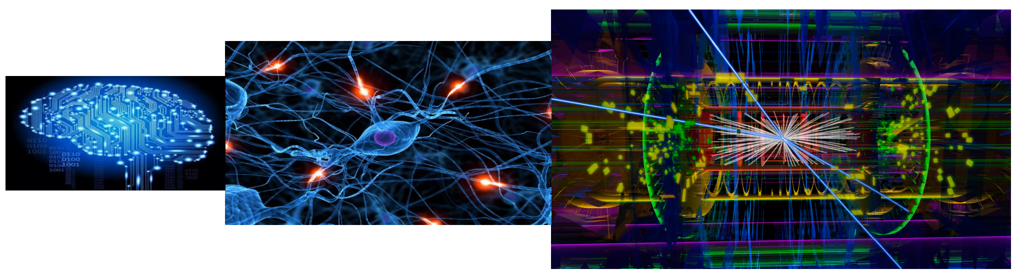
MVA Techniques II: Advanced Methods

S. Gleyzer[1], H. Prosper[2], C. Rosemann[1]

[1]DESY, [2]Florida State University



DESY Statistics School 2014

April 3, 2014

Part II

- Focus on advanced multivariate methods
  - Methods in **Theory**
    - Overview
  - Methods in **Practice**
    - Classification Tutorials
      - Simple gaussians, $H \rightarrow ZZ \rightarrow 4l$ example
    - Regression Tutorials
      - Calorimetry exercise, function estimation with Bayesian Neural Networks

Resources

Literature

G. James, et al. “Introduction to Statistical Learning” Springer 2013

C.M. Bishop “Pattern Recognition and Machine Learning” Springer 2006

J. R. Quinlan “C4.5: Programs for Machine Learning” Morgan Kaufmann 1992

Talks/Tutorials

Harrison Prosper’s INFN statistics school 2013 talk

https://agenda.infn.it/getFile.py/access?contribId=11&resId=1&materialId=slides&confId=571

TMVA @ Root Users Workshop 2013

http://indico.cern.ch/event/217511/contribution/37/material/slides/0.pdf

Past DESY Statistics schools

http://www.terascale.de/schools_and_workshops/

Tools

TMVA: by A. Höcker et al. http://tmva.sourceforge.net

SPR: by I. Narsky http://statpatrec.sourceforge.net/

R: http://www.rproject.org

MLPFit by Jerome Schwindling
http://schwind.web.cern.ch/schwind/MLPfit.html

C4.5/C5.0 by J.R. Quinlan
http://www.rulequest.com/Personal/c4.5r8.tar.gz

Rulefit by J. Friedman http://statweb.stanford.edu/~jhf/R_RuleFit.html

CLUS http://dtai.cs.kuleuven.be/clus/

Applications

Many in HEP

Some typical use cases

- **Classification**
  - Particle identification (ID)
    - Is this a **pear** (photon/electron) or an **apple** (jet)?
  - Searches for new physics
    - Various MVA methods used by past and current physics analyses to find new physics or set limits on theoretical models
      - Does this look like SUSY or background event?
- **Regression**
  - Calorimetry
    - Energy deposited by particles in non-compensating multi-layered calorimeter better measured by a function of individual energy deposits, cluster shapes obtained with MVA

Classification Methods

Available Methods

Again a *short* list of multivariate (MVA) methods that can be used for classification:

- Random Grid Search
- Fisher Discriminant
- Quadratic Discriminant
- Naïve Bayes (Likelihood)
- Kernel Density Estimation
- Binary Decision Trees
- Neural Networks
- Bayesian Neural Networks
- Support Vector Machines
- Random Forests
- Genetic Algorithms
- Predictive Clustering

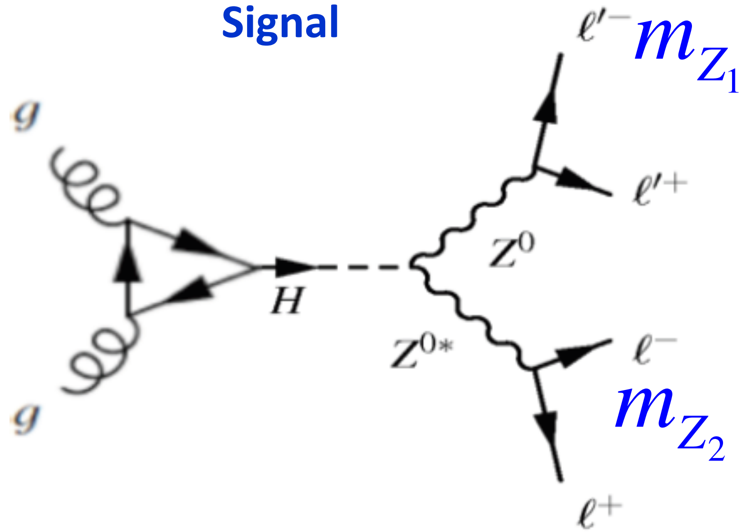
Lecture I

Lecture II

Illustrative Example

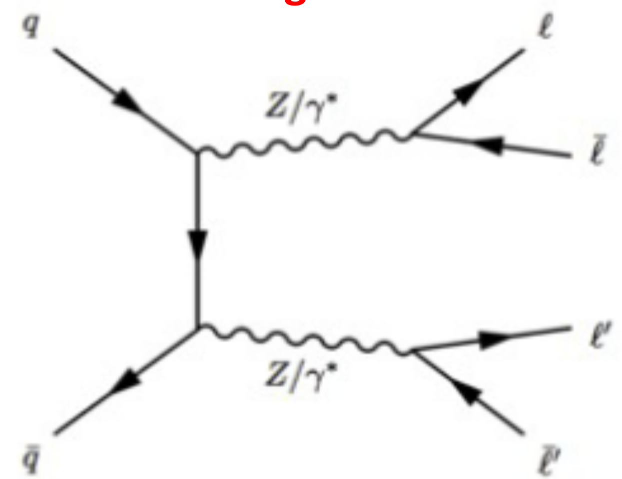
H to ZZ to 4 Leptons

Signal



$$pp \to H \to ZZ \to \ell^+\ell^-\ell'^+\ell'^-$$

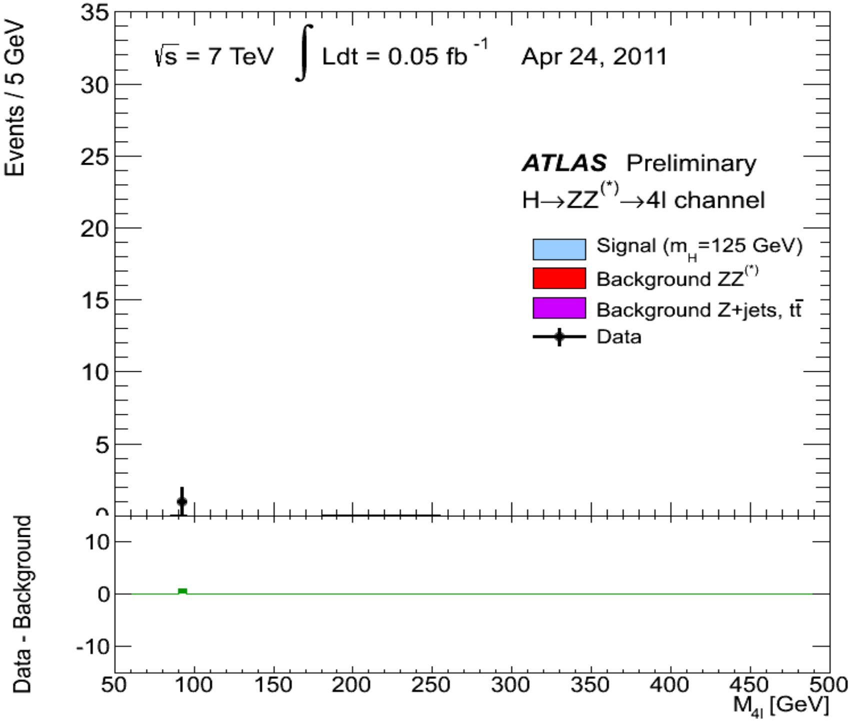
Background



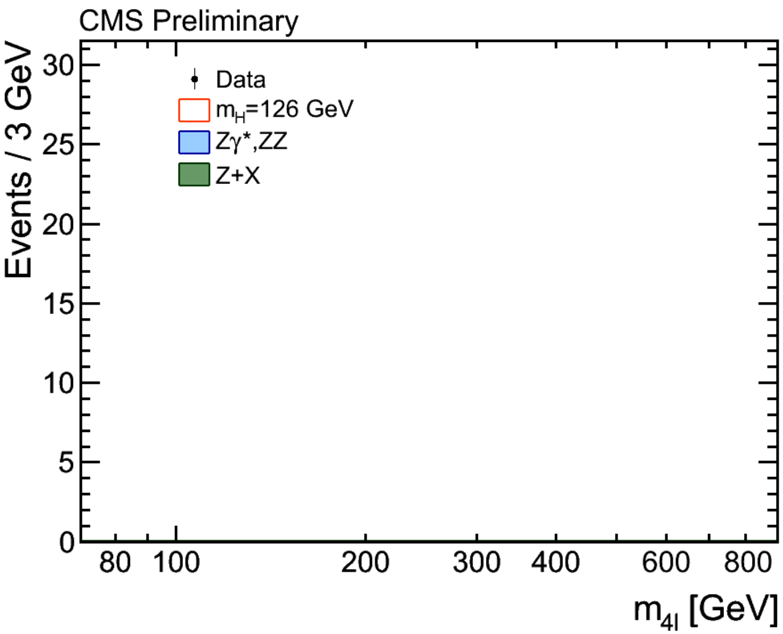
$$pp \to ZZ \to \ell^+\ell^-\ell'^+\ell'^-$$

We'll (re)use this example to illustrate a few of the methods.

H to ZZ to 4 Leptons

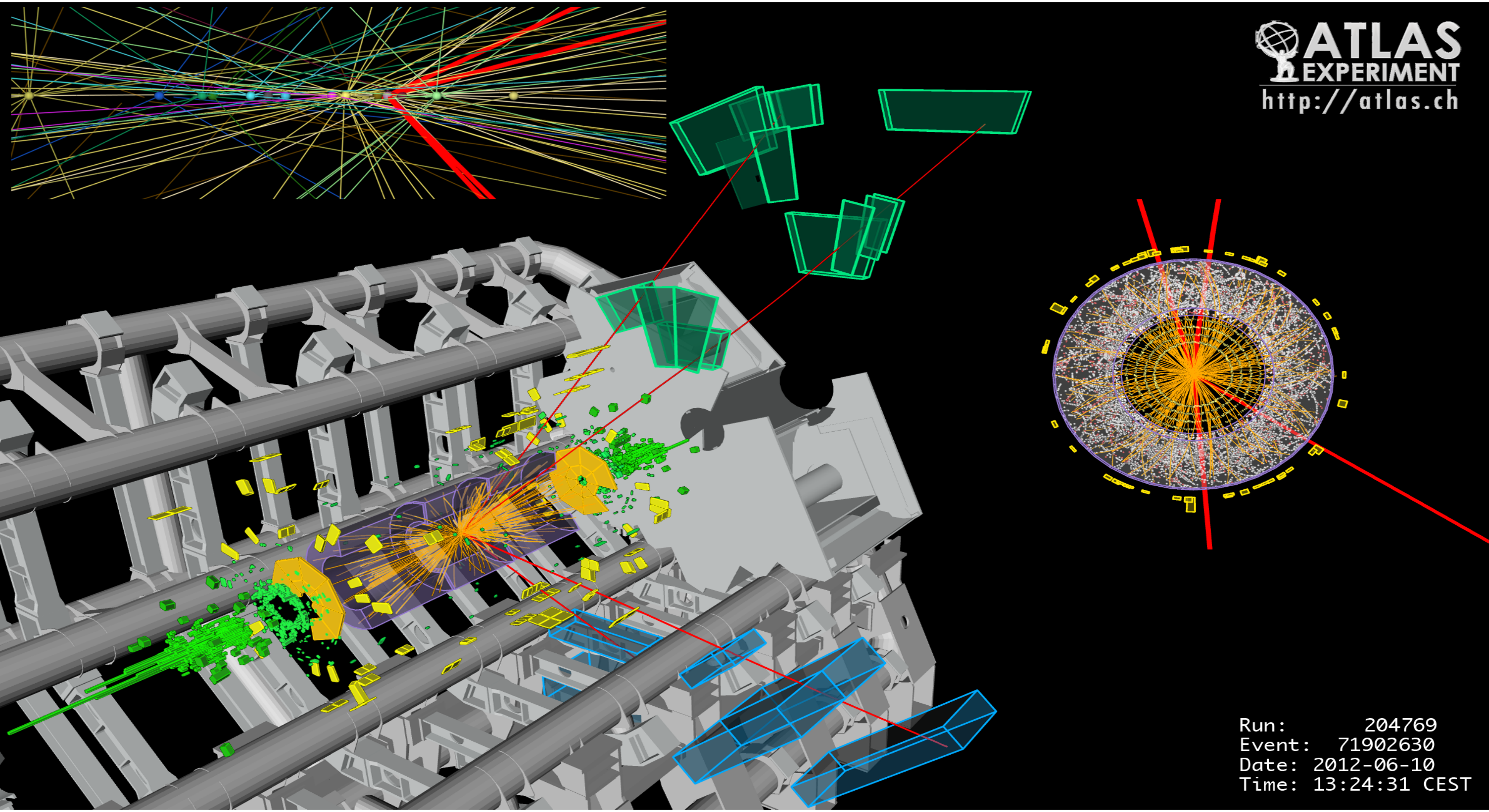


ATLAS

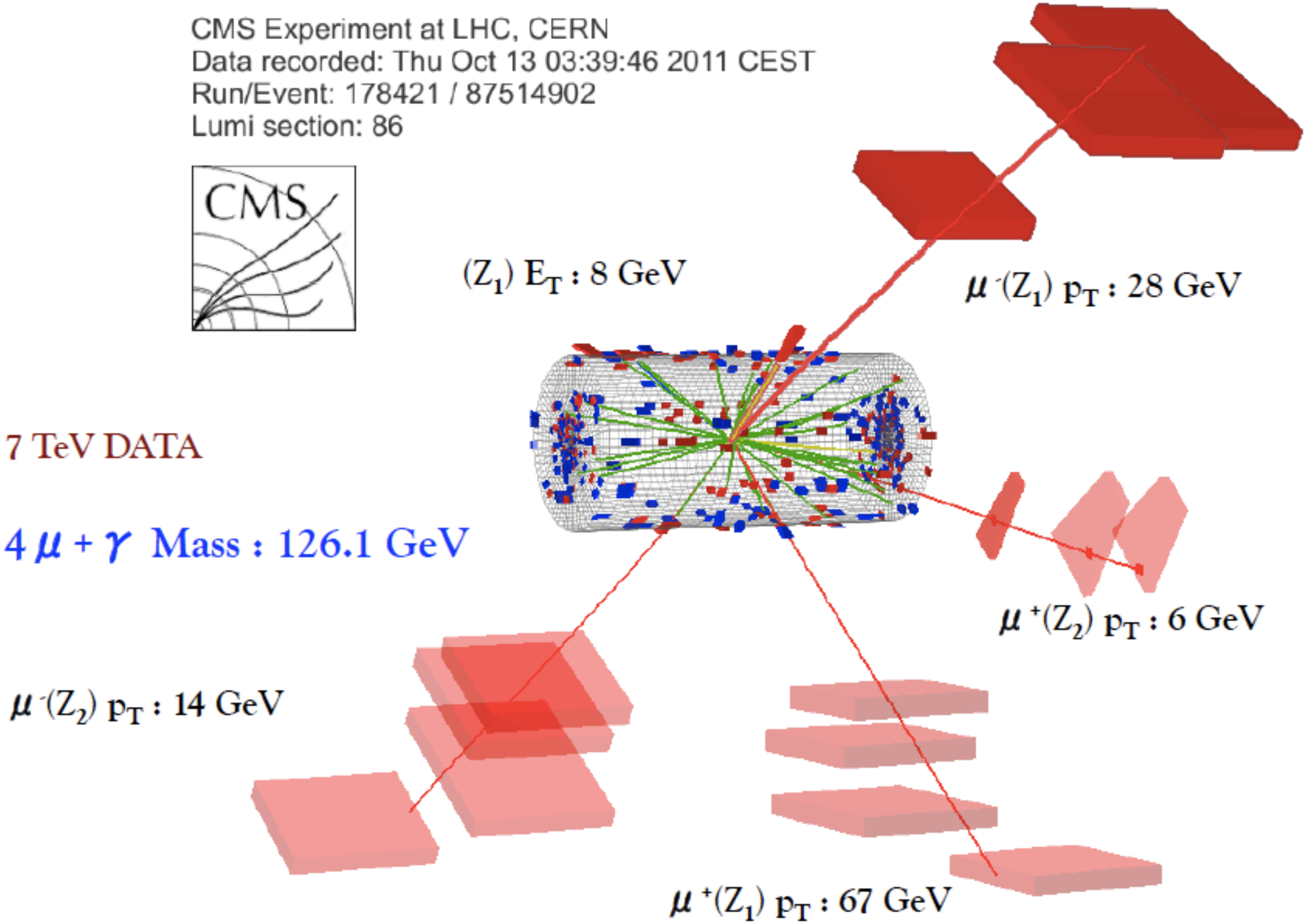


CMS

4-Lepton Event ATLAS



4-Lepton Event CMS



Feature Selection

Feature Selection

Picking variables:

- Usually pick variables (features) that show standalone discrimination power
- Try to cover all degrees of freedom
  - don't worry if you end up with a few extra variables, they can be winnowed afterwards

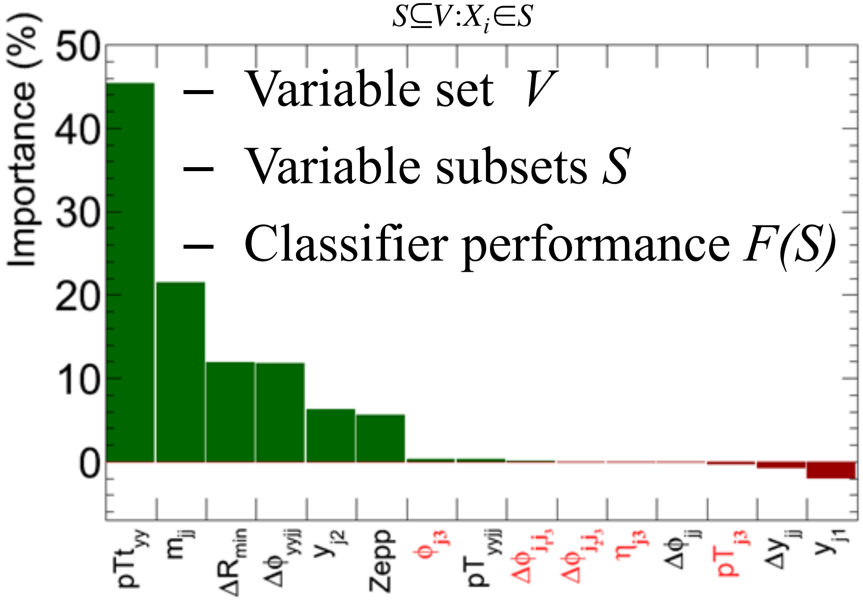
Important! how well the variables work together in the classifier

Feature Selection

Paradigm: tool for variable selection in classification context

- Variable importance → proportional to classifier performance in which variable participates

$$VI(X_i) = \sum_{S \subseteq V : X_i \in S} F(S) \times W_{X_i}(S)$$

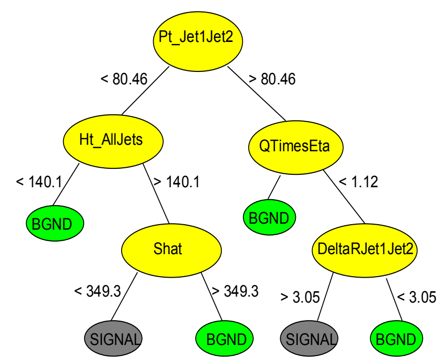


- Variable set V
- Variable subsets S
- Classifier performance $F(S)$

$$W_{X_i}(S) \equiv 1 - \frac{F(S - \{X_i\})}{F(S)}$$

Amount of classifier loss (or gain) if variable X_i is removed

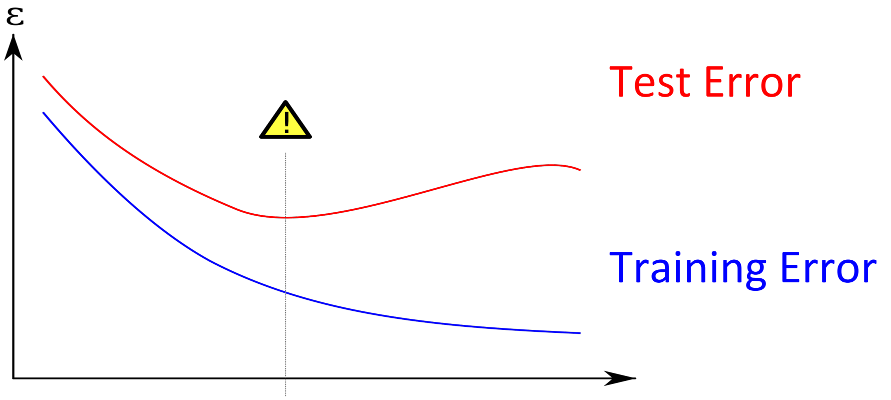
Building Classifiers



Training

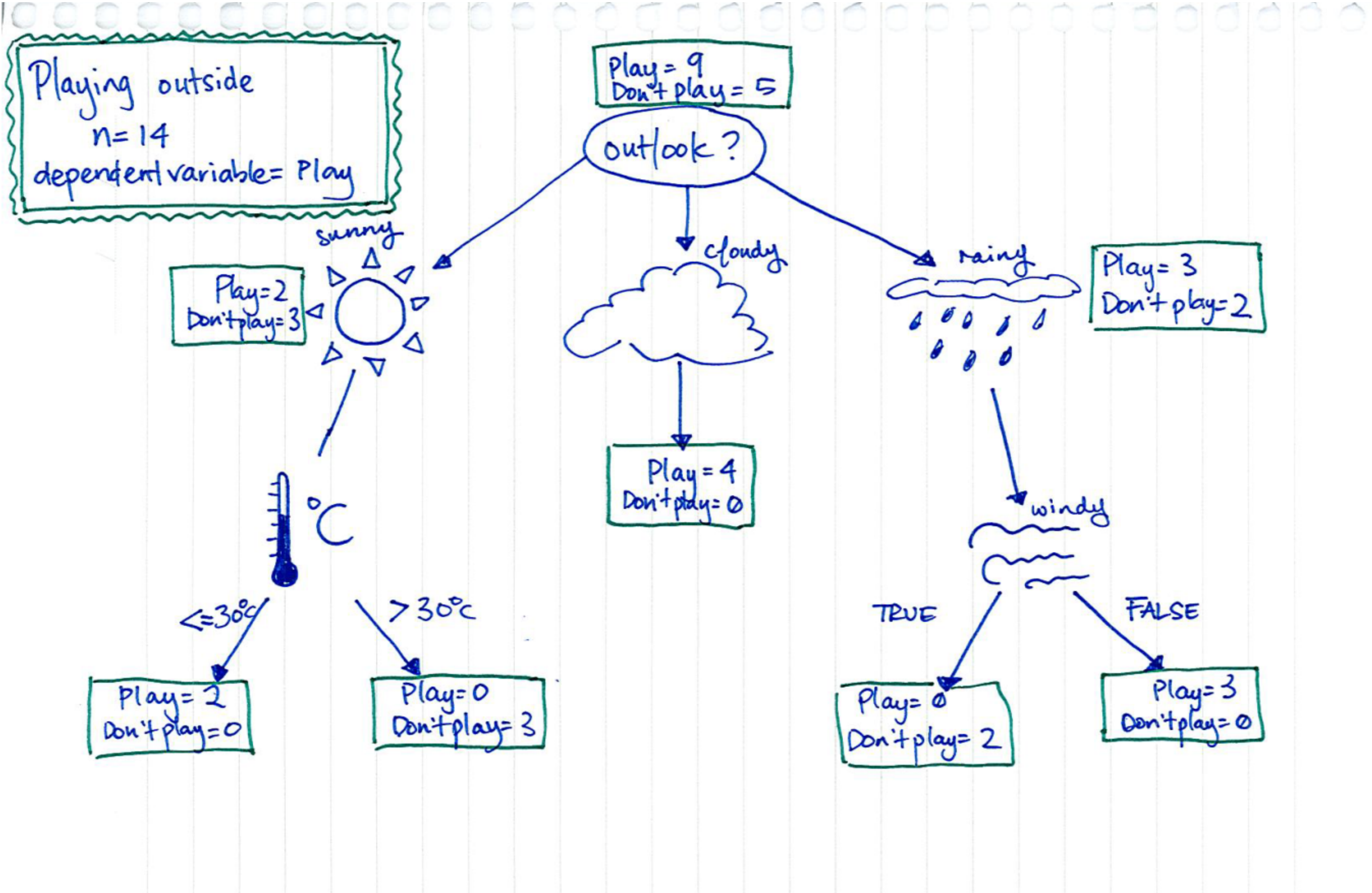
In supervised (as opposed to unsupervised or semi-supervised) learning scenario, data are labeled at the training stage

- Split data into at least two sets
  - Keep training and test (evaluation) sets separated



  - Monitor training/test error rates
    - Watch out for overtraining

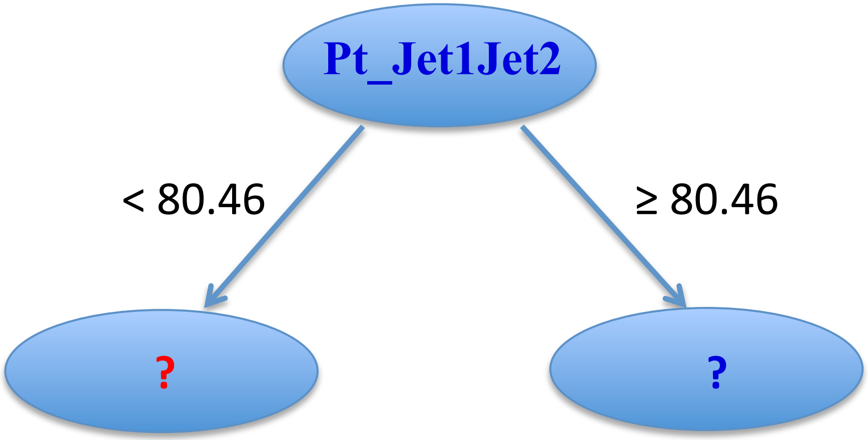
Binary Decision Trees


Playing outside
n= 14
dependent variable= Play
Play = 9
Don't play = 5
outlook ?
sunny
Play=2
Don't play=3
cloudy
Play = 4
Don't play= 0
rainy
Play= 3
Don't play= 2
°C
<=30°C
> 30°C
Play= 2
Don't play= 0
Play= 0
Don't play= 3
windy
TRUE
FALSE
Play= 0
Don't play= 2
Play= 3
Don't play= 0

Binary Decision Trees

Building a tree:

- Scan along each variable and propose a **DECISION**
  - A cut on value that maximizes class separation (binary branching)

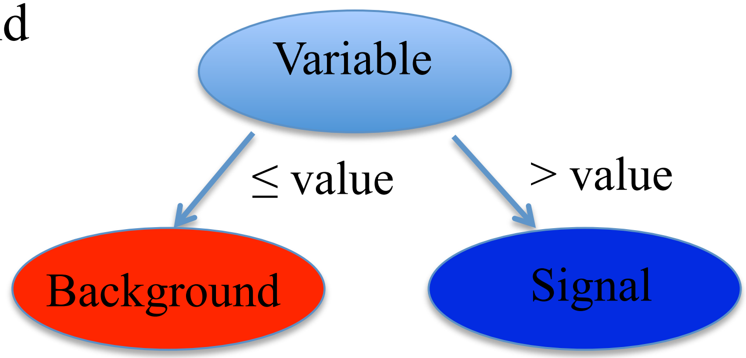


Binary Decision Trees

Building a tree:

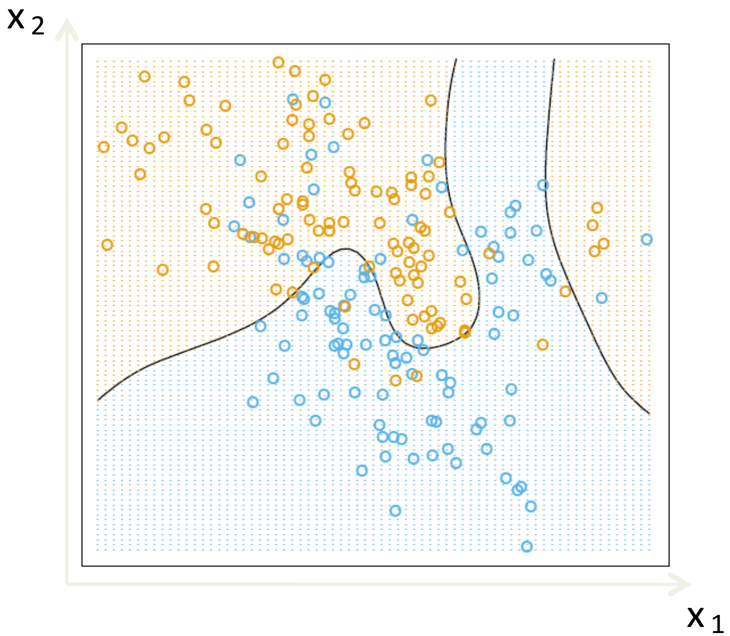
- Choose **decision** that leads to greatest separation among classes **signal**/**background**
  - Based on the information gained from split
    - Build regions of increasing purity
    - Stop when no further improvement from additional branching
    - Reach terminal node (leaf) and assign purity-based class

$$\frac{N_{signal}}{N_{signal} + N_{background}}$$

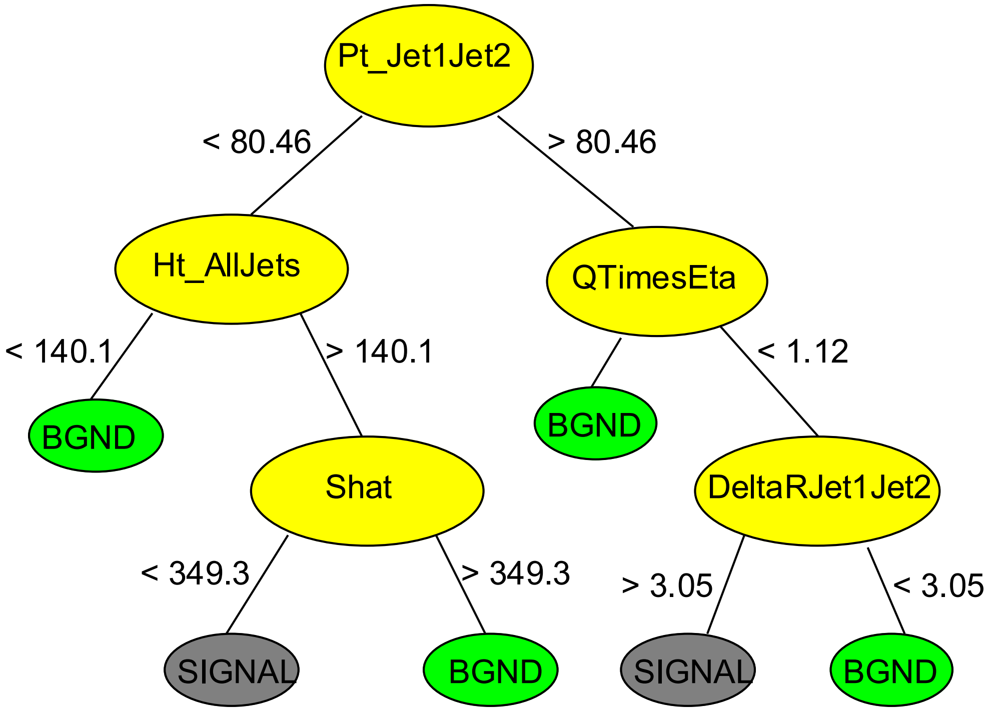


Representation

2-D Example



Typical N-D Tree



Pruning

Decision trees can become large and complex and risk over-fitting the data

Pruning: remove parts of the tree that are less powerful or possibly noisy

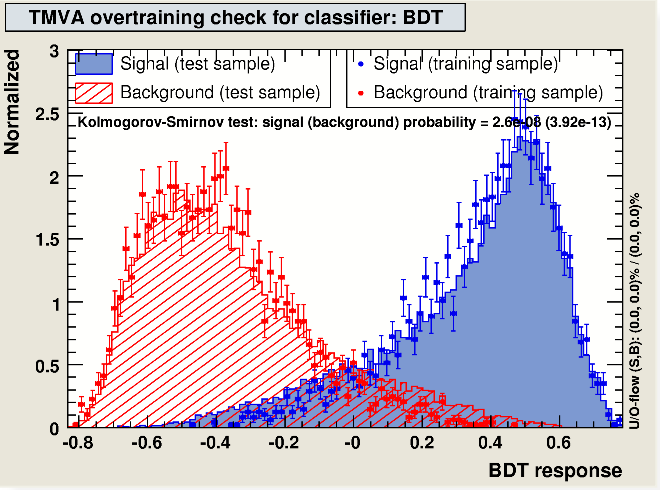
– start from the leaves and work back up

Pruned trees smaller in size, easier to interpret

Over-Training

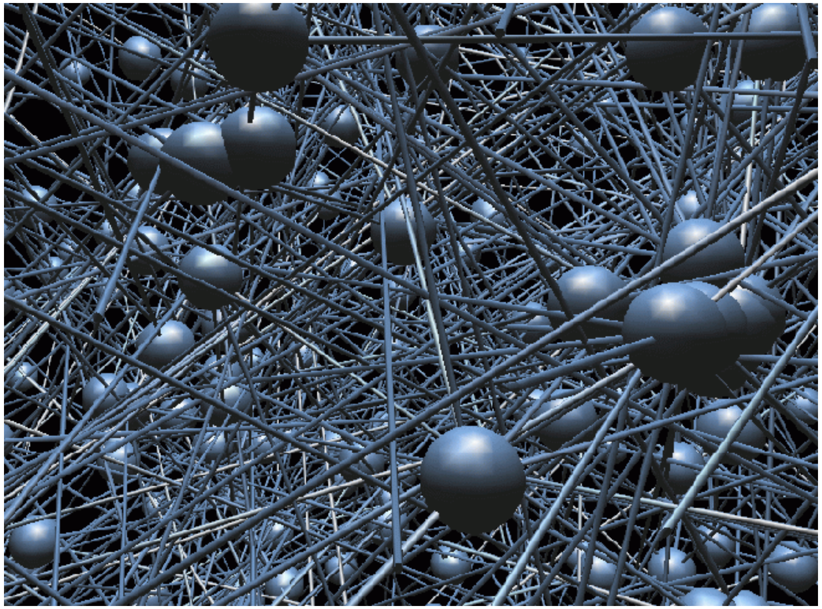
Over-training or over-fitting sometimes occurs when too many parameters for data size

- **Diagnose with**
  - divergent Training/ Testing error slopes
  - K-S test
- **Treat with**
  - Prune decision trees
  - Winnow: reduce number of parameters

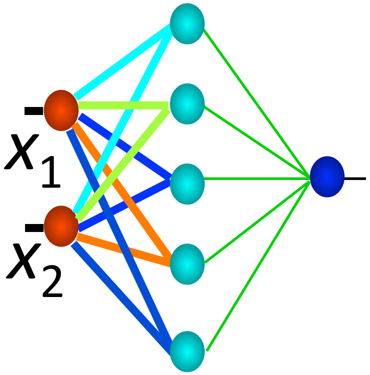


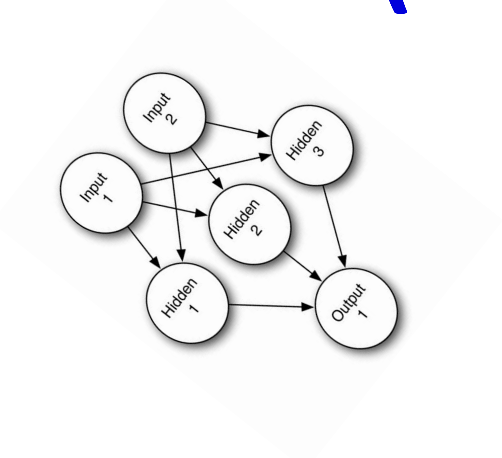
PHYSICS AT THE TERA SCALE
Helmholtz Alliance

DESY

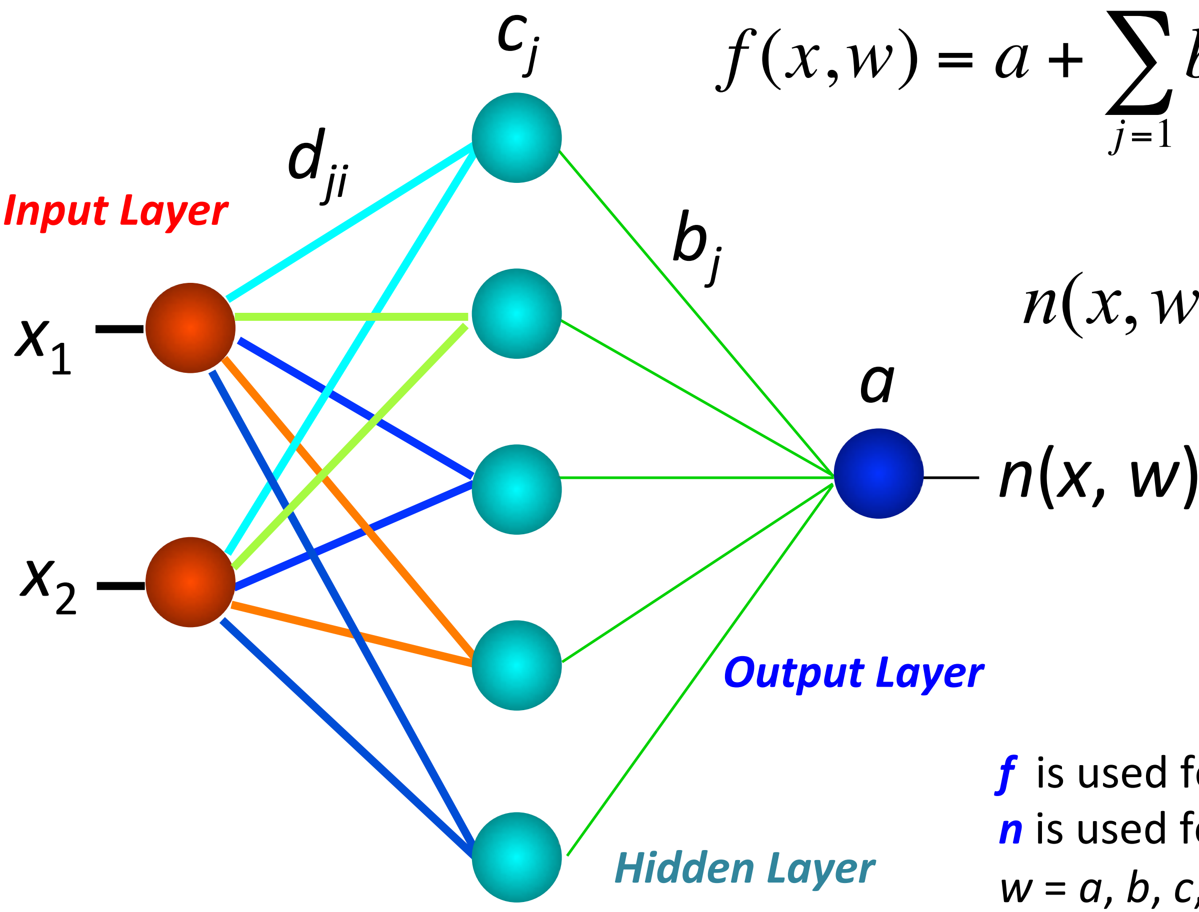


Neural Networks (NN)



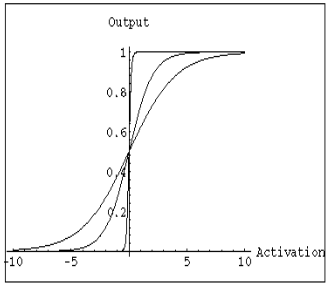


Graphical Representation



$$f(x,w) = a + \sum_{j=1}^{H} b_j \tanh\left[c_j + \sum_{i=1}^{I} d_{ji} x_i \right]$$

$$n(x,w) = \frac{1}{1 + \exp[-f(x,w)]}$$

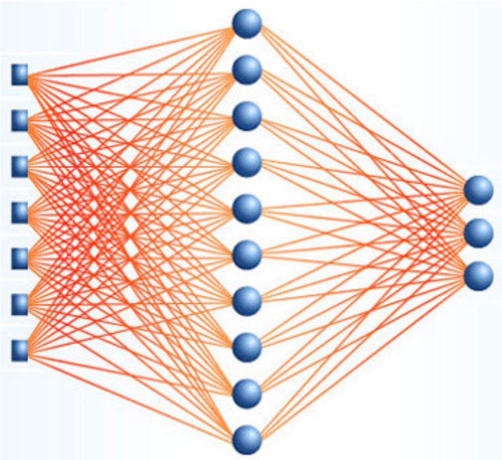


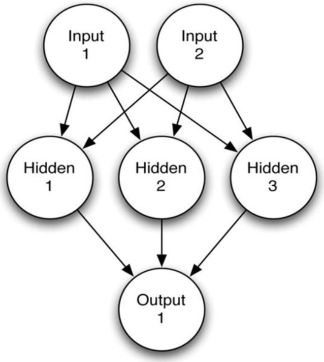
f is used for regression
n is used for classification
w = a, b, c, d

Network Weights

Compute optimal network weights with derivatives dE/dw

– Calculate gradients of errors for adjustable weights





Inputs go forward in feed-forward neural networks

Errors go backward! **Back-propagation**

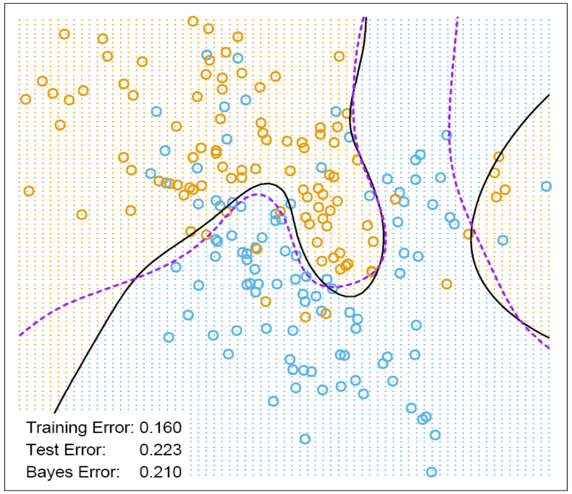
Neural Networks

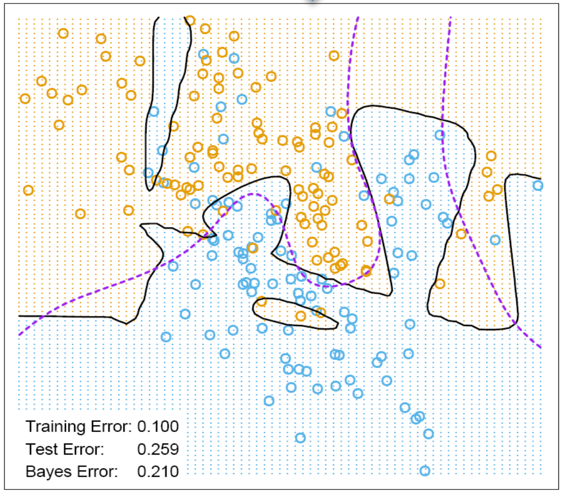
Can approximate any continuous function

Complexity determined by number of hidden layers and hidden nodes/layer

Many types of neural networks!

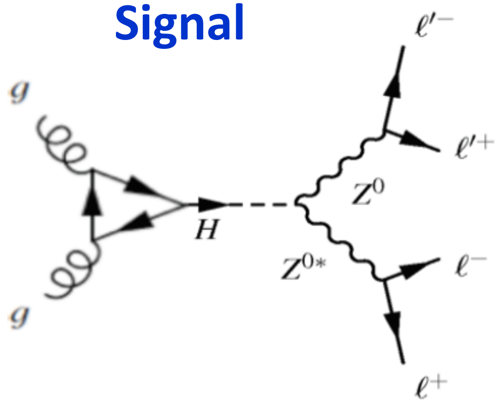
Watch out for overtraining





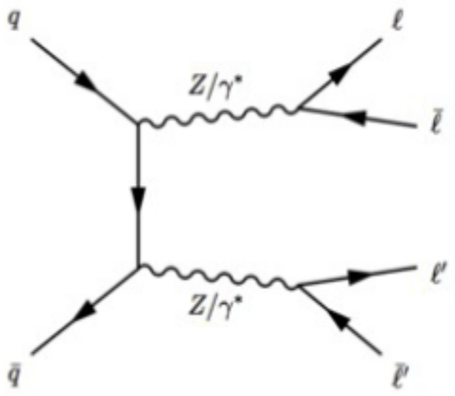
H to ZZ to 4Leptons

Signal

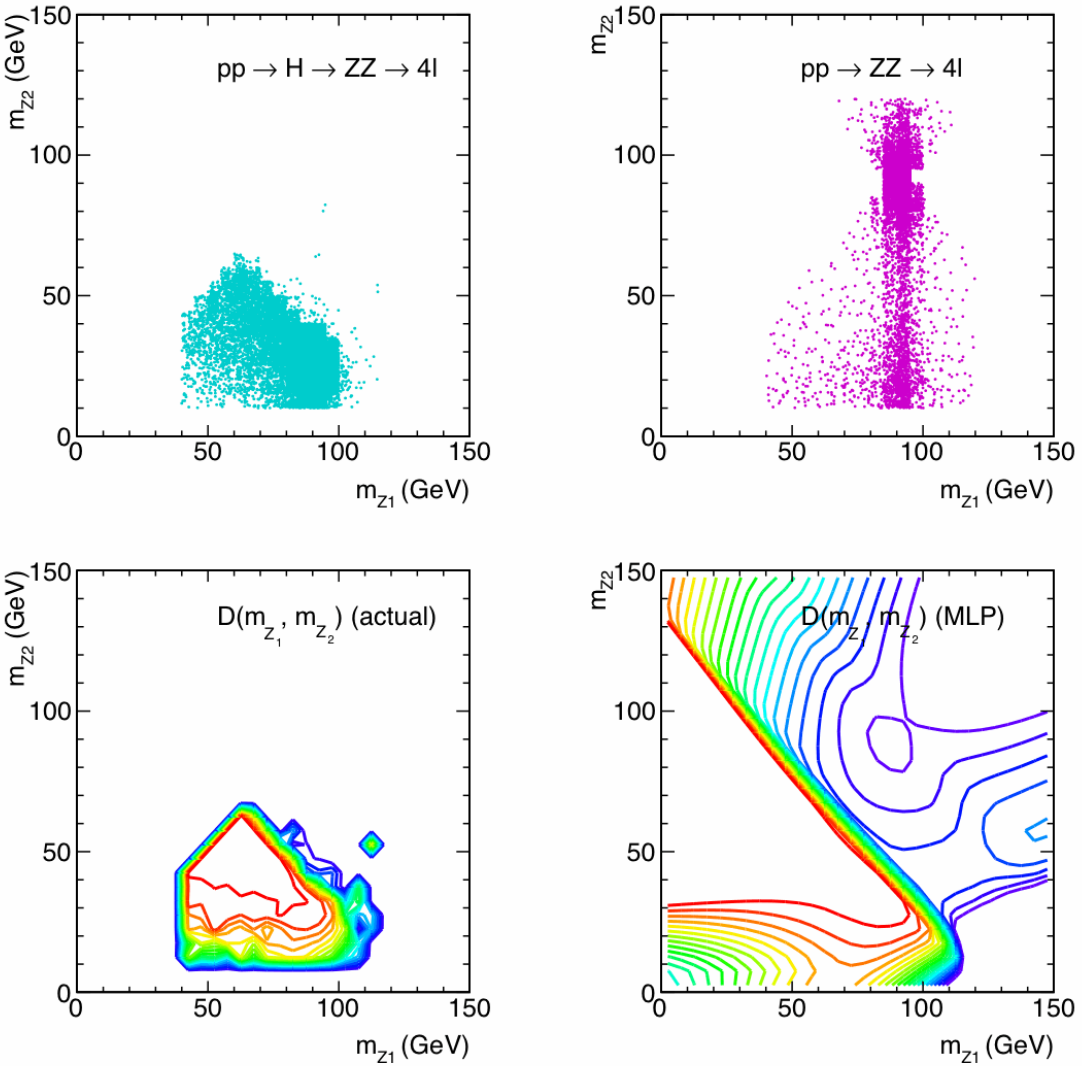


$$pp \to H \to ZZ \to \ell^+\ell^-\ell'^+\ell'^-$$

Background



$$pp \to ZZ \to \ell^+\ell^-\ell'^+\ell'^-$$



Classifier Performance

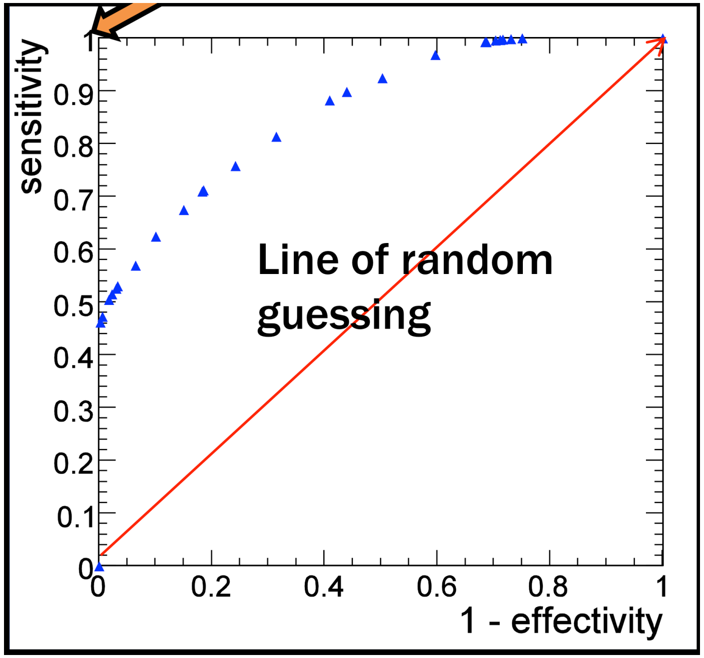
Classifier Performance

Receiver Operating Characteristic (ROC) curve

Commonly used metric

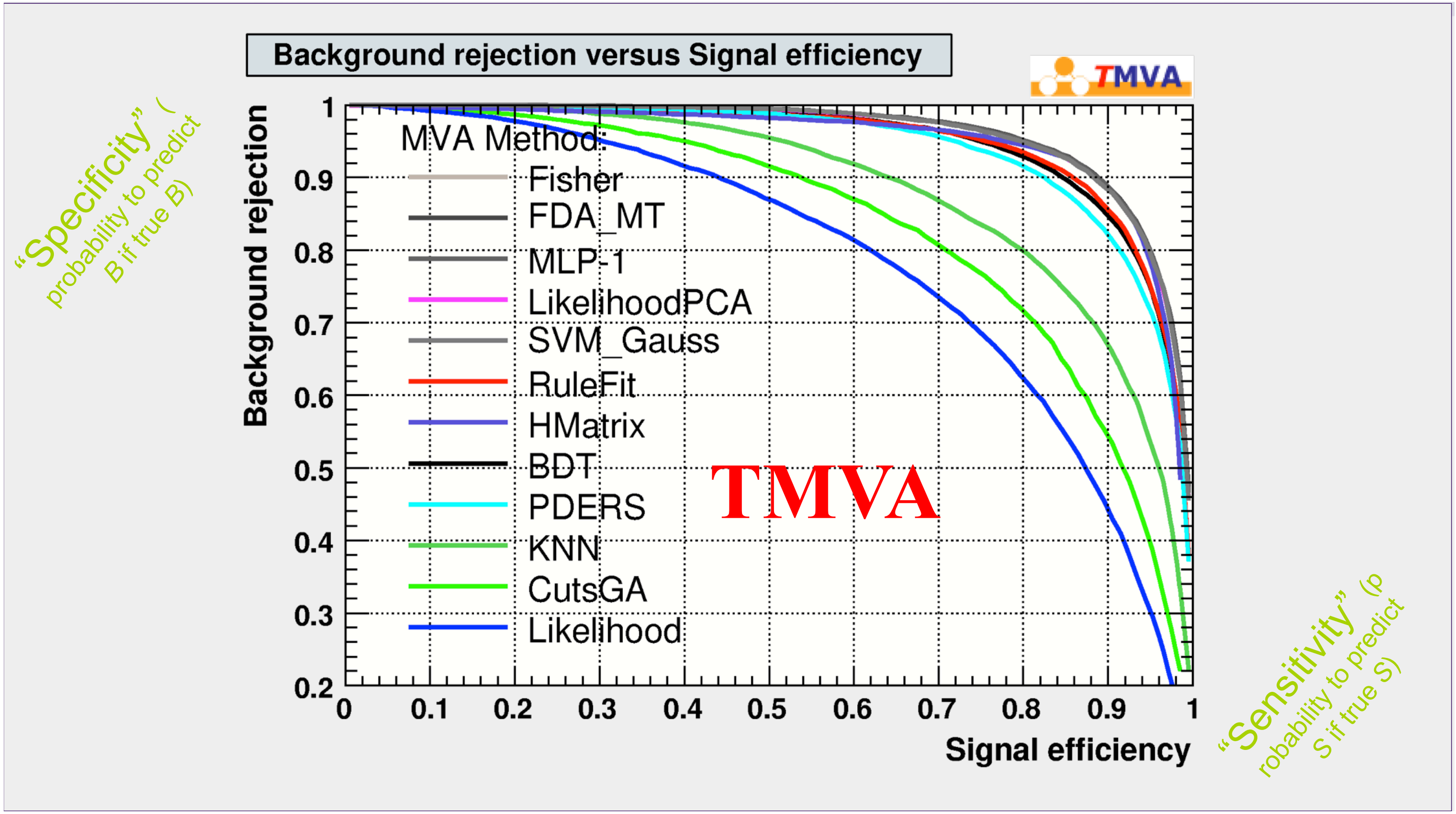
Shows the **relationship** between correctly classified positive cases (sensitivity) and incorrectly classified negative cases (1-effectivity)

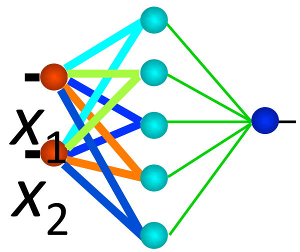
Perfect Classifier

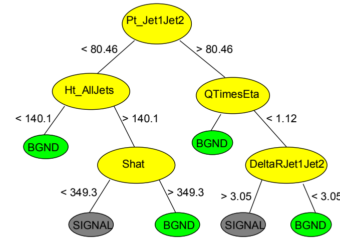


Classifier Performance

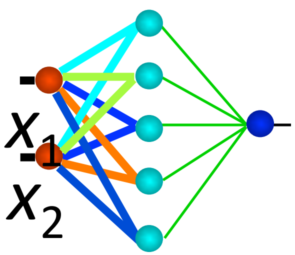
Receiver Operating Characteristic (ROC)

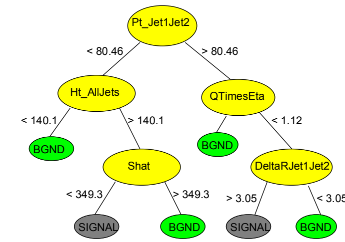


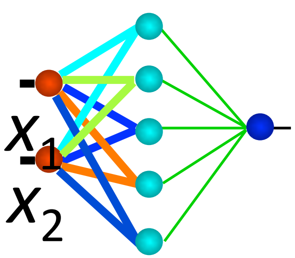


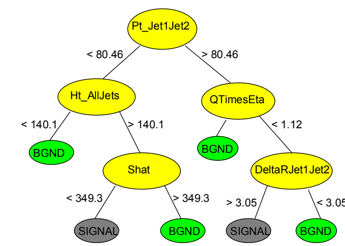


Ensemble Methods









Ensemble Methods

Suppose you have a collection of discriminants $f(x, w_k)$, which, individually, perform only marginally better than random guessing.

$$f(x) = a_0 + \sum_{k=1}^{K} a_k f(x, w_k)$$

From such discriminants, weak learners, it is possible to build highly effective ones by averaging over them:

Jerome Friedman & Bogdan Popescu (2008)

Ensemble Methods

Usually used with decision trees but they are more general. Most popular methods are:

Bagging

- Each tree trained on a **bootstrap sample** drawn from training set

Random Forest

- Bagging with **randomized trees**
  - Random subsets of features used at each split

Boosting

- Each tree trained on a **different weighting** of full training set

Adaptive Boosting

Adaptive Boosting

Train in stages

- Adaptive weights
  - ADABoost: Freund & Schapire 1997
- **Misclassified** events get a larger weight going into the next training stage
  - Classify with a majority vote from all trees
- **Works** very well to improve classification power of “greedy” decision trees
  - can be used with other classifiers

Adaptive Boosting

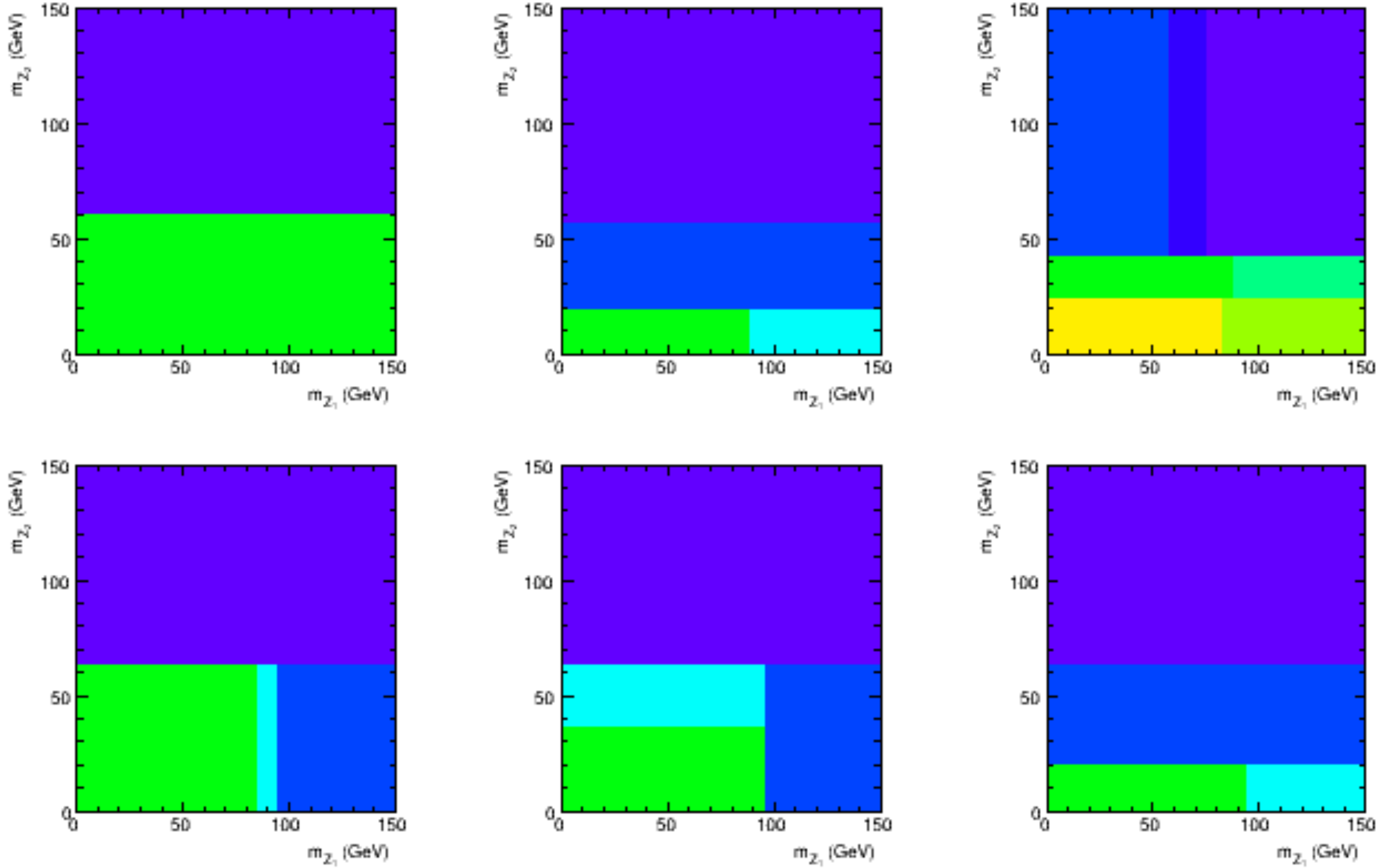
Repeat K times:

1. Create a decision tree $f(x, \mathbf{w})$
2. Compute its error rate ε on the ***weighted*** training set
3. Compute $\alpha = \ln(1 - \varepsilon) / \varepsilon$
4. Modify training set: ***increase weight*** of ***incorrectly classified examples*** relative to the weights of those that are correctly classified

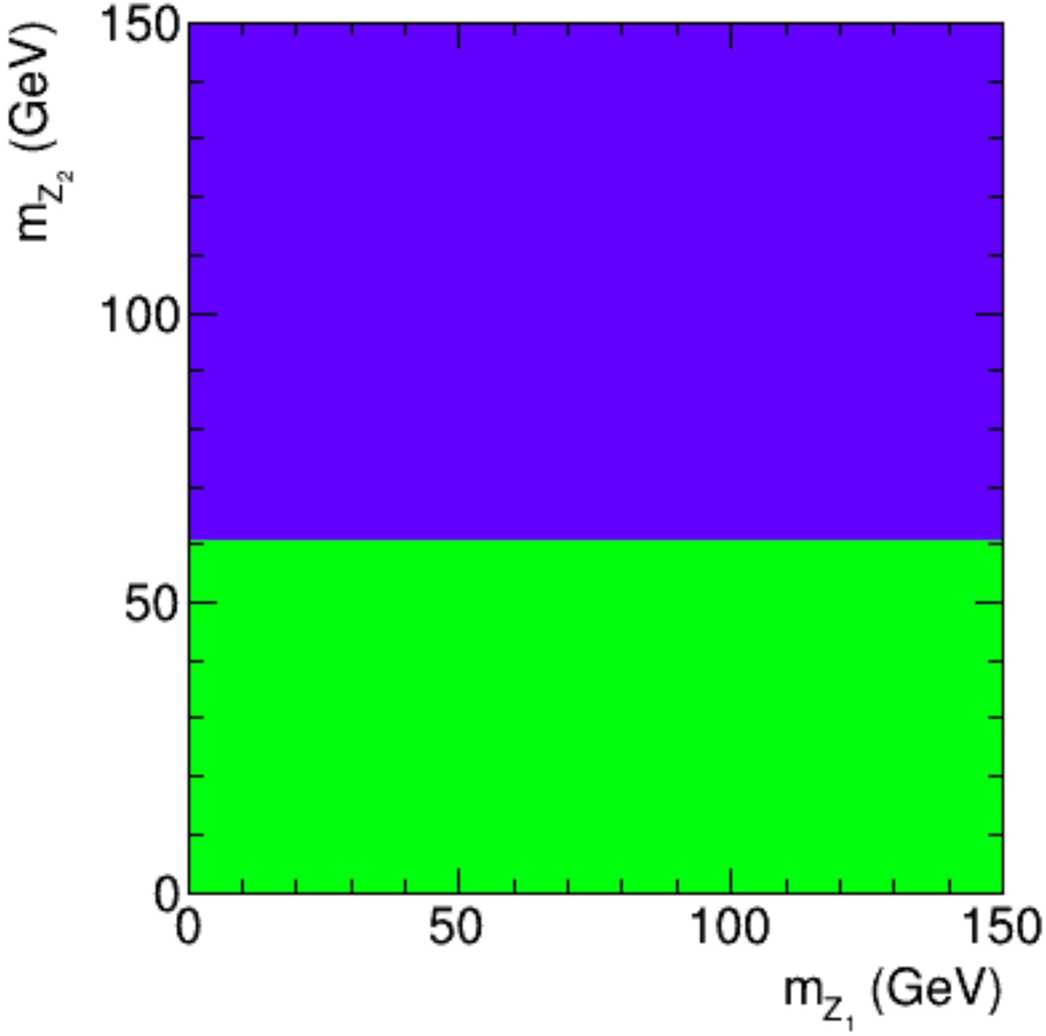
Then compute weighted average $f(x) = \sum \alpha_k \, f(x, w_k)$

Y. Freund and R.E. Schapire.
Journal of Computer and Sys. Sci. **55** (1), 119 (1997)

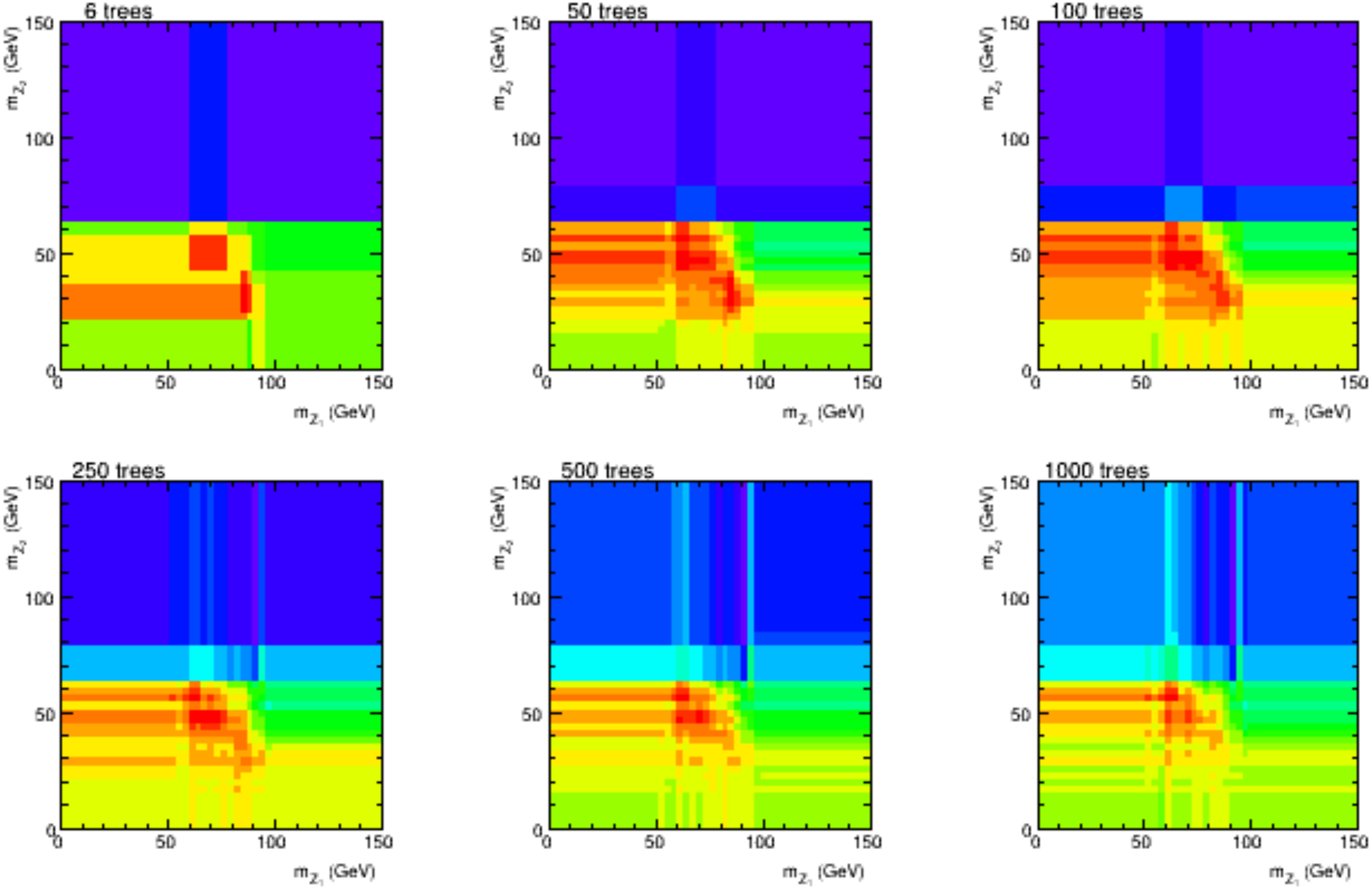
First 6 Decision Trees



First 100 Decision Trees

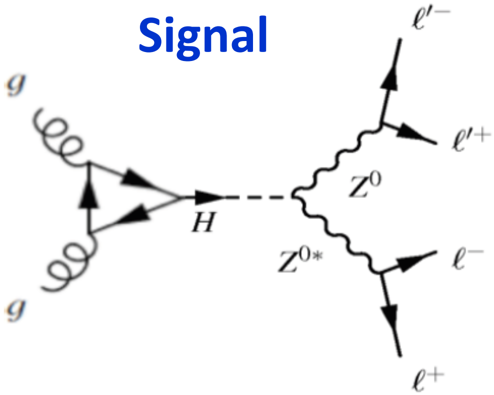


Averaging over a Forest



H to ZZ to 4Leptons

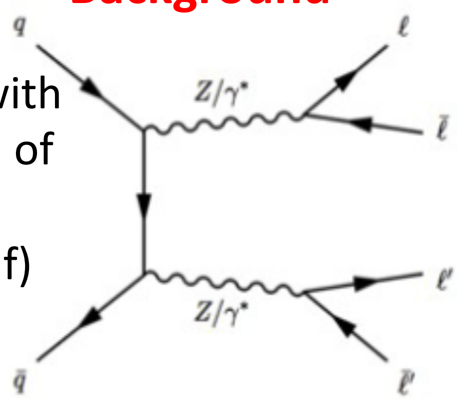
Signal



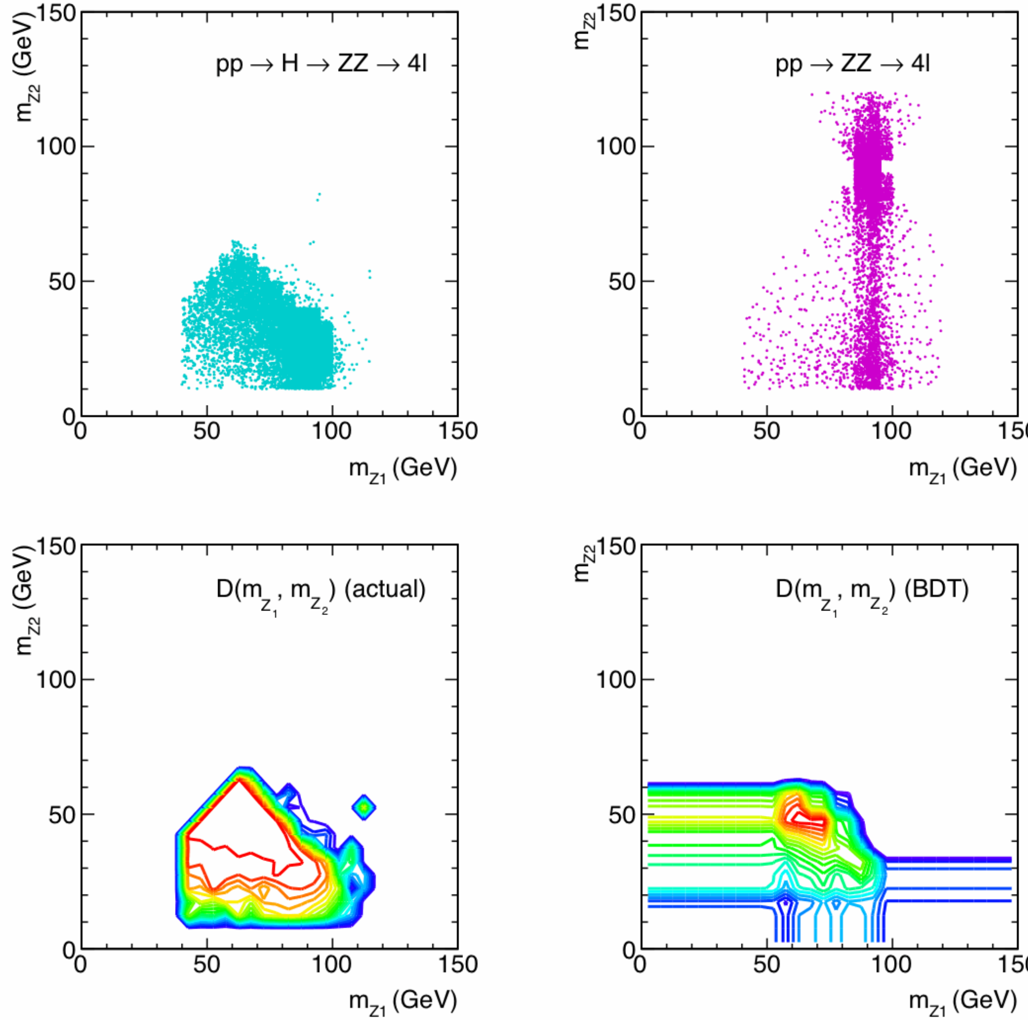
$$pp \to H \to ZZ \to \ell^+\ell^-\ell'^+\ell'^-$$

Background

200 trees with a minimum of 100 counts per bin (leaf)



$$pp \to ZZ \to \ell^+\ell^-\ell'^+\ell'^-$$



Function Estimation

Function Estimation

Comet Problem by Gauss (1805)

Approximate trajectory of a comet from observations

Approach: minimize difference between measurement and predictions in a systematic fashion

Vary regression model parameters

Function estimation

- Think of decision trees as **multidimensional histograms**
  - Bins are recursively constructed
  - Each associated to the value of f(x) to be approximated

- To go from classification to regression change the evaluation criteria used in the learning algorithm
  - from **maximum separation gain** to **minimal variance** from resulting subspace cuts

Regression Example

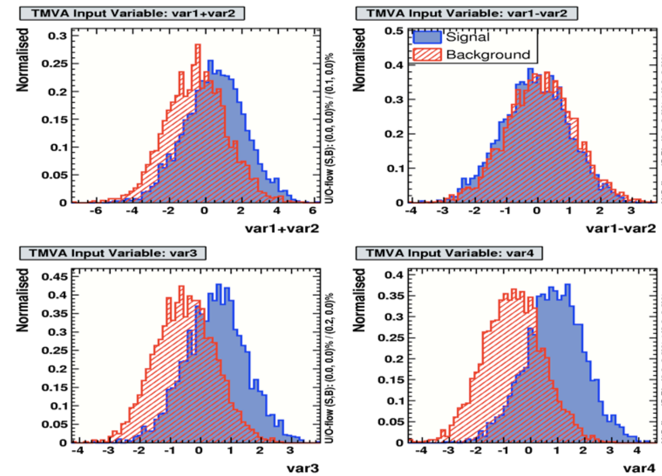
Improve calorimeter resolution by applying regression

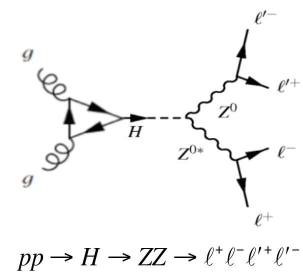
Inputs: electromagnetic shower information, other calorimetric variables

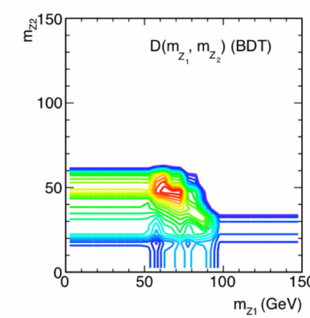
Target Output: calorimeter energy

Exercises

- Classification Exercises A,B
  - Simple Gaussians
- Classification Exercise C
  - Real HEP example H→ZZ→4l
- Regression Exercise D
  - Toy calorimeter regression







Optional exercises: Advanced classification
BNN regression

Summary

- **Machine learning** provides powerful multivariate methods for many classification/regression problems in HEP

- **Ensemble methods** are powerful extensions of these methods

- **Comprehensive tools** developed by HEP community widely used (TMVA, SPR, Paradigm)
  – try them ☺

Plenty of problems await to be TACKLED!

Proceed to Classification Tutorial B

Additional Material

Hilbert's 13th Problem

Problem 13: Prove the conjecture

In general, it is ***impossible*** to do the following:

$$f(x_1,\ldots,x_n) = F(\, g_1(x_1),\ldots,\, g_n(x_n)\,)$$

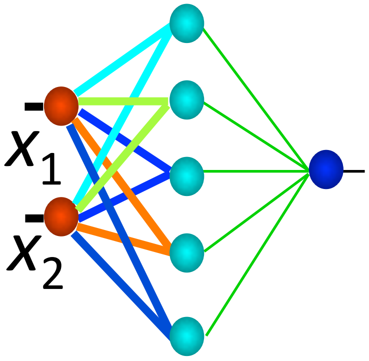
But, in 1957, Kolmogorov ***disproved*** Hilbert's conjecture!
Today, we know that functions of the form

$$f(x_1,\cdots,x_I) = a + \sum_{j=1}^{H} b_j \tanh\left[c_j + \sum_{i=1}^{I} d_{ji} x_i\right]$$

can provide arbitrarily accurate approximations.
(Hornik, Stinchcombe, and White,

Neural Networks **2**, 359-366 (1989))

Bayesian Neural Networks (BNN)



Bayesian Neural Networks

Given: $p(w \mid T) = p(T \mid w)\, p(w) / p(T)$

over the parameter space of the functions

$$n(\boldsymbol{x}, w) = 1 / [1+\exp(-f(\boldsymbol{x}, w))]$$

can estimate $p(s \mid \boldsymbol{x})$ as follows

$$p(s \mid \boldsymbol{x}) \sim n(\boldsymbol{x}) = \int n(\boldsymbol{x}, w)\, p(w \mid T)\, dw$$

$n(\boldsymbol{x})$ is called a **Bayesian Neural Network** (BNN)

Bayesian Neural Networks

Generate Sample

N points $\{w\}$ from $p(w \mid T)$ using a Markov chain Monte Carlo (MCMC) technique and

average over the last M points

$$n(\boldsymbol{x}) = \int n(\boldsymbol{x}, w)\, p(w \mid T)\, dw$$

$$\sim \sum n(\boldsymbol{x}, w_i) / M$$

H to ZZ to 4Leptons

Dots

$p(s \mid x) = H_s / (H_s + H_b)$

H_s, H_b, 1-D histograms

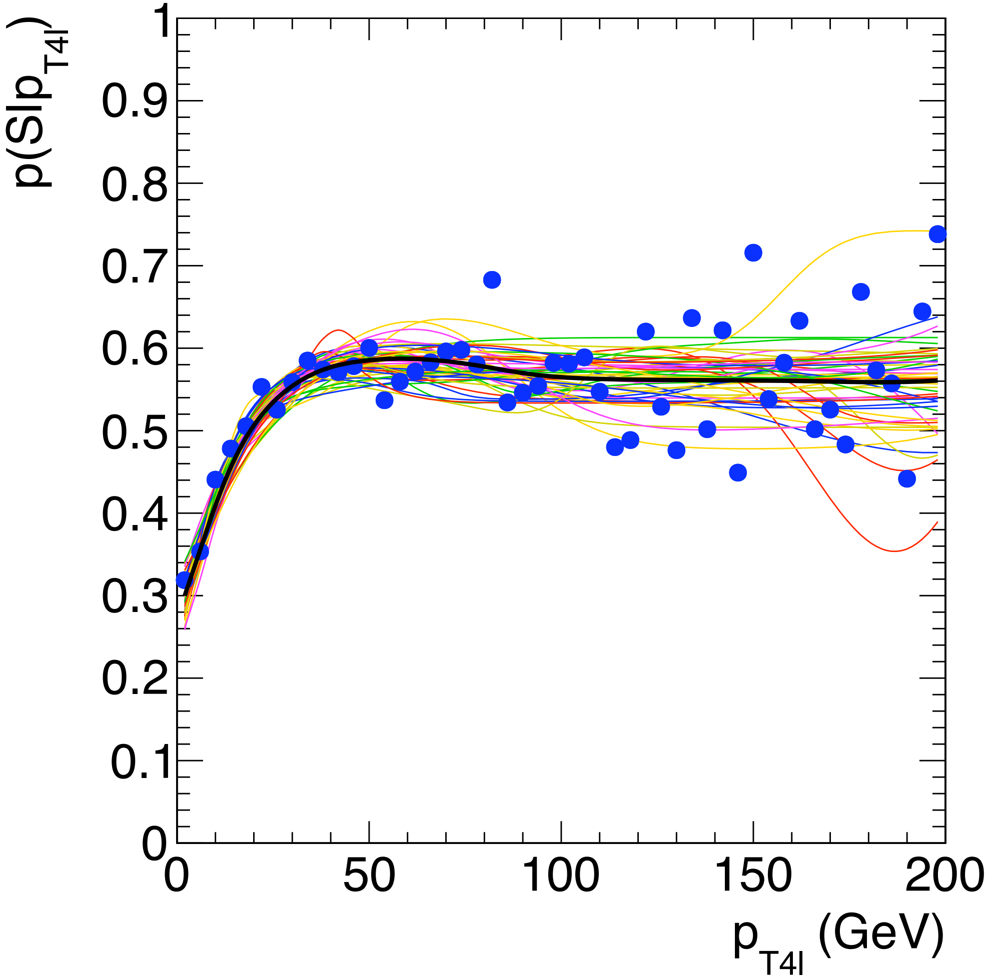
Curves

Individual NNs

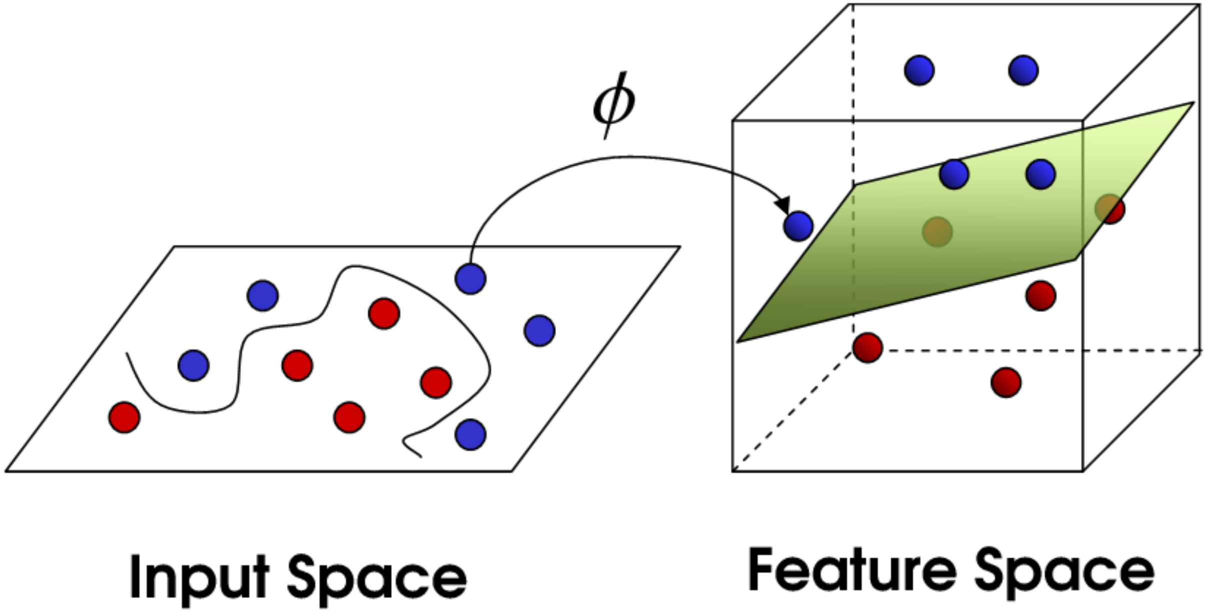
$n(x, w_k)$

Black curve

$n(x) = < n(x, w) >$



Support Vector Machines



Support Vector Machines

Generalization of the Fisher discriminant

– Boser, Guyon and Vapnik, 1992

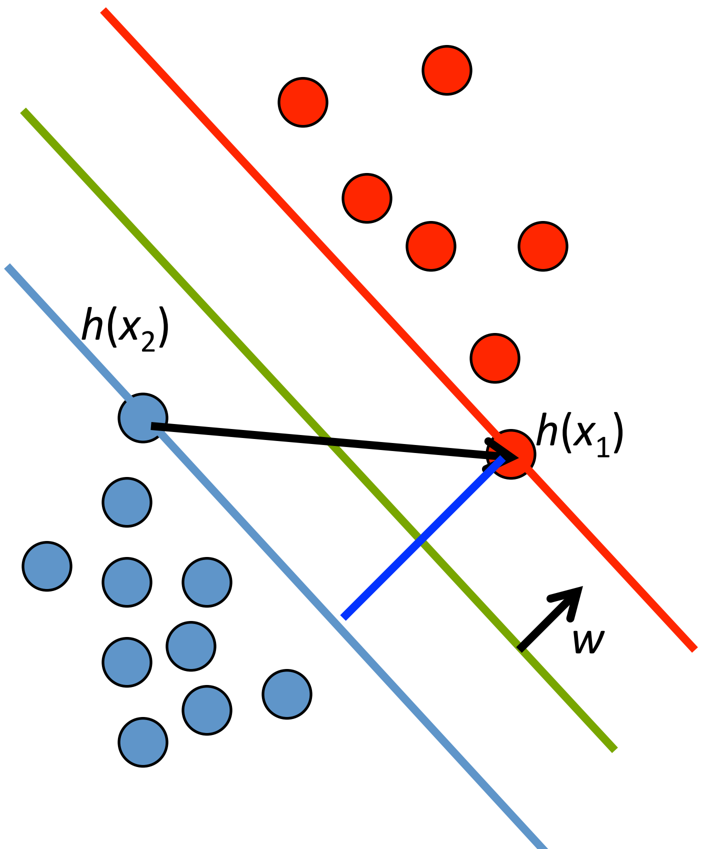
Basic Idea

Data that are non-separable in *d*-dimensions may be better separated if mapped into a space of higher (usually, infinite) dimension

$$h : \Re^d \rightarrow \Re^\infty$$

As in the Fisher discriminant, a hyper-plane is used to partition the high dimensional space $f(x) = w \cdot h(x) + c$

Support Vector Machines



Consider ***separable*** data in the high dimensional space

green plane: $w.h(x) + c = 0$

red plane: $w.h(x_1) + c = +1$

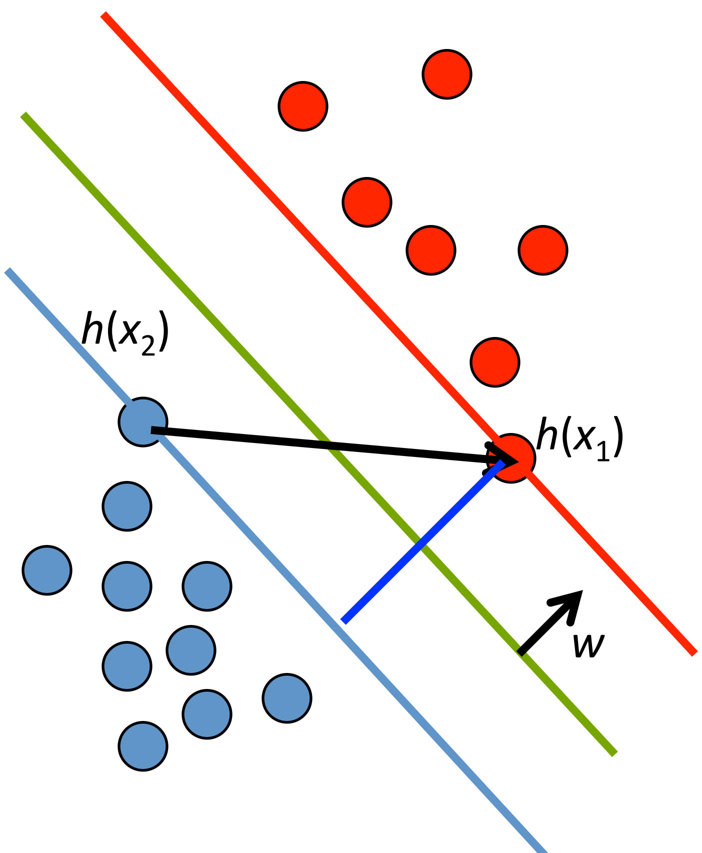
blue plane: $w.h(x_2) + c = -1$

subtract **blue** from **red**

$$w.[h(x_1) - h(x_2)] = 2$$

and normalize the high dimensional vector w $\hat{w}.[h(x_1) - h(x_2)] = 2/||w||$

Support Vector Machines



$m = \hat{w}.[h(x_1) - h(x_2)]$, the distance between the **red** and **blue** planes, is called the **margin**. The best separation occurs when the margin is as large as possible.

