Multivariate Analysis Methods in Particle Physics *

Article *in* Annual Review of Nuclear and Particle Science · November 2011

DOI: 10.1146/annurev.nucl.012809.104427

1 author:

Pushpa Bhat
Fermi National Accelerator Laboratory (Fermilab)
987 PUBLICATIONS **27,420** CITATIONS

SEE PROFILE

FERMILAB-PUB-10-054-E

Advanced Analysis Methods in Particle Physics

Pushpalatha C. Bhat

Fermi National Accelerator Laboratory, Batavia, IL 60510, USA

email: pushpa@fnal.gov

(Submitted to *Annual Review of Nuclear and Particle Science)*

"*That is positively the dopiest idea I have heard*." - Richard Feynman, when he signed on to work on the Connection Machine, at the *Thinking Machines Corporation,* in the summer of 1983.

Key Words

Multivariate methods, optimal analysis, neural networks, Bayesian inference, Tevatron, Large Hadron Collider (LHC)

Abstract

Each generation of high energy physics experiments is grander in scale than the previous – more powerful, more complex and more demanding in terms of data handling and analysis. The spectacular performance of the Tevatron and the beginning of operations of the Large Hadron Collider have placed us at the threshold of a new era in particle physics. The discovery of the Higgs boson or another agent of electroweak symmetry breaking and evidence of new physics may be just around the corner. The greatest challenge in these pursuits is to extract the extremely rare signals, if any, from huge backgrounds that arise from known physics processes. The use of advanced analysis techniques is crucial in achieving this goal. In this review, I discuss the concepts of optimal analysis, some important advanced analysis methods and a few examples. The judicious use of these advanced methods should enable new discoveries and produce results with better precision, robustness and clarity.

Contents

1 INTRODUCTION

The ambitious goal of understanding nature at the most fundamental level has led to the development of particle accelerators and detectors at successively grander scales. The revolutionary discoveries at the beginning of the twentieth century opened up the quantum world. By the middle of the century, the Standard Model (SM) of particle physics (1-6) was being built and by the turn of the century, the last quark (7, 8) and the last lepton (9**)** of the Standard Model had been found. Despite this spectacular success, a vital part of the Standard Model, the "Higgs mechanism" (10-13), still awaits experimental evidence. Moreover, there are indications that the SM particles and forces might be telling us only a part of the story. Since the SM accounts for only 4% of what makes up the universe, the rest must be explained in terms of matter and phenomena we have yet to uncover. The evidence for dark matter in the universe, the evidence for an accelerating universe, the discovery of neutrino oscillations, and the persistent discrepancies in some of the precision measurements in SM processes are some of the strong indicators of the existence of new physics beyond the SM. It appears that new physics is inevitable at the TeV energy scale (*Terascale*). We might be at the threshold of another extraordinary century in physics.

Since the discovery of the top quark in 1995 (7, 8, 14), the searches for the Higgs boson and for new physics have taken center-stage. The luminosity upgrades of the Fermilab Tevatron (15) in the past decade have produced unprecedented amounts of proton-antiproton collision data at the center of mass energy ($\sqrt{s}$) of 1.96 TeV. These data, in conjunction with the use of advanced analysis methods, have enabled observations of the electroweak production of single top quarks (16,17) and sensitive searches for the Higgs boson and for physics beyond the SM. The Large Hadron Collider (LHC) (18), with a design energy of $\sqrt{s}$ = 14 TeV, will open new energy frontiers that might help answer some of the most pressing particle physics questions of today.

The investments in the accelerator facilities and experiments – intellectual and monetary – and the total time span of the undertakings are so great that they cannot be easily replicated. Therefore, it is of the utmost importance to make the best use of the output of this investment – the data we collect. While the advances in computing technology have made it possible to handle vast amounts of data, it is crucial that the most sophisticated techniques be brought to

bear on the analysis of these data at all stages of the experiment. Over the past century, instrumentation has advanced from photographic detectors to those integrated with ultra-fast electronics that produce massive amounts of digital information every second. Likewise, data analysis has progressed from visual identification of particle production and decays to searches for bumps in invariant mass spectra of exclusive final state particles to event counting in inclusive data streams. The rates of interactions and the number of detector channels to be read out have grown by orders of magnitude over the course of the past few decades. We can no longer afford to write out data to storage media based on simple interaction criteria. However, the events that we seek to study are extremely rare, so data analysis in contemporary high energy physics (HEP) experiments begins when a high energy interaction or *event* occurs. The electronic data from the detectors must be transformed into useful physics information in real-time. The *trigger system* is expected to select interesting events for recording and discard uninteresting (background) events. Information from different detector systems is used to extract event features such as the number of tracks, high transverse momentum objects[1], and object identities. The extracted features are then used to decide whether the event should be recorded. At the LHC experiments, for example, the event rate will be reduced from 40 MHz to ~200 Hz for recording. The online processing of data is performed by a combination of hardware and software components.

More detailed analysis of the recorded data is performed offline. The common offline data analysis tasks are: charged particle tracking, energy and momentum measurements, particle identification, signal/background discrimination, fitting, measurement of parameters, and derivation of various correction and rate functions. The most challenging of the tasks is identifying events that are both rare and obscured by the wide variety of processes that can mimic the signal. This is a veritable case of "finding needles in a haystack" for which the conventional approach of selecting events by using cuts on individual kinematic variables can be far from optimal.

The power of computers coupled with important developments in *machine learning* algorithms, particularly the back-propagation algorithm for training neural networks (NN), brought a revolution in multivariate data analysis in the late 1980s. There was much skepticism about

[1] By objects, I mean electrons, muons, jets arising from quark or gluon fragmentation, etc.

these methods in the early 1990s when they were brought into HEP analyses (19-23). However, following several successful applications (24-29), particle physicists have largely accepted the use of NNs and other multivariate methods. It is now evident that without these powerful techniques, many of the important physics results that we have today would not have been achievable using the available datasets. In this review, my goal is to provide an introduction to the concepts that underlie these advanced analysis methods and describe a few popular methods. I also discuss some analysis examples and prospects for future applications.

2 OPTIMAL ANALYSIS CONCEPTS

"*Keep it simple, as simple as possible, not any simpler*" Albert Einstein

The goal in data analysis is to extract the best possible results. Here I discuss the types of analysis tasks we perform, explain why the sophistication of multivariate methods is necessary to obtain optimal results, and introduce the concepts and the general framework that underlie the popular methods.

The broad categories of analysis tasks are: (a) classification (b) parameter estimation and (c) function fitting. Classification is the process of assigning objects or events to one of the possible discrete classes. Parameter estimation is the extraction of one or more parameters by fitting a model to data. By function fitting I mean the derivation of continuous functions of variables. Mathematically, in all these cases, the underlying task is that of functional approximation.

Classification of objects or events is, by far, the most important analysis task in HEP. Common examples are the identification of electrons, photons, τ-leptons, *b*-quark jets, and so on, and the discrimination of signal events from those arising from background processes. Optimal discrimination between classes is crucial to obtain signal-enhanced samples for precision physics measurements. Measurements of track parameters, vertices, physical parameters such as production cross sections, branching ratios, masses and other properties are examples of parameter estimation (or regression). Some examples of function fitting are the derivation of correction functions, tag rate functions and fake rate functions.

These categories of tasks are also referred to as *pattern recognition* problems.[2]

2.1 Multivariate Treatment of Data

Data characterizing an object or an event generally involve multiple quantities referred to as *feature variables.* These may be, for example, the four-vectors of particles, energy deposited in calorimeter cells, deduced kinematic quantities, and global event characteristics. The variables, generally, are also correlated. To extract results with maximum precision it is necessary to treat these variables in a fully multivariate way.

The feature variables that describe an object or an event can be represented by a vector $\boldsymbol{x} = (x_1, x_2, ..., x_d)$ in a *d*-dimensional *feature space.* The objects or events of a particular type or class can be expected to occupy specific contiguous regions in the feature space. When correlations exist between variables, the *effective dimensionality* of the problem is smaller than *d.*

Pre-processing of data is the first step in an analysis. This is also referred to as *feature extraction* or *variable selection.* Having selected a set of variables, one may apply a transformation to the variables to yield a representation of the data that exhibits certain desirable properties. The pre-processing could be a simple scaling of the variables or a sophisticated transformation such as decorrelation of variables or combining them to construct physics-motivated variables. In some applications, this pre-processing may be the only necessary multivariate treatment of the data. In others, it serves as the starting point for a more refined analysis. Given $\boldsymbol{x}$, the goal is to construct a function $y = f(\boldsymbol{x})$ with properties that are useful for subsequent decision-making and inference. That is, we seek to extract a map $f : \Re^d \rightarrow \Re^N$, preferably with $N << d$. ($\Re^m$: real vector space of dimension *m*.) In practice, we try to approximate the desired function with $\tilde{y} = f(\boldsymbol{x}, \boldsymbol{w})$, where $\boldsymbol{w}$ are some adjustable parameters. I discuss the general approach for obtaining the functional approximation in the following sections.

The power of multivariate analysis is illustrated by a simple two-dimensional example. **Figure 1(a), (b)** show the distributions of two observables *x*1 and *x*2 that arise from two bivariate

[2] Pattern recognition also encompasses *knowledge discovery* by data exploration which deals with data-driven extraction of features, and deriving empirical rules via data-mining.

Gaussian distributions (**Figure 1(c)**). The one-dimensional projections (**Figure 1(d,e)**), namely the marginal densities $f(x1) = \int G(x1, x2)dx2$ and $f(x2) = \int G(x1, x2)dx1$ overlap considerably and there are no obvious cuts on $x1$ and $x2$ that would separate the two classes. However, when we examine the data in two dimensions, we see that the two classes are largely separable. Therefore, a cut applied to the linear function (30), $\tilde{y} = ax1 + bx2$, called a linear discriminant, shown in **Figure 1(f))**, can provide *optimal discrimination* of the two classes. The linear function separating the two classes shown in **Figure 1(c)** is a simple example of a decision boundary. Optimal discrimination, most simply, is a procedure that minimizes the probability of misclassification.

2.2 Machine Learning

The availability of vast amounts of data and challenging scientific and industrial problems characterized by multiple variables paved the way to the development of automated algorithms for *learning from data.* The primary goal of learning is to be able to respond correctly to future data. In conventional statistical techniques, one starts with a mathematical model and finds parameters of the model either analytically or numerically using some optimization criteria. This model then provides predictions for future data. In machine learning, an approximating function is inferred automatically from the given data without requiring a priori information about the function.

Machine Learning:

Machine Learning is the paradigm for automated *learning from data* using computer algorithms. It has origins in the pursuit of Artificial Intelligence, particularly, in Frank Rosenblatt's creation of the Perceptron around 1960 (31).

In machine learning, the most powerful approach to obtain the approximation $f(\boldsymbol{x}, \boldsymbol{w})$, of the unknown function $f(\boldsymbol{x})$, is *supervised learning*, in which a training data set, comprising feature vectors (inputs)[3] and the corresponding targets (or desired outputs), is used. The training data set $\{y, \boldsymbol{x}\}$, where y are the targets (from the true function $f(\boldsymbol{x})$), encodes information about the input-output relationship to be learned. In HEP, the

[3] I use feature vectors and inputs, interchangeably.

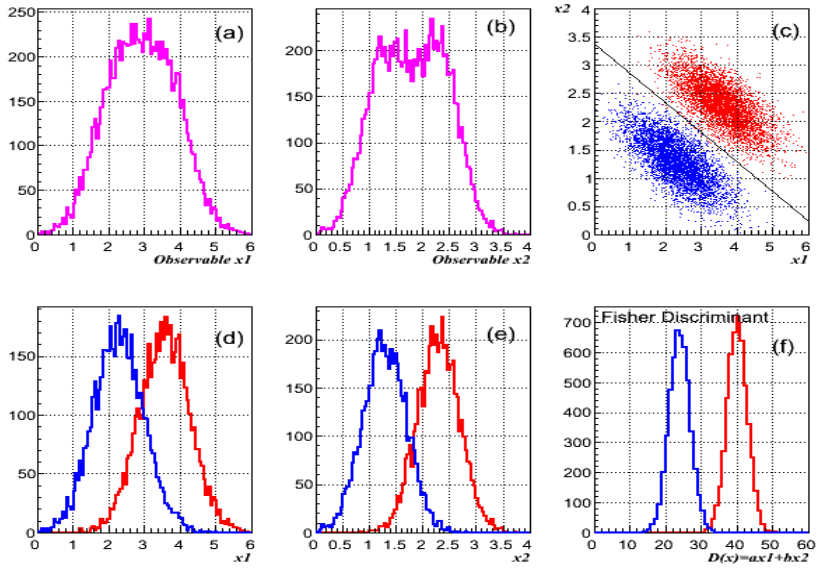


Figure 1

(a,b) Distributions of two hypothetical observables x1 and x2 arising from a mixture of two classes with bivariate Gaussian densities; (c) bivariate densities of the two classes (d,e) 1D marginalized densities and (f) a linear discriminant function f(x1,x2) that reveals two distinct distributions. An optimal cut placed on the discriminant results in the linear decision boundary shown in (c).

training data set generally comes from Monte Carlo simulations. The function $f(\boldsymbol{x})$ is discrete for classification ({0,1} or {-1,1} for binary classification) and is continuous for regression. (Therefore, the distinction between discrimination and regression is not fundamental.) The goal of learning (or training) is to find the parameters $\boldsymbol{w}$ of our model, that is, a functional approximation for the desired input-output map.

In all approaches to functional approximation (or function fitting), the information loss incurred in the process has to be minimized. The information loss is quantified by a loss function $L(y, f(\boldsymbol{x}, \boldsymbol{w}))$. In practice, the minimization is more robust if one minimizes the loss function averaged over the training data set. A learning algorithm, therefore, directly or indirectly, minimizes the average loss, called the *risk*, quantified by a risk function $R(\boldsymbol{w})$ that measures the

cost of mistakes made in the predictions, and finds the best parameters $\boldsymbol{w}$. The *empirical risk* (an approximation to true risk) is defined as the average loss over all (*N*) predictions,

$$R(\boldsymbol{w}) = \frac{1}{N}\sum_{i=1}^{N} L\{y_i, f(\boldsymbol{x_i}, \boldsymbol{w})\}. \quad 1.$$

A commonly used risk function is the mean square error,

$$R(\boldsymbol{w}) = E(\boldsymbol{w}) = \frac{1}{N}\sum_{i=1}^{N} (y_i - f(\boldsymbol{x_i}, \boldsymbol{w}))^2. \quad 2.$$

If the optimization has to take into account any constraint $Q(\boldsymbol{w})$, it can be added to the risk function to give the *cost function* to be minimized,

$$C(\boldsymbol{w}) = R(\boldsymbol{w}) + \lambda Q(\boldsymbol{w}), \quad 3.$$

where λ is an adjustable parameter that determines the strength of the constraint imposed. The cost function in the case of a mean square error is the well known constrained χ^2 fit. The function $f(\boldsymbol{x}, \boldsymbol{w})$ obtained by the procedure converges, in the limit of a large training data set, to the function $f(\boldsymbol{x})$ that minimizes the true risk function.

The risk minimization can be performed using many algorithms. Each attempts to find the global minimum of the cost function in the parameter space. In practice, however, it is usually only possible to find a local minimum. The generic method is that of gradient descent. Other popular methods include Levenberg-Marquardt (32), simulated annealing (33) and genetic algorithms (GAs) (34). The constraint in the cost function is typically used to control model complexity (i.e., *over-fitting*), and is known as regularization. The performance of the classifier or estimator is generally evaluated using a test data set independent of the training set.

A method that can approximate a continuous nonlinear function to arbitrary accuracy is called a *universal approximator*. Neural networks are examples of universal approximators.

Two other important approaches to learning are unsupervised and reinforcement learning. In the former approach, no targets are provided and the algorithm finds associations among the input

vectors. In the latter, correct outputs are rewarded and incorrect ones are penalized. These methods are not further discussed in this review.

2.3 The Bayesian Framework

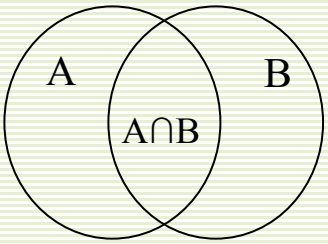
"Today's posterior distribution is tomorrow's prior." – David Lindley

The Bayesian approach to statistical analysis is that of inductive inference. It allows the use of prior knowledge and new data to update probabilities. Therefore, it is a natural paradigm for learning from data. It is an intuitive and rigorous framework for handling classification and parameter estimation problems. At the heart of Bayesian inference (35) is Bayes theorem,

$$p(B \mid A) = \frac{p(A \mid B)\, p(B)}{p(A)}, \qquad 4.$$

where the conditional probabilities $p(B \mid A)$ and $p(A \mid B)$ are referred to as the *posterior probability* and *likelihood*, respectively, $p(B)$ is the *prior probability* of *B*, and the denominator is simply the total probability of *A*, $p(A) = \int p(A \mid B)\, p(B) dB$. If *B* is discrete, then the integral is replaced by a sum.

Conditional Probabilities:



$$p(A \mid B) = \frac{p(A \cap B)}{p(B)}$$

$$p(B \mid A) = \frac{p(A \cap B)}{p(A)}$$

Bayes theorem follows immediately from these expressions.

p(A|B): Probability of A, given B

p(B|A): Probability of B, given A

Let us consider a binary classification problem in which an event must be classified either as due to a signal process s, or as due to a background process b. This is achieved by placing a cut on the ratio of the probabilities for the two classes,

$$r(\boldsymbol{x}) = \frac{p(s \mid \boldsymbol{x})}{p(b \mid \boldsymbol{x})} = \frac{p(\boldsymbol{x} \mid s)\, p(s)}{p(\boldsymbol{x} \mid b)\, p(b)}, \qquad 5.$$

where $p(\boldsymbol{x} \mid s)$ and $p(\boldsymbol{x} \mid b)$ are the likelihoods of the data for signal and background classes, respectively, and $p(s)$ and $p(b)$ are the prior probabilities. The discriminant *r* is known as the Bayes discriminant, where $r(\boldsymbol{x})$=*constant* defines a decision

boundary in the feature space. Bayes rule is to assign a feature vector to the signal class if $p(s \mid \boldsymbol{x}) > p(b \mid \boldsymbol{x})$. This rule minimizes the probability of misclassification. Any classifier that minimizes the misclassification rate is said to have reached the *Bayes limit*. The problem of discrimination, then, mathematically reduces to that of calculating the Bayes discriminant $r(\boldsymbol{x})$ or any one-to-one function thereof.

The posterior probability for the desired class *s*, becomes

$$p(s \mid \boldsymbol{x}) = \frac{p(\boldsymbol{x} \mid s)\,p(s)}{p(\boldsymbol{x} \mid s)\,p(s) + p(\boldsymbol{x} \mid b)\,p(b)} = \frac{r}{1+r}. \qquad 6.$$

There are parametric and non-parametric methods to estimate $p(\boldsymbol{x} \mid s)$ and $p(\boldsymbol{x} \mid b)$ which I discuss in the next section. If one minimizes the mean square error function (Equation 2) where the targets are {0,1}, then $f(\boldsymbol{x}, \boldsymbol{w})$, if flexible enough, will directly approximate the posterior probability, $p(s \mid \boldsymbol{x})$. NNs, being universal approximators, are one such class of functions.

Neyman-Pearson Lemma:

When the hypotheses are fully specified, the Bayes Rule of assigning a feature vector to the most probable class is identical to the Neyman-Pearson criterion (36) of comparing the likelihood ratio of the two hypotheses to a threshold value k_α *and accepting hypothesis* H_0 *if*

$$\frac{L(x \mid H_0)}{L(x \mid H_1)} > k_\alpha$$

where α *defines the desired significance level.*

Because $p(s)$ and $p(b)$ are not always known, one can calculate the discriminant function

$$D(\boldsymbol{x}) = \frac{s(\boldsymbol{x})}{s(\boldsymbol{x}) + b(\boldsymbol{x})}, \qquad 7.$$

where $s(\boldsymbol{x}) = p(\boldsymbol{x} \mid s)$ and $b(\boldsymbol{x}) = p(\boldsymbol{x} \mid b)$. The posterior probability for the signal class is related to this discriminant function by

$$p(s \mid \boldsymbol{x}) = \frac{D(\boldsymbol{x})}{[D(\boldsymbol{x}) + (1 - D(\boldsymbol{x}))/k]}, \qquad 8.$$

where $k = p(s)/p(b)$. The discriminant $D(\boldsymbol{x})$ is often referred to (misleadingly) as the likelihood discriminant in HEP. The discriminating power of $D(\boldsymbol{x})$, which is a one-to-one function of $p(s \mid \boldsymbol{x})$, is the same as that of $p(s \mid \boldsymbol{x})$.

When many classes $C_k\,(k = 1,2,...,N)$ are present, the Bayes posterior probability can be written as,

$$p(C_k \mid \boldsymbol{x}) = \frac{p(\boldsymbol{x} \mid C_k)\,p(C_k)}{\sum p(\boldsymbol{x} \mid C_k)\,p(C_k)}. \qquad 9.$$

The Bayes rule for classification is to assign the object to the class with highest posterior probability. This is also the criterion in hypothesis testing.

In problems of parameter estimation, the posterior probability for a model parameter θ is,

$$p(\theta \mid \boldsymbol{x}) = \frac{p(\boldsymbol{x} \mid \theta)\,p(\theta)}{p(\boldsymbol{x})}, \qquad 10.$$

where $p(\theta)$ is the prior probability of θ. Thus in the Bayesian approach, one has a probability distribution of possible values for the parameter θ, whereas in conventional machine learning methods one calculates a maximum likelihood estimate for θ. However, the two approaches are closely related. The minimization of the error or cost function in the machine learning approach is equivalent to maximizing the Bayesian posterior probability.

3 POPULAR METHODS

In this section, I discuss, with minimal mathematics, several methods that are particularly relevant for and popular in HEP – from the simplest to the most sophisticated multivariate methods. The interested reader may consult many excellent books for details about these methods and algorithms (37-41).

3.1 Grid Searches

The conventional approach to separating signal from background is to apply a set of cuts such as $x_1 > z_1, x_2 > z_2 ...$ where $(z_1, z_2 ... z_d)$ forms a cut-point in the *d*-dimensional feature space. (This is sometimes referred to as "cut-based" method in HEP.) These "rectangular" cuts are usually arrived at by a process of trial and error informed by common sense and physics insight. Unfortunately, there is no guarantee that this procedure will lead to optimal cuts (as illustrated by

the example in Section 2). One can obtain the best set of rectangular cuts by performing a systematic search over a grid in feature space. A search performed over a regular grid, however, is inefficient. Much time can be spent scanning regions of feature space that have few signal or background points. Moreover, the number of grid points grows as M^d, which increases rapidly with bin count M and dimensionality d – a problem known as the "curse of dimensionality". A better way is to use a random grid search (RGS) (42), in which a distribution of points that form a random grid is used as the set of cut-points. The cut-points could be obtained, for example, from signal events generated by a Monte Carlo simulation. This is illustrated in **Figure 2** with an example in a two-dimensional feature space. The results can be plotted as the efficiency for retaining signal versus the efficiency for background for each cut, as shown in **Figure 2(b)**. The optimal cuts are those that maximize signal efficiency for desired background efficiency.[4] The methods, discussed later, provide optimal cuts that would be at least as good, and in fact, most often superior to the best cuts from grid searches. Comparison of the RGS results with those from a neural network in **Figure 2(b)** show that the neural network cuts are better in general and significantly better when large background rejection is desired.

The random grid search can be used for (a) a rapid search for the best rectangular cuts, (b) to compare the efficacy of variables or (c) to serve as a benchmark for more sophisticated multivariate analyses.

3.2 Linear Methods

In grid searches, the decision boundaries are lines or planes parallel to the axes of the feature space. As illustrated in **Figure 1**, optimal separation of classes may require decision boundaries rotated relative to the axes of the original feature space.

In a linear model, the mapping can be written as,

$$\tilde{y}(\boldsymbol{x}) = w_0 + w_1 x_1 + w_2 x_2 + ... = \sum_i w_i x_i = \boldsymbol{w}\boldsymbol{x}, \qquad 11.$$

where $\boldsymbol{w}$ is the vector of weights. (I use weights and parameters interchangeably.)

[4] The plot is akin to the ROC (Receiver Operating Characteristic) curve, which was invented in the 1950s to study radio signals in the presence of noise and is used in signal detection theory.

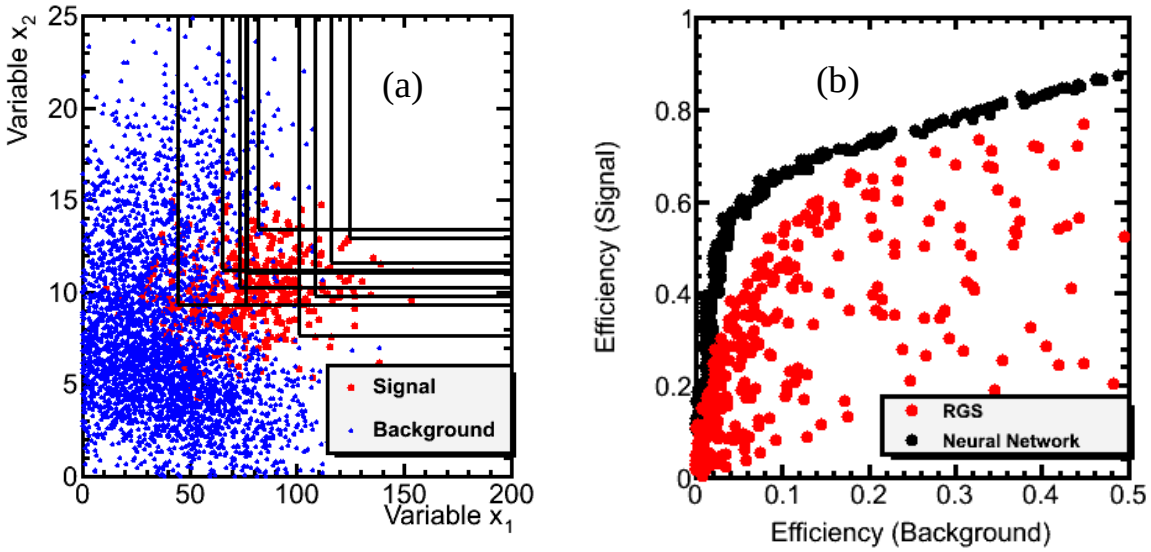


Figure 2

Random grid search (RGS) algorithm for finding best cuts in feature space and comparison with neural network results. (a) Simulated signal and background 2D distributions and example cuts (lines parallel to axes) using the signal sample. (b) Efficiency of RGS cuts to select signal events plotted against efficiency for background events (arbitrarily scaled). Each red point corresponds to an RGS cut. Note that most of the RGS cuts are sub-optimal and those at the upper edge of the distribution of red points provide the best set of "rectangular" cuts in the sense of maximizing signal efficiency for a given background. The results of a neural network (black points) trained on the same data are clearly superior to RGS cuts. Particularly, in the region of interest of low background efficiency, the gain in signal efficiency from neural networks is significant.

Fisher (30) pioneered the earliest successful applications of linear discriminants. Fisher's approach to discrimination between classes was to find a linear combination of input variables that maximizes the ratio of the distance between the class means to the sum of the class variances along the direction of $\boldsymbol{w}$. If we consider two sets of feature vectors $\boldsymbol{x_s}$ and $\boldsymbol{x_b}$ from the signal and background classes, with means and variances μ_s and σ_s, and μ_b and σ_b, respectively, the Fisher criterion is to maximize

$$F(\boldsymbol{w}) = \frac{(\mu_s - \mu_b)^2}{\sigma_s^2 + \sigma_b^2}, \qquad 12.$$

which for the parameters $\boldsymbol{w}$, yields

$$\boldsymbol{w} = \boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_s - \boldsymbol{\mu}_b), \qquad 13.$$

where $\boldsymbol{\Sigma}$ is the common covariance matrix for the classes. The Fisher discriminant can also be derived from Bayes discriminant starting with a Gaussian density for each class,

$$f(\boldsymbol{x}) = \frac{1}{(2\pi)^{d/2}|\boldsymbol{\Sigma}|^{1/2}} \exp\left[-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right]. \qquad 14.$$

Taking the logarithm of the Bayes discriminant (Equation 5), we obtain,

$$\log\frac{p(s \mid \boldsymbol{x})}{p(b \mid \boldsymbol{x})} = \frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_b)^T \boldsymbol{\Sigma}_b^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_b) - \frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_s)^T \boldsymbol{\Sigma}_s^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_s) + \frac{1}{2}\log\frac{|\boldsymbol{\Sigma}_s^{-1}|}{|\boldsymbol{\Sigma}_b^{-1}|} + \log\frac{p(s)}{p(b)}. \qquad 15.$$

This is the general form of the Gaussian classifier. After omitting non-essential terms that are independent of $\boldsymbol{x}$, the Gaussian classifier can be written as

$$F = D(\boldsymbol{x}) = \frac{1}{2}(\chi_b^2 - \chi_s^2), \qquad 16.$$

where $\chi^2 = (\boldsymbol{x}-\boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})$. If the covariance matrices for the two classes are equal, that is, if $\boldsymbol{\Sigma_s} = \boldsymbol{\Sigma_b} = \boldsymbol{\Sigma}$, then one obtains Fisher's linear discriminant. If not, Equation 16 is a quadratic function of the feature variables. However, if we consider the augmented feature space with variables $x_1, x_2, x_1^2, x_2^2,$ and $x_1 x_2$, then the quadratic discriminant function in the original space becomes a linear discriminant in the augmented five-dimensional space.

The Gaussian Classifier is sometimes referred to as the *H*-matrix method, where $H = \boldsymbol{\Sigma}^{-1}$. This method is employed in electron identification in DØ (see Ref. 43) where feature variables characterizing longitudinal and transverse shower shapes in the calorimeter are used to construct a Gaussian classifier.

So far, I have discussed Gaussian densities as the relevant models. In the case of non-Gaussian densities, one can still use linear methods provided that the data are mapped into a space of sufficiently high dimensions as is done in support vector machines (39).

3.3 Naïve Bayes or Likelihood Discriminant

When the feature variables are statistically independent, the multivariate densities can be written as products of one dimensional densities without loss of information. In this case, the discriminant in Equation 7 becomes

$$D(\boldsymbol{x}) = \frac{\Pi_i s_i(x_i)}{\Pi_i s_i(x_i) + \Pi_i b_i(x_i)}, \qquad 17.$$

where $s_i(x_i)$ and $b_i(x_i)$ are the densities of the i^{th} variable from the signal and background classes, respectively. The univariate densities can be readily estimated by simple parameterizations (or by non-parametric methods discussed below). It may be computationally easier to parameterize the likelihood ratio $L_i = s_i(x_i)/b_i(x_i)$ of the individual variables and calculate the discriminant as $D(x) = L/(1+L)$ where $L = \exp\sum L_i$ (24).

3.4 Kernel-based Methods

In principle, multivariate densities can be estimated simply by histogramming the multivariate data $\boldsymbol{x}$ in M bins in each of the d feature variables. The fraction of data points that fall within each bin yields a direct estimate of the density at the value of the feature vector $\boldsymbol{x}$ at, say, the center of the bin. The bin width (and therefore the number of bins M) must be chosen such that the structure in the density is not washed out (due to too few bins) and such that the density estimation is not too spiky (due to too many bins). Unfortunately, this method suffers from the curse of dimensionality as in the case of the standard grid search. We would need a huge number of data points in order to fill the bins with a sufficient number of points.

More efficient methods for density estimation are based on sampling neighborhoods of data points. Let us take the simple example of a hypercube of side h as the kernel[5] function H in a d-

[5] A kernel is a symmetric function that integrates to unity.

dimensional space. Such a hypercube can be placed at each point $\boldsymbol{x}_n$, counting the number of points that fall within it and dividing that number by the volume of the hypercube and the total number of points:

$$\tilde{p}(\boldsymbol{x}) = \frac{1}{N}\sum_{n=1}^{N}\frac{1}{h^d}H\left(\frac{\boldsymbol{x}-\boldsymbol{x}_n}{h}\right), \qquad 18.$$

where N is the total number of points, and $H(u)$=1 if $\boldsymbol{x}$ is in the hypercube, otherwise $H(u)$=0.

The method is the same as histogramming, but with overlapping bins (hypercubes) placed around each data point. Smoother and more robust density estimates can be obtained by using smooth functional forms for the kernel function. A common choice is a multivariate Gaussian,

$$H(u) = \frac{1}{(2\pi h)^{d/2}}\exp\left(-\frac{\left\|\boldsymbol{x}-\boldsymbol{x}_n\right\|^2}{2h^2}\right), \qquad 19.$$

where the width of the Gaussian acts as a smoothing parameter (the bandwidth) that is chosen appropriately for the problem. If the kernel functions satisfy

$$H(u) \geq 0; \int H(u)du = 1, \qquad 20.$$

then the estimator satisfies $\tilde{p}(x) \geq 0$ and $\int \tilde{p}(\boldsymbol{x})d\boldsymbol{x} = 1$.

Bandwidth selection is a critical aspect of this algorithm. In the standard kernel methods, the parameter h is the same for all points and consequently the density estimation can be over-smoothed in some regions and spiky in some others. This problem is addressed by use of adaptive kernels or the K-nearest neighbor approach.

Adaptive Kernels: In the adaptive kernel method, the kernel width depends on the local density of data points. We can define the local kernel width $h_i = \lambda_i h$ where h is the global width and λ_i is a scaling factor determined by the local density. A simple ansatz is that λ_i is inversely proportional to the square root of the density of sample points in the locality. Even in this method, setting the global width is an issue, especially for multiple dimensions.

K-Nearest Neighbor Method: In this method, a kernel, say a hypersphere, is placed at each point $\boldsymbol{x}$ and instead of fixing the volume V of the hypersphere and counting the number of points that fall within it, we vary the volume (i.e., the radius of the hypersphere) until a fixed number of points lie within it. Then, the density is calculated as,

$$\tilde{p}(\boldsymbol{x}) = \frac{K}{NV}. \qquad 21.$$

The class densities thus estimated can be used to calculate the discriminant from Equation 7.

The probability density estimation (PDE) method using kernels can be used in both discrimination and regression problems. The method is employed by DØ in an analysis that extracts the top quark mass in dilepton final states (44).

3.5 Neural Networks

Feed-forward neural networks, also known as multilayer perceptrons (MLP), are the most popular and widely used of the multivariate methods. A schematic of a neural network (NN) is shown in **Figure 3(a)**. An MLP consists of an interconnected group of neurons or nodes arranged in layers; each node processes information received by it with an activation (or transformation) function, then passes on the result to the next layer of nodes. The first layer, called the input layer, receives the feature variables, followed by one or more hidden layers of nodes and the last layer outputs the final response of the network. Each of the interconnections is characterized by a weight, and each of the processing nodes may have a bias or a threshold. The weights and thresholds are the network parameters, often collectively referred to as weights, whose values are learned during the training phase. The activation function is generally a non-linear function that allows for flexible modeling. NNs with one hidden layer are sufficient to model the posterior probability to arbitrary accuracy. Although neural networks are typically described, as above, in terms of neurons and activation, it is useful to think of them as simply a specific class of non-linear functions.

In the schematic shown in **Figure 3(a)** which has one hidden layer of nodes and, a data set with *d* feature variables $\boldsymbol{x} \equiv \{x_1, x_2, \ldots x_d\}$, the output of the network is

$$O(\boldsymbol{x}) = f(\boldsymbol{x}, \boldsymbol{w}) = g(\theta + \sum_j w_j h_j) = p(s \mid \boldsymbol{x}), \qquad 22.$$

where h_j is the output from the hidden nodes:

$$h_j = g(\theta_j + \sum_i w_{ij} x_i). \qquad 23.$$

The non-linear activation function g is commonly taken as a sigmoid

$$g(a) = \frac{1}{1+e^{-a}}. \qquad 24.$$

If $g(a) \sim a$, the outputs h_j at the hidden layer would be linear combinations of the inputs and the network with a single layer of weights would be a linear model. The sigmoid function is linear close to $a \sim 0$, nonlinear for higher values of *a*, and saturates for large values; it maps the input interval $(-\infty, \infty)$ onto (0,1). Therefore, a network with a sigmoidal activation function contains a linear model as a special case. The function g at the output is usually chosen to be a sigmoid for classification and a linear function for regression. The network parameters are determined by minimizing an empirical risk function, usually the mean square error between the actual output O_p and the desired (target) output y_p,

$$E = \frac{1}{N} \sum_{p=1}^{N} (y_p - O_p)^2, \qquad 25.$$

over all the data in the training sample, where *p* denotes a feature vector[6]. As mentioned in section 2.3, a network trained for signal/background discrimination with y_p=1 for the signal class and y_p=0 for the background can directly approximate the Bayesian posterior probability, $p(s \mid x)$.

Two examples of using NNs for binary classification where discriminating boundaries are nonlinear are shown in **Figures 3 & 4**. For results shown in **Figure 3**, the same data sets as in the example of **Figure 2** are used to train an NN with 2 inputs, 8 hidden nodes, one output node (2-8-1) to map the feature space onto a one-dimensional discriminant. Any cut on the NN

[6] Note that Equation 25 is essentially same as Equation 2.

discriminant, shown in **Figure 3(b)**, therefore, corresponds to a nonlinear contour cut (decision boundary) in the feature space as shown in **Figure 3(c)**. The signal probability in feature space as calculated by the NN is shown in **Figure 3 (d)**. **Figure 4** shows results with an NN (2-10-1) for a slightly more complicated problem. The simulated data for the two classes, the NN decision boundaries along with the Bayes decision boundaries calculated from the known class densities are shown in **Figure 4(a)**. **Figure 4(b)** shows the signal probability in feature space given by the NN. For feature space with dimensions larger than two, the discriminant for binary classification will still be one-dimensional and a cut placed on the discriminant will correspond to a hypersurface in the feature space.

There are several heuristics that are helpful in the construction of NNs. Since the hidden nodes are critical in the modeling of the function, the number needed depends on the density of the underlying data. Too few nodes lead to under-fitting and too many lead to over-fitting. To avoid over-fitting, one can employ *structure stabilization* (optimizing the size of the network) and *regularization.* In the former, one starts either with large networks and then *prunes* connections or starts with small networks and adds nodes as necessary. In regularization, one penalizes complexity by adding a penalty term to the risk function. It is considered useful to scale the inputs appropriately. The standard advice is to scale the magnitude of the input quantities such that they have a mean around zero and a standard deviation of one. Generally, it suffices to make sure that the inputs are not much greater than one. The starting values of weights are chosen randomly. When using standard scaled inputs as suggested above, the starting weights can be chosen randomly in the range of -0.7 to 0.7. A network is trained by cycling through the training data hundreds or thousands of times. The performance of the network is periodically tested on a separate set of data. The training is stopped when the error on the test data begins to increase.

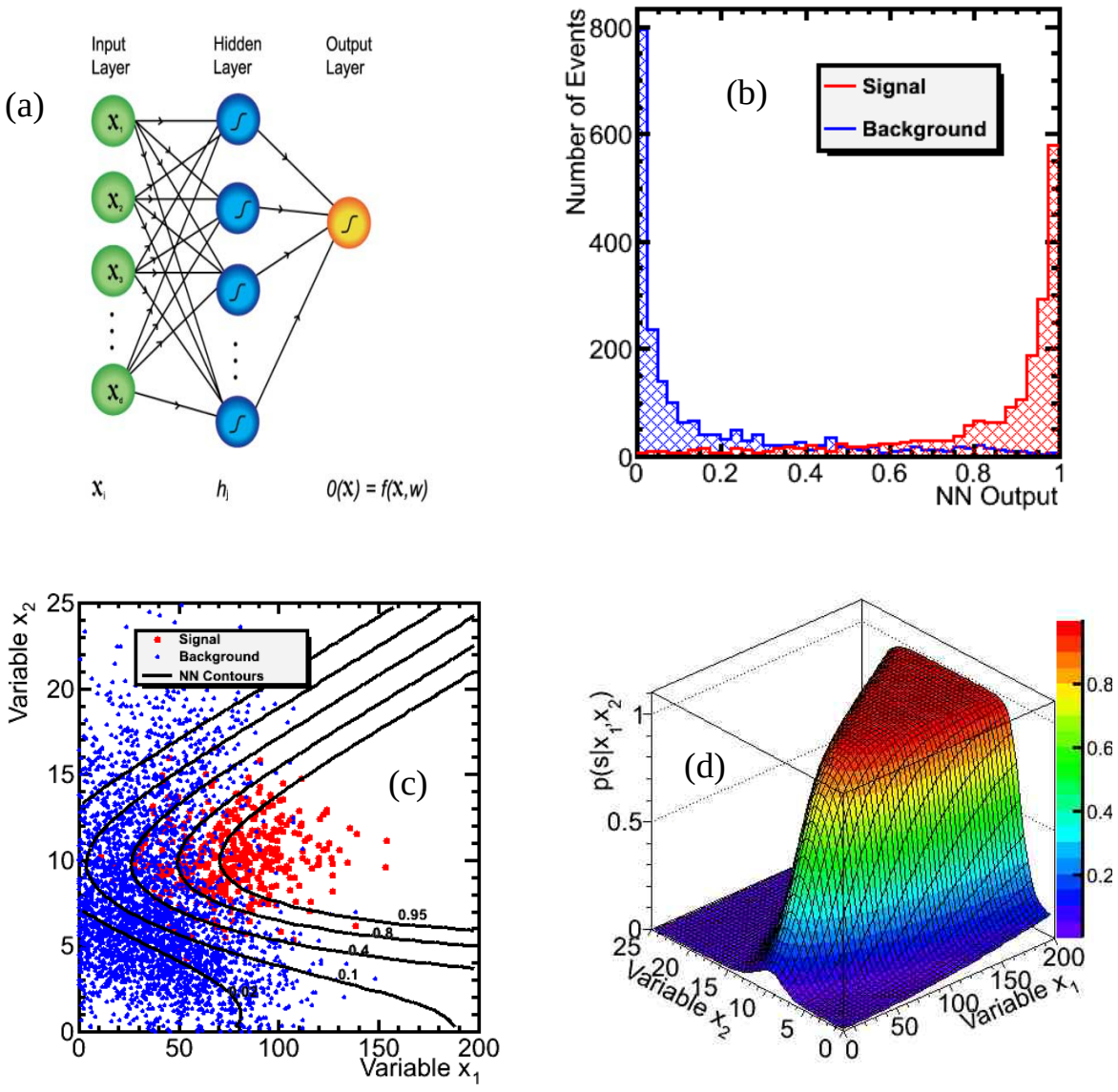


Figure 3

(a) A schematic representation of a three-layer feed-forward neural network; (b) distributions of NN output (discriminant) trained on data shown in (c) (same data as in **Figure 2**); (c) equi-probability contours (decision boundaries) corresponding to cuts of 0.02, 0.1, 0.4, 0.8 and 0.95 on the NN output shown in (b) superposed on signal and background data distributions. The data points to the right of each contour have NN output values above the displayed cut. (d) Signal probability surface as given by the NN output, $D(x_1,x_2) \sim p(s|x_1,x_2)$, in the feature space.

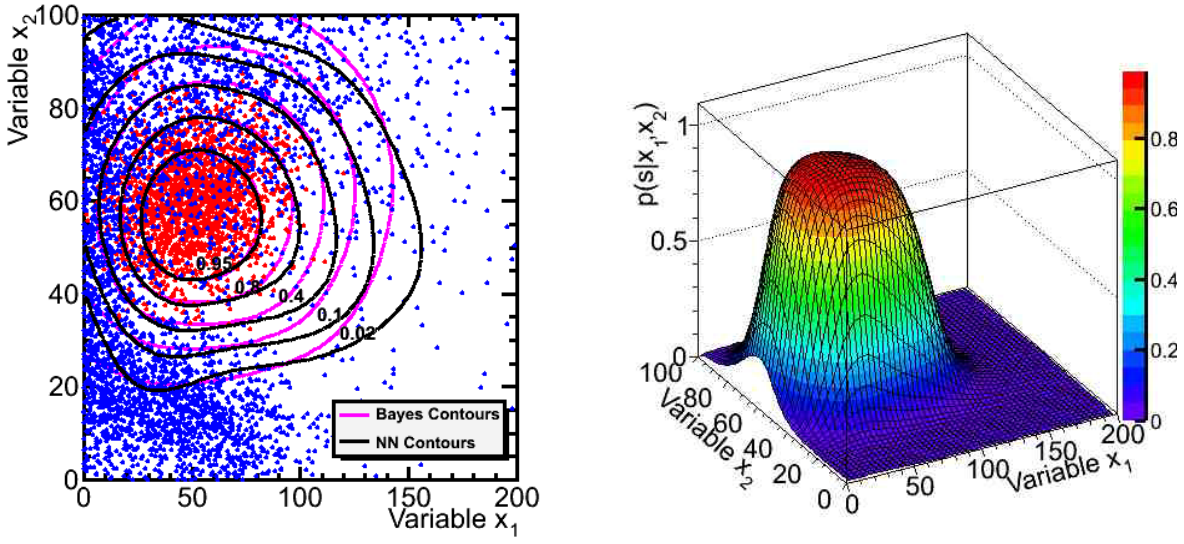


Figure 4

(a) Decision boundaries calculated by an NN trained on the simulated data for two classes (shown superposed) compared with Bayes decision boundaries calculated as per Equation 7 using known class densities. Data points within each NN contour have NN output values above the corresponding cut value shown. (b) Signal probability given by NN output as a function of the feature variables x_1 and x_2.

3.6 Bayesian Neural Networks

In the conventional methods for training NNs, one attempts to find a single "best" network, that is, a single "best" set of network parameters (weights). Bayesian training provides a posterior density for the network weights: $p(\boldsymbol{w} \mid \textit{training data})$. The idea behind Bayesian neural networks (BNN) is to assign a probability density to each point $\boldsymbol{w}$ in the parameter space of the NN. Then, one performs a weighted average over all points, that is, over all possible networks. Given the

training data T= $\{y,\boldsymbol{x}\}$, the probability density assigned to point $\boldsymbol{w}$ (i.e., to a network) is given by Bayes' theorem

$$p(\boldsymbol{w} \mid T) = \frac{p(T \mid \boldsymbol{w})\, p(\boldsymbol{w})}{p(T)}. \qquad 26.$$

Then, for a given input vector, the posterior distribution of weights gives rise to a distribution over the outputs of the networks which are then averaged,

$$\tilde{y}(\boldsymbol{x}) = \int f(\boldsymbol{x}, \boldsymbol{w})\, p(\boldsymbol{w} \mid T) d\boldsymbol{w}. \qquad 27.$$

Implementation of Bayesian learning is far from trivial given that the dimensionality of the parameter space is typically very large. Currently, the only practical way to perform the high-dimensional integral in Equation 27 is to sample the density $p(\boldsymbol{w} \mid T)$ in some appropriate way, and to approximate the integral using the average

$$\tilde{y}(\boldsymbol{x}) \approx \frac{1}{K} \sum_{k=1}^{K} f(\boldsymbol{x}, \boldsymbol{w}_k), \qquad 28.$$

where K is the number of points $\boldsymbol{w}$ sampled. Typically the sampling is done using a Markov chain Monte Carlo method (45).

There are several advantages to BNNs over conventional NNs (45, 46). Each point $\boldsymbol{w}$ corresponds to a different NN function in the class of possible networks and the average is over networks. Therefore, one expects to produce an estimate of the signal class probability $p(s \mid \boldsymbol{x})$ that is less likely to be affected by over-training. Moreover, in the Bayesian approach, there is less need to severely limit the number of hidden nodes because a low probability density will be assigned to points $\boldsymbol{w}$ that correspond to unnecessarily large networks, in effect pruning them away. The network can be as large as is computationally feasible so that the class of functions defined by the network parameter space includes a subset with good approximations to the true mapping.

One of the issues in the training of a BNN is to check that the Markov chain has converged. There are many heuristics available for this. But, in practice, one runs many chains or a single long chain and checks that the results are stable. Also, every Bayesian inference requires the specification of a prior. The choice, in this case, is not obvious. However, a reasonable class to choose from is the class of Gaussian priors centered at zero that favors smaller rather than larger weights. Smaller weights yield smoother fits to data.

3.7 Decision Trees

Decision trees (DT) (47, 48) employ sequential cuts as in the standard grid search to perform the classification (or regression) task, but with a critical difference. At each step in the sequence, the best cut is searched for and used to split the data and this process is continued recursively on the resulting partitions until a given terminal criterion is satisfied. The DT algorithm starts at the so-called *root node* (see **Figure 5**) with the entire training data set containing signal and background events. At each iteration of the algorithm, and for each node, one finds the best cut for each variable and then the best cut overall. The data are split using the best cut thereby forming two branch nodes. One stops splitting when no further reduction in impurity is possible (or when the number of events is judged too small to proceed further). The measure that is commonly used to quantify impurity is the so called the *Gini* index, which is given by,

$$Gini = (s+b)P(1-P) = \frac{sb}{s+b}, \qquad 29.$$

where $P = s/(s+b)$ is the signal purity ($\equiv D(\boldsymbol{x})$ in our definition), and s and b are the signal and background counts at any step in the process. The splitting at a branch node is terminated if the impurity after the split is not reduced. The node then becomes a terminal node (also known as a *leaf*) and an output response – for instance, $D(\boldsymbol{x}) = s/(s+b)$ is assigned to the leaf.

Note that geometrically the DT procedure amounts to recursively partitioning the feature space into hypercubic regions or bins with edges aligned with the axes of the feature space. So essentially, a DT creates M disjoint regions or a d-dimensional histogram with M bins of varying bin-size and a response value is assigned to each of these bins. A DT, therefore, gives a piece-

wise constant approximation to the function being modeled, say, the discriminant $D(\boldsymbol{x})$. As the training data set becomes arbitrarily large, and the bin sizes approach zero, the predictions of a DT approaches that of the target function, provided the number of bins also grow arbitrarily large (but at a rate slower than the size of the data set).

The DT algorithm is applicable to discrimination of *n*-classes, even though what I have described is the binary decision tree method used in 2-class signal/background discrimination. An illustration of a binary decision tree for a problem characterized by two variables and the resulting partition of the feature space are shown in the schematics in **Figure 5**. Results of using the boosted decision tree algorithm for the previous 2D example are also shown.

Decision trees are very popular because of the transparency of the procedure and interpretation. They also have some other advantages: (a) tolerance to missing variables in the training data and test data; (b) insensitivity to irrelevant variables since the best variable on which to cut is chosen at each split and therefore ineffective ones do not get used; (c) invariance to monotone transformation of variables which makes preprocessing of data unnecessary. However, decision trees also have serious limitations: (a) instability with respect to the training sample (a slightly different training sample can produce a dramatically different tree); (b) sub-optimal performance due to the piece-wise constant nature of the model, which means that the predictions are constant within each bin (region represented by a leaf) and discontinuous at its boundaries; (c) poor global generalization because the recursive splitting results in the use of fewer and fewer training data per bin and only a small fraction of the feature variables may be used to model the predictions for individual bins.

Most of these limitations, fortunately, have been overcome with the use of ensemble learning techniques such as boosting, bagging and random forests, which I discuss below.

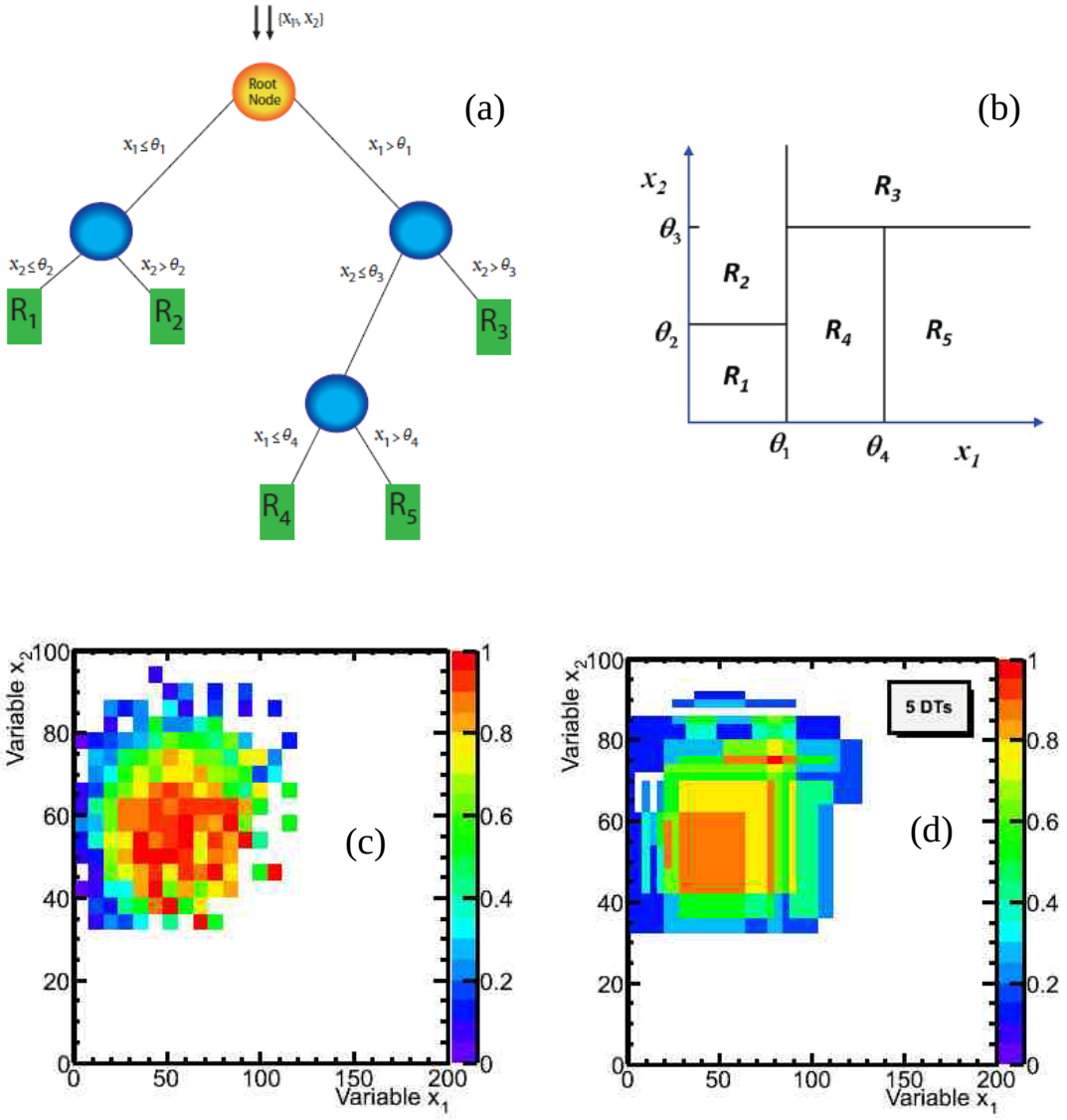


Figure 5

(a) A schematic of a binary decision tree with two feature variables x_1 and x_2. (b) Illustration of the corresponding partitions of the 2D feature space (see text for details). (c) Signal probability calculated as the ratio of signal counts divided by signal +background counts in bins of two-dimensional histograms for data set shown in the previous figure. (d) Signal probability approximated using five decision trees (using AdaBoost) using the same data.

Ensemble Learning:

We have discussed several methods to perform functional approximation. The goal is to minimize an appropriate cost function and create approximations that provide best predictive performance and incorporate the correct tradeoff between bias and variance. Bias in a predictor[7] comes from differences between the learned function and the true function, while variance is a measure of the sensitivity of the learned function to inputs. Averaging over multiple predictors has been shown to provide the best compromise between bias and variance, while providing generalization error that can be much smaller than that of an individual predictor. The fundamental insight is that it is possible to build highly effective classifiers from predictors of modest quality.

Here I briefly outline a few of these ensemble techniques (49, 50).

Boosting: The idea behind boosting is to make a sequence of classifiers that work progressively harder on increasingly "difficult" events. Instead of seeking one high performance classifier, one creates an ensemble of classifiers, albeit weak, that collectively have a "boosted" performance. For an ensemble of *M* classifiers, one can write, for the predictions of the final classifier,

$$\tilde{y}(\boldsymbol{x})=\sum_{m=1}^{M}\alpha_m y_m(\boldsymbol{x},\boldsymbol{w_m}), \qquad 30.$$

where $\boldsymbol{w_m}$ are the parameters of the m^{th} classifier. The weighting coefficients $\alpha_{\boldsymbol{m}}$ are defined and determined differently in each algorithm. In AdaBoost, the first successful high performance boosting algorithm, the underlying functions are decision trees. $\tilde{y}(\boldsymbol{x})$, in that case, is a boosted decision tree (BDT). The coefficients are taken as $\alpha_{\boldsymbol{m}} = \ln\left[\frac{1-\varepsilon_m}{\varepsilon_m}\right]$ where $\varepsilon_{\boldsymbol{m}}$ is the (event-weighted) misclassification error for the m^{th} decision tree. The BDTs, unlike single DTs, have been found to be very robust. A striking feature of AdaBoost is that the misclassification rate on the training set approaches zero exponentially as the number of trees increases but the error rate on an independent test sample remains essentially constant. This resistance of the AdaBoost to over-fitting is not yet fully understood.

[7] A predictor is a discriminant, a classifier or an estimator.

Bagging: Bagging (Bootstrap Aggregating) is a simple average of the outputs of *M predictors*, usually classifiers, where each is trained on a different bootstrap sample (i.e., a randomly selected subset) drawn from a training sample of N events. In Equation 30, $\alpha_m = 1/M$ in case of bagging.

Random Forests: In principle, this algorithm, like the other two described above, can be applied to any predictor whose construction can incorporate randomization. In practice, however, random forests use decision trees. Many classifiers are trained, each with a randomly chosen subset of feature variables at each split providing a random forest of decision trees. The output for each event is the average output of all trees in the random forest. Further randomization can be introduced through the use of bootstrap samples as in the case of bagging.

3.8 Other Methods

I briefly discuss two other (unrelated) techniques that are used in HEP analyses – the genetic algorithms which are used for optimization of parameter searches, and the matrix element method, a semi-analytical approximation of probability densities.

Genetic Algorithms:

While neural networks are inspired by the workings of the human brain, genetic algorithms (GA) are inspired by ideas from evolutionary biology and genetics. Genetic algorithms evolve a population of candidate solutions for a problem using principles that mimic those of genetic variation and natural selection, such as crossover, inheritance, mutation, and survival of the fittest. These algorithms can be used to determine the parameters of a model in functional approximation.

The steps involved in a GA are as follows – (1) randomly generate an initial population of candidate solutions (or parameters $\boldsymbol{w}$) , (2) compute and save the fitness for each individual solution in the current population, (3) generate n off-springs of the members of the population by crossover (i.e., swap some of the parameter values between candidate vectors) with some probability and mutate the off-springs with some probability, (4) replace the old population with the new one, which gives the new generation. The procedure is repeated until a set of sufficiently fit candidates have emerged.

Genetic algorithms can be applied to any optimization problem. One such application is in Neuroevolution (51), which allows both the NN structure and the NN parameters (weights and thresholds) to be evolved.

Matrix Element Method:

The Matrix Element (ME) method is not a machine-learning method but rather a semi-analytical calculation of the probability densities $p(\boldsymbol{x} \mid \boldsymbol{s})$, $p(\boldsymbol{x} \mid \boldsymbol{b})$ from which a discriminant can be computed using Equation 7 in the usual way. It is motivated by the desire to use the theoretical knowledge about physics processes and measured observables (four-vectors) directly to construct multivariate discriminants and estimators. All of the physics information about a high energy event is contained in the matrix element describing the collision process. The probability to observe data $\boldsymbol{x}$ (typically the four-vectors of objects in the final state) from a given physics process can be written as

$$p(\boldsymbol{x} \mid process_i) = \frac{1}{\sigma_i}\frac{d\sigma_i}{d\boldsymbol{x}} \qquad 31.$$

where $\frac{d\sigma_i}{dx}$ is the differential cross-section. The differential cross-section is a convolution of the cross-section (proportional to the square of the matrix element) for the partonic process, the parton distribution functions (PDFs) and the response function of the detector – integrated over phase space and summed over all possible configurations that contribute to the final state. The detector response function, say $\xi(\boldsymbol{x}, \boldsymbol{y})$, gives the probability for partonic variables $\boldsymbol{y}$ to give rise the observation $\boldsymbol{x}$ after event reconstruction.

In case of parameter estimation, the event probability is built using

$$P_{event}(\boldsymbol{x}, \theta) = \sum_{i=process} f_i p_i(\boldsymbol{x} \mid \theta), \qquad 32.$$

where the summation is over all processes (signal and backgrounds) that may give rise to the observed event. One then uses either a Bayesian or maximum likelihood fit to extract the parameters of interest θ.

The ME method was first used in the measurement of the top quark mass in the lepton+jets final state by the DØ collaboration (52). Since then it has been used in a number of other analyses. The method is computationally demanding because of the need to perform a multi-dimensional integration for each feature vector.

3.9 Tools

Many easy-to-use packages that implement the methods discussed above are now widely available. Some of them are specific NN implementations such as Jetnet (53) and MLPFit (http://schwind.home.cern.ch/schwind/MLPfit.html), Stuttgart Neural Network Simulator (http://www.ra.cs.uni-tuebingen.de/SNNS/) for NNs, and FBM (45) and NEUROBAYES (54) for BNNs . NNs with genetic evolution of weights are implemented in the NEAT (55) package. RULEFIT (56) implements rule-based learning methods such as decision trees. There are general multivariate analysis packages such as TMVA (57) in ROOT (http://root.cern.ch) and StatPatternRecognition (58) that have many methods implemented. The TMVA software, for example, enables the user to try out different methods simultaneously and compare their efficacies directly.

4 ANALYSIS EXAMPLES

Because of their demonstrated power, advanced analysis methods are becoming common tools in several aspects of HEP analysis – most notably, in the identification of particles (e.g., electrons, photons, τ and *b*- jets) and in signal and background discrimination.

In this section, I briefly describe a few important physics analyses that illustrate both the potential of the methods and the challenges. I begin with the first precision measurement of the top quark mass at DØ. Then, I briefly discuss the recent observation of single top quark production which was an important milestone not only because it provides further validation of the SM and because the single top production rate is particularly sensitive to new physics beyond the SM, but also because of its sophisticated use of the multivariate methods. This observation of single top production also provides an analysis test-bed for what has become the Holy Grail of particle physics, namely, the search for the Higgs boson. Finally, I make some comments on the

Higgs boson searches and briefly discuss an interesting application alluded to earlier, namely, the fitting of the parton distribution functions using neural networks and genetic algorithms.

4.1 An Early Successful Example: The Top Quark Mass

The top quark mass measurement was the first important result in hadron collider physics to benefit from multivariate methods. The DØ experiment did not have a silicon vertex detector during the first run (Run I) of the Tevatron. Instead, *b*-tagging relied on the presence of soft muons from the decay of *b*-quarks, the efficiency for which was only 20% in the $t\bar{t} \to W^+bW^-\bar{b} \to l\nu bq\bar{q}\bar{b}$ process. At CDF, which had the ability to tag *b*-jets with its silicon vertex detector, the efficiency was approximately 53%. Nonetheless, in spite of this technical disadvantage, DØ measured the top quark mass with a precision approaching that of CDF by using multivariate techniques for separating signal from background.

Two multivariate methods – a variant of the likelihood discriminant technique (naïve Bayes) and a feed forward NN method, were used to compute a discriminant $D \equiv p(top \mid \boldsymbol{x})$ for each event. A fit of the data, based on a Bayesian method (59), to discrete sets of signal and background models in the $[p(top \mid \boldsymbol{x}), m_{fit}]$ plane was used to extract the top quark mass (m_{fit} is the mass from a kinematic fit to the $t\bar{t}$ hypothesis). The distributions of variables and the discriminants are shown in **Figure 6**. By combining the results of the fits from the two methods, DØ measured $m_t = 173.3 \pm 5.6(stat) \pm 5.5(syst)\ \mathrm{GeV/c^2}$ (24), which was a factor of two better than the result obtained using conventional methods and the same data set. This example underscores that even very early in the life of an experiment, huge gains can be obtained through a judicious, yet advanced treatment of a few simple variables.

Most of the measurements of the top quark mass at CDF and DØ since this first successful application of a multivariate approach have used some kind of multivariate method – NNs, matrix element or the likelihood method. Currently, the measured world average top quark mass is $m_t = 173.3 \pm 1.1\ \mathrm{GeV/c^2}$ (60).

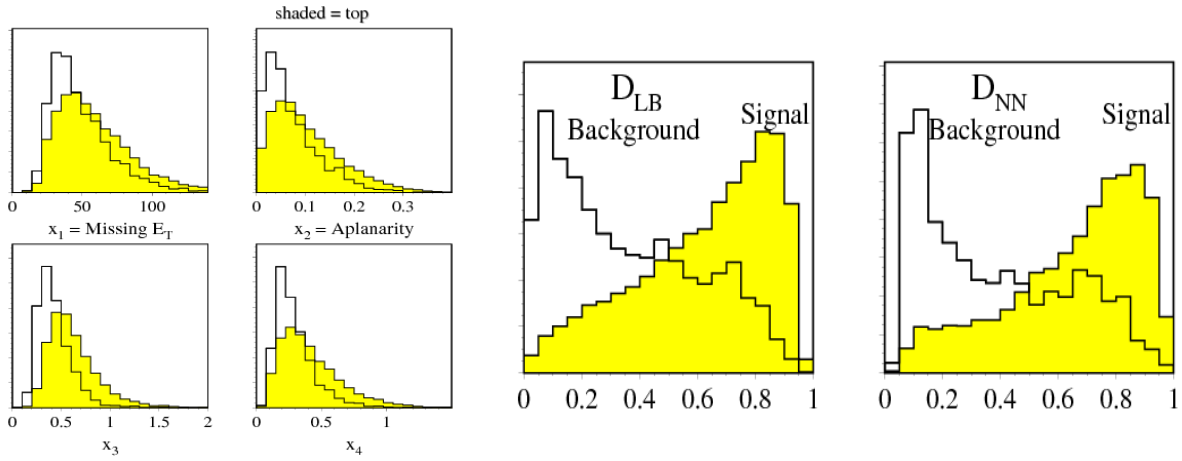


Figure 6

(Left) Distributions of discriminant variables x_1, x_2 , x_3 , x_4 (see Ref. 24 for definitions) used in the first direct precision measurement of the top quark mass at DØ and (right) the distributions of the final multivariate discriminants. The filled histograms are for signal and unfilled ones are for background. All histogram areas are normalized to unity.

4.2 Single Top Quark Production at the Tevatron

The top quark was discovered in 1995 through the pair production process $p\bar{p} \rightarrow t\bar{t}$ via the strong interaction. The SM predicts electroweak production of a single top quark along with a *b*-quark or a *b*-quark and a light quark with a cross section of $\sigma_t \sim 3$ pb ($\sigma_{t\bar{t}} \sim 6.8$ pb, assuming m_t = 175 GeV/c^2). Although the top quark discovery was made with data sets corresponding to an integrated luminosity of ~ 50 pb^{-1}, the single top quark observation required 50 - 60 times more luminosity (2.3 fb^{-1} at DØ, 3.2 fb^{-1} at CDF) and took another fourteen years (61,62). What makes single top quark events so difficult to extract from data is the fact that the final state contains fewer features than in $t\bar{t}$ to exploit for the purpose of discriminating signal from the overwhelming background of W+jets and QCD multijet production (wherein a jet is misidentified as a lepton). The use of multivariate methods was indispensable in the analyses in both experiments.

Single top quarks are produced at the Tevatron through the s-channel $q\bar{q} \to t\bar{b}$ ($\sigma \sim 0.95$ pb) and t-channel $q'g \to tqb$ ($\sigma \sim 2.05$ pb) processes (63). The top quark almost always (as per the SM) decays to a W boson and a b-quark. Final state channels involving leptonic decays of the W boson and at least one b-tagged jet are considered by both experiments, in order to have better signal-to-background ratio from the outset. Both experiments use NNs to enhance the b-tag efficiency and purity.

After implementing the initial selection criteria, requiring a high-p_T lepton, high-p_T jets, and large missing transverse energy, both experiments estimate a very similar overall signal-to-background ratio of ~ 0.05, (CDF: 0.053, DØ: 0.048). CDF observes 4726 events while expecting 4524±511 background and 255 ± 21 signal events (62), while DØ observes 4,519 events with an expected background of 4427±213 events and an expected signal of 223 ± 30 events (61). At this point in the analysis, the signal, in both cases, is smaller than the uncertainties in the background estimates.

The single top signal is further discriminated from the backgrounds through the use of multivariate techniques. DØ performs three independent analyses using (a) BNNs, the first such application in HEP, (2) BDTs and (3) the ME method. In addition to these techniques, CDF uses the likelihood discriminant method. Because the results from these methods are not completely correlated, the discriminant outputs are further combined into a single discriminant (referred to as the combination discriminant by DØ, and the super discriminant by CDF). The final discriminant is then used to extract the cross section for single top quark production and the signal significance. The signal to background ratio in the signal region of the final discriminants (>5) is about a factor of 100 larger than that of the base samples. Using the final discriminants and a Bayesian technique, the cross sections are measured to be 2.3±0.5 pb by CDF (at m_t=175 GeV/c^2) and 3.94±0.88 pb by DØ (at m_t = 170 GeV/c^2). The significance of the signal is five standard deviations in both results.

The analyses, depending on the channel, use as few as 14 to as many as 100 variables. To ensure that the background is modeled correctly, both CDF and DØ compared thousands of distributions of the data sample with the modeled backgrounds. The output discriminant modeling was also verified at various stages with control samples from known physics processes.

The observation of the single top quark production at the Tevatron is described in a dedicated review in this volume (64).

4.3 Searches for the Higgs Boson

Since the discovery o the top quark 15 years ago, the Higgs boson has been the most sought after particle. The intense searches by the four experiments (ALEPH, DELPHI, L3 and OPAL) at the e^+e^- collider LEP at CERN ($\sqrt{s}$ = 189 - 209 GeV) before it was decommissioned, resulted in a 95% C.L. lower bound on the Higgs boson mass of 114.4 GeV/c^2 (65-67). In 2000, studies of the Higgs discovery reach at the Tevatron (68, 69) led to the conclusion that the use of multivariate methods could significantly enhance the potential for its discovery at the Tevatron if the planned luminosity upgrades for Run II were to be implemented. With the help of several fb^{-1} of data accumulated in Run II and the help of advanced analysis techniques, the Tevatron experiments have reached the sensitivity levels to detect hints of the Higgs boson or to rule out certain masses beyond the range of LEP exclusion.

The predicted cross sections for the production of SM Higgs at the Tevatron are more than an order of magnitude smaller than for single top production in the mass regions of interest. The dominant production process at the Tevatron is $gg \rightarrow H$, with cross sections between 1 pb and 0.2 pb in the mass range of 100-200 GeV/c^2. The cross sections are between 0.5 pb and 0.03 pb for $q\bar{q}' \rightarrow WH$ or ZH and between 0.1 pb and 0.02 pb for $q\bar{q} \rightarrow q\bar{q}H$ in the same mass range. The dominant decay channels are $H \rightarrow b\bar{b}$ for m_H < 135 GeV/c^2 and H → WW* for m_H > 135 GeV/c^2 (W* is off-shell if m_H < 160 GeV/c^2). The $gg \rightarrow H \rightarrow b\bar{b}$ channel suffers from very large QCD multijet background. Therefore, for m_H < 135 GeV/c^2, the WH and ZH production channels are used for the searches. For m_H > 135 GeV/c^2, gg→ H → WW* is the most promising channel.

The searches for the SM Higgs boson have been performed in 90 mutually exclusive final states (36 for CDF and 54 for DØ). The analysis channels are sub-divided based on lepton-type, number of jets and number of *b*-tags. The most important features that can help discriminate Higgs signal from background are efficient *b*-tagging and good dijet mass resolution (in low

mass Higgs searches). To achieve high *b*-tag efficiency, both experiments use a NN to combine outputs of simpler discriminants based on secondary vertex and decay track and jet information. CDF constructs two separate networks to discriminate *b*-jets from c-jets and *b*-jets from light-quark jets. DØ has built an NN *b*-tagger to discriminate *b*-jets from all other types of jets. The DØ NN *b*-tagger gives significantly higher efficiencies compared to that of the next best method based on the JLIP (jet lifetime probability) algorithm (70). The benefit of the *NN* tagger is estimated to be equivalent to nearly doubling the luminosity (71) in SM Higgs boson searches. Also, CDF has used a multivariate approach for *b*-jet energy correction and has demonstrated improved dijet mass resolution which in turn helps the Higgs search sensitivity (72).

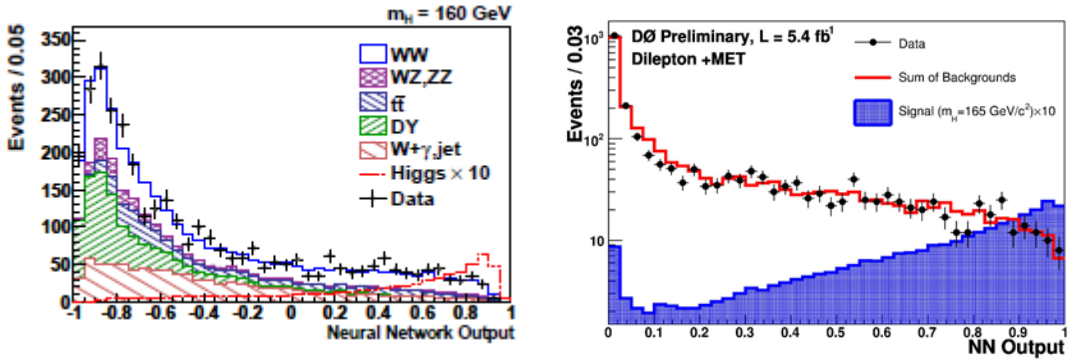


Figure 7.

Neural network output distributions from H→ WW* analyses at the Tevatron. (Left) CDF results showing data compared with total and individual backgrounds. Also shown is the expected distribution for the SM Higgs signal for m_H = 160 GeV/c^2. (Right) DØ results comparing data with total background in the dilepton + missing transverse energy channel. Here, the Higgs signal distribution is shown for m_H = 165 GeV/c^2. In both cases, the signal is scaled up by a factor of ten relative to the SM prediction.

Both CDF and DØ use NNs, boosted decision trees and other multivariate discriminants in all their analyses. In the case of $H \to WW^*$ analysis, CDF has found that the multivariate techniques provide a gain factor of 1.7 to 2.5 (depending on m_H) in effective integrated luminosity over an optimized cut-based selection. Some example NN discriminants are shown

in **Figure 7**. The combined results from the two experiments provide 95% C.L. upper limits on Higgs boson production that are a factor of 2.7 (0.94) times the SM cross-section for $m_H = 115(165)\ \mathrm{GeV/c^2}$. As of December 2009, the combination of results from the two experiments, using data sets of luminosities of up to 5.2 $\mathrm{fb^{-1}}$, has yielded a 95% C.L. exclusion for a SM Higgs for 163 $\mathrm{GeV/c^2} < m_H < 166\ \mathrm{GeV/c^2}$ (73-75).

4.4 Determination of Parton Distribution Functions

One of the exciting applications of multivariate methods is in the parametrization of parton distribution functions (PDFs) with NNs by the *NNPDF* collaboration (76). The PDFs are essential inputs in making predictions of physics processes at hadron colliders. The PDFs are determined by fitting the theoretical predictions to various sets of experimental measurements, primarily from deep-inelastic scattering of leptons on hadrons (or nuclei). The Tevatron experiments have produced numerous results on a variety of hard interaction processes, thereby providing precision tests of the SM akin to the LEP and SLC electroweak measurements. The tests of these results as well as predictions for searches beyond the SM demand very precise determination of the PDFs. The PDF uncertainties are sometimes the dominant ones in an analysis, and it is therefore important to have reliable estimates of them.

The standard approach to fitting PDFs is to assume a specific parameterized functional form $f(x,Q_0^2) = x^{\alpha}(1-x)^{\beta}P(x)$ and determine the parameters and the associated errors from a fit to the data by minimizing χ^2. The choice of a specific functional form, as discussed above, results in an inflexible model that introduces unnecessary systematic errors (bias in the region of sparse or no data) and uncertainties that are underestimated unless informed, but ad hoc, corrections are made in the fitting procedure. One way to build more flexible models for PDFs is to rely on the fact that NNs are universal approximators.

To train the NNs that model the PDFs, an ensemble of Monte Carlo data sets, "replicas" of the original experimental data points, are generated. The Monte Carlo data sets have points that are Gaussian distributed about the experimental data points, with errors and covariance equal to the corresponding measured quantities. The Monte Carlo set thus gives a sampling of the probability distribution of the experimental data. The *NN* architecture uses two inputs (x and $\log x$), two hidden layers with two neurons each, and one output [$f(x,Q_0^2)$] at a reference scale Q_0^2. GAs are used for optimization,

yielding a set of *NN* parameters for each replica. The mean value of the parton distribution at the starting scale for a given $\boldsymbol{x}$ is found by averaging over all the networks and the uncertainty is given by the variance of the values. The errors on the PDFs from the *NNPDF* fits are larger than those from other global fitting methods, which may indicate that the latter methods have underestimated the errors, as noted above. Moreover, the PDF uncertainties as a function of $\boldsymbol{x}$ behave as expected: where there are no constraints the uncertainties are large while they are small where the data points provide strict constraints.

5 OPEN ISSUES

Over the past two decades, a lot of experience has been gained in the use of advanced multivariate analysis methods in particle physics and spectacular results have been obtained because of their use. However, there are still some important open issues to be considered.

- **Choosing the variables:** How do we choose the best set of feature variables so that no more than a prescribed amount of information is lost? Even though ranking the efficacy of individual variables for a given application is straightforward, the best way to decide which combination of variables to use can only currently be done, by evaluating the performance of different sets in the given application. Choosing variables will not be an issue if the chosen method can make use of all of the observables directly. This is not an issue for decision trees which can use unlimited number of variables and for the ME method which uses the four-vectors directly. However, validating the modeling of high-dimensional feature space is extremely challenging, as discussed below.
- **Choosing a method:** The so called "No free lunch theorem" states that there is no one method that is superior to all others for all problems. This prompts the question: Is there a way to decide which method is best for each problem? Here, again, one needs to try out different methods for a given application and compare performance. In general, however, one can expect Bayesian neural networks, boosted decision trees and random forests to provide excellent performance over wide range of problems. The ME method, though equally popular, has the disadvantage of being computationally very demanding, and has not been shown to be superior in any of the recent applications. Nor is there any reason to expect it to be superior.

- **Optimal Learning:** How can one test convergence of training and know when the training cannot be improved further? Additionally, how can one verify that a discriminant is close to the Bayes limit? The practice is to stop training when the prediction error on an independent test data set begins to increase. Once again, methods such as BNNs, BDTs and RFs are robust, and are less likely to be affected by overtraining. The most direct way to optimize learning may be to make use of the desired criteria for the specific analysis, that is, say to maximize the signal significance or to minimize the uncertainty in the desired measurement. But then, the interpretation of the discriminant (or the estimator) that one obtains may not be straight-forward.
- **Testing the procedures:** For complicated analyses with many feature variables and small signals, it is necessary to validate the procedure itself or even the whole chain of analysis. Given that doing so is computationally demanding, are there alternative and reliable methods of validation? If not, it is important that an algorithm be computationally efficient so that an analysis can be repeated for many scenarios to ensure the robustness of the results.
- **Modeling of backgrounds:** By far, the most important issue of any non-trivial analysis is how to ensure the correctness of modeling of the backgrounds (and signal). If the data used in modeling the signal and background are faulty, the results will be unreliable. When we use a large number of variables, how do we verify the modeling? How many arbitrary functions of the variables do we need to check? If we use, for example, 100 variables in a multivariate analysis, how can we check modeling of the 100-dimensional density? The larger the number of feature variables used, the higher the burden of verifying the correctness of the modeling. In simple applications such as in particle identification, data from well-understood physics processes can be used to cross-check results. But when discriminating new signals from very large backgrounds, the task of verifying a multivariate density in high dimensions is daunting. The number of combinations of variables and functions thereof that one needs to check grows rapidly with the number of feature variables used. In fact, only an infinitely large number of arbitrary functions can guarantee that all correlations have been verified. The practical questions are as to how many and what checks are needed to achieve a specified level of confidence in the validity of the results?

- **Systematic uncertainties:** To estimate systematic uncertainties in results obtained is in principle straightforward: the uncertainties in the model parameters are propagated through the analysis chain using samples with model parameters altered within uncertainties. Currently, multivariate classifiers (or estimators) are built with model parameters set at their nominal values. A better approach would be to build the classifiers (or estimators) using ensembles of samples that incorporate systematic uncertainties (77).

6 SUMMARY AND PROSPECTS

Advanced analysis methods that match the sophistication of the instruments used in high energy physics research and meet the challenges imposed by the vast data sets and extremely rare signals are imperative. The field already has several high profile results that simply could not have been obtained without such methods. Clearly, there is no going back!

In this article, I have provided an overview, with a unified perspective, of the concepts and methods of optimal analysis. I have discussed a range of methods: from the simple to the sophisticated, especially those that make use of multivariate universal approximators. I have discussed some useful heuristics, outlined open issues and presented just a few examples of successful applications of these methods over the past 15 years. There are other examples from the Tevatron, as well as from LEP (78, 79), HERA (80), the b-factories (81) and neutrino experiments (82).

The LHC experiments (see http://cms.web.cern.ch/cms/; http://atlas.ch/) are planning to use advanced methods in many analyses, but there is some concern about whether their use in the early data-taking period is appropriate due to the expected lack of good understanding of the detectors and systematic effects. These are valid concerns. Nevertheless, there are ample opportunities for safe use of these advanced methods including (a) when it is possible to ascertain the correctness of modeling using well known physics processes such as Z boson decays, QCD $b\bar{b}$ events, etc., and (b) when one has arrived at a set, albeit small, of well understood variables.

Moreover, the following points should be kept in mind:

1. Even two or three variables treated in a multivariate manner can provide significant gains over cuts applied directly to the variables.

2. Combining simple classifiers based on a few variables can help cross-check the modeling more easily and may significantly boost the final performance and precision of the results.
3. One can employ available easy-to-use analysis kits to attempt two or more methods, thereby ensuring that there are no bugs in the procedure or biases arising from possible incorrect use of a method. For example, one could use a feed-forward neural network, Bayesian neural network and boosted decision trees and check the consistency of the results.
4. One can use data as the background model in channels where the signal to background ratio is initially very small. One advantage of this approach is that the data (necessarily) model both physics and instrumental backgrounds precisely.

The bar for the quality of the analyses, especially when a potential discovery is at stake, should be (and almost certainly will be) set very high. The advanced methods I have described need to be used in every step of the data analysis chain, if possible, to reap maximum benefits. But, as is true of all scientific methods and tools, these methods should be used with a great deal of diligence and thought. We would be well served to follow the principle of Occam's razor, which in this context can be stated thus: if we have two analyses of comparable quality we should choose the simpler one. I am sure Einstein would agree.

DISCLOSURE STATEMENT

The author is not aware of any affiliations, memberships, funding, or financial holdings that might be perceived as affecting the objectivity of this review.

ACKNOWLEDGMENTS

My research is supported in part by the U.S. Department of Energy under contract number DE-AC02-07CH11359. I thank my DØ and CDF colleagues for extensive work on the applications of the advanced analysis methods over the years. My special thanks go to Harrison Prosper, Serban Protopopescu, Mark Strovink and Scott Snyder for delightful collaboration on the early multivariate analyses at DØ. I also thank Bob Cousins, Paul Grannis, Dan Green, Rob Roser and Harrison Prosper for reading this manuscript and providing very useful feedback. I thank Chandra Bhat and Shreyas Bhat for useful discussions and comments and Jenna Caymaz for preparing schematic diagrams in **Figures 3** (a) and **5** (a), (b).

LITERATURE CITED

1. Glashow S. *Nucl. Phys.* 22:579 (1961)
2. Weinberg S. *Phys. Rev. Lett.* 19:1264 (1967)
3. Salam A. *Elementary Particle Physics,* Almquvist and Wiksells, Stockholm, (1968)
4. Glashow S, Iliopoulos J, and Maiani L. *Phys. Rev.* D2:1285 (1970)
5. Gross D, Wilczek F. *Phys. Rev.* D8: 3633 (1973); *ibid., Phys. Rev. Lett.* 30:1343 (1973)
6. Politzer HD. *Phys. Rev. Lett.* 30:1346 (1973)
7. CDF Collaboration (Abe F, et al.), *Phys. Rev. Lett.* 74:2626 (1995); CDF Collaboration (Abe F, *et al.*), Phys. Rev. D50:2966 (1994)
8. DØ Collaboration (Abachi S, et al.), *Phys. Rev. Lett.* 74:2632 (1995)
9. DONUT Collaboration (Kodama K, et al.), *Phys.Lett.*B504:218 (2001)
10. Higgs PW. *Phys. Lett.* 12:132 (1964)
11. Higgs PW. *Phys. Rev. Lett.* 13:508 (1964)
12. Guralnik GS, Hagen CR and Kibble TWB. *Phys. Rev. Lett.* 13:585 (1964)
13. Anderson PW. *Phys. Rev.* 130:439 (1963)
14. Wimpenny SJ, Winer BL. *Annu. Rