kind cannot really make you expert
- Hope that I was successful with:
  - Give you enough insight so that neural network is not black box anymore
  - Teaching you enough not to be afraid to try
- Feel free to talk to me, I’m at Campus Süd, building 30.23, room 9-11