Note: because $m \sim 1/||w||$, maximizing the margin is equivalent to minimizing

$$||w||^2$$

Support Vector Machines

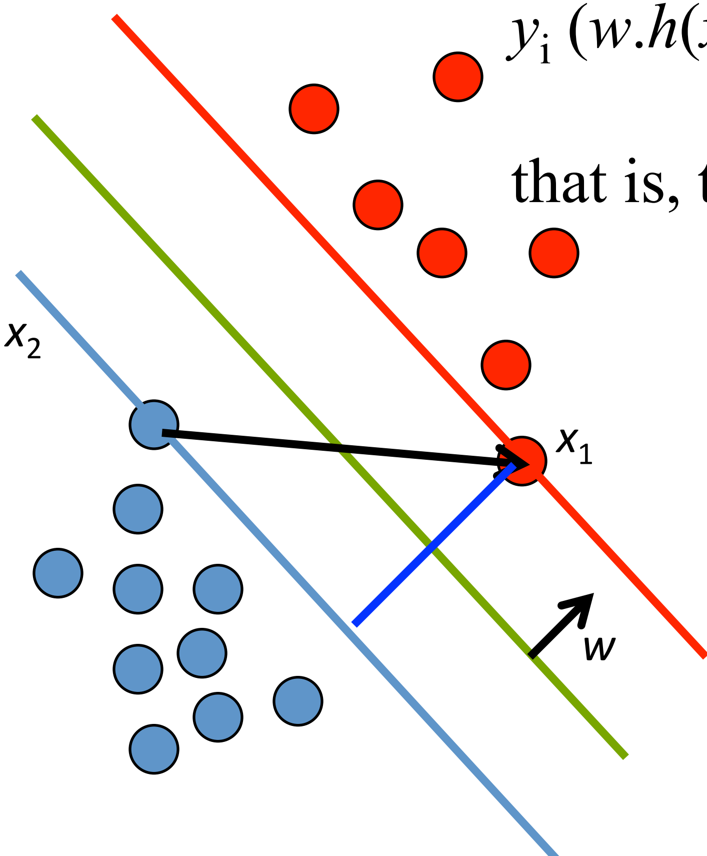
Label the **red** dots $y = +1$ and the **blue** dots $y = -1$. The task is to minimize $||w||^2$ subject to the constraint

$$y_i\,(w.h(x_i) + c) \geq 1, \qquad i = 1 \ldots N$$

that is, the task is to minimize

$$L(w, c, \alpha) = \tfrac{1}{2}\left\|w\right\|^2 - \sum_{i=1}^{N} \alpha_i \left[y_i \left(w \cdot h(x_i) + c \right) - 1 \right]$$

where the $\alpha > 0$ are Lagrange multipliers



Support Vector Machines

When ***L*(w,c,α)** is minimized with respect to *w* and *c*, the function ***L*(w,c,α)** can be transformed to

$$E(\alpha) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j h(x_i) \cdot h(x_j)$$

At the minimum of $E(\alpha)$, the only non-zero coefficients α are those corresponding to points *on* the **red** and **blue** planes: the so-called **support vectors**. The key idea is to replace the scalar product $h(x_i).h(x_j)$ between two vectors of infinitely many dimensions by a **kernel function** $K(x_i, x_j)$.

- The (unsolved) problem is how to choose the correct kernel for a given problem?

Genetic Algorithms

Central idea: adaptation. Inspired by evolutionary biology concepts of mutation, selection, cross-over (recombination) J.H. Holand, 1975

Begin with a large population of random solutions

- Evaluate each one
  - Fitness function (some form of S/√B)
  - Keep the best subset
    - Use it to build new solutions
    - Allow mutation, cross-over
    - Optimize over number of epochs/cycles

Used most frequently in HEP for rectangular cut optimization (computationally intensive)

Extensions

Classification

- Relatively easy to extend existing classifiers to handle more classes: just add more classes

Regression

- Very hard to do well
  - Nevertheless, very practical
- Less explored area in machine learning

Multi-Function Regression

For problems that require simultaneous estimation of N functions (that are possibly related)

- N single-function regression model solution is too cumbersome
- Also less accurate
- Correlations among functions may be important and need to be accounted for

Multi-function regression models are a better solution in this case

Multi-Objective Model

- Properly take into account **dependencies** between output attributes (their correlations)

- **improved performance results** compared to single-objective models, especially in ensembles

  – usually smaller and easier to interpret

  – very useful for transformations

Predictive Clustering

Example of a **multi-function regression** model based on trees or rules

- **Decision trees** are equated to clustering trees by P. Langley in 1996, first noted by Fisher in 1993
- **Cluster "hierarchy"**

  Each tree node corresponds to a cluster

  Root node contains full dataset partitioned recursively into sub-clusters

Clustering Concept

Use **decision tree induction** to obtain clusters with:

- **minimal intra-cluster distance**
  - between examples from the same cluster
- **maximal inter-cluster distance**
  - between examples from different clusters
  - In classification trees distance metric is class enthropy

Clustering Example

14 input variables {a, b, c, d…}

– 4 of them strongly correlated

14 target outputs to estimate {A, B, C, D…}

– 4 of them strongly correlated

Challenge: build a predictive model to describe simultaneously all the outputs {A,B,C,D…}, provided a corresponding set of inputs.

For example: These can be correlated EM shower-shapes

CLUS 2.0

Predictive clustering implementation

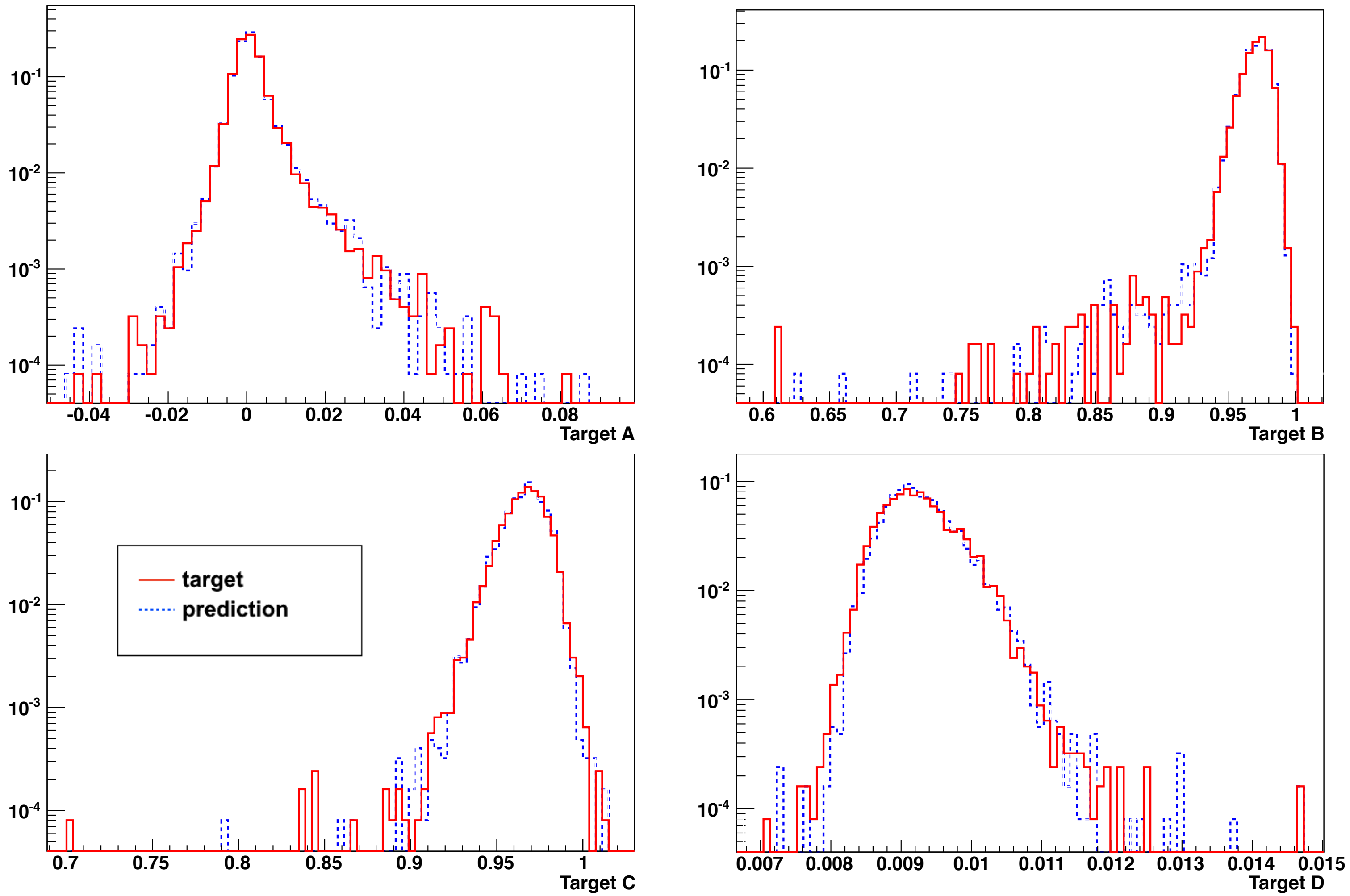
- **Decision tree** and rule induction system
- Designed for multi-task learning and multi-label classification
- **Well-suited** for both classification and regression problems

Learning Procedure

Train the predictive clustering model by providing a "map" between inputs and outputs. Let it learn.

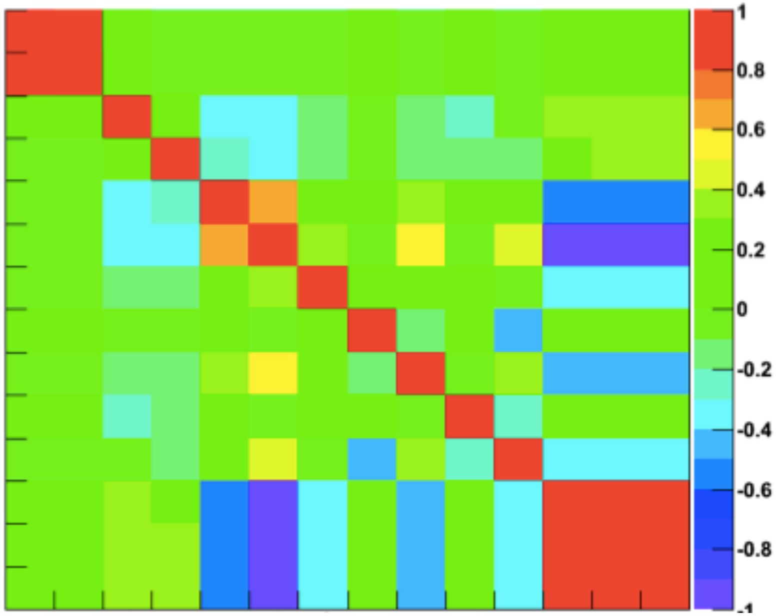
Evaluate: Use the Test set to compare predictions on "unseen" data to the Target values of the outputs.

Illustrative Example

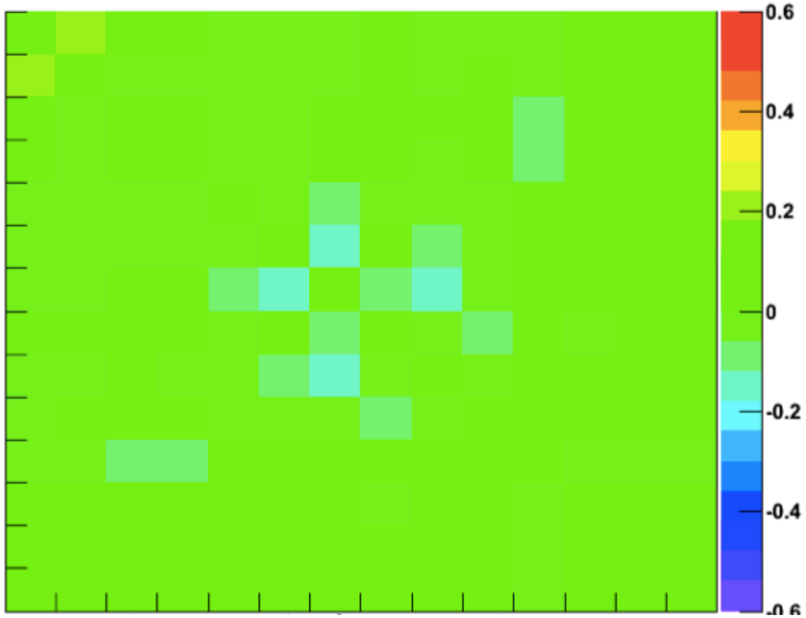


Correlations

Target Correlations



Prediction-Target Difference



Very close to Zero

Clustering Rules

Clustering rules can be constructed from predictive clustering trees

Main difference: simple rules focus on the accuracy connected to the target

Predictive clustering rules focus on:

- target attribute accuracy
- tight or compact rule coverage of the instances by computing their distance metric

Summary II

- **Many multivariate methods available: pick the one that best suits your problem**
  - Good starting points: random grid search, boosted decision trees, neural networks
  - Then: support vector machines, random forests , bayesian neural networks, predictive clustering

- **Both classification and regression can be generalized to multiple classes and targets**
  - Predictive clustering is a good example of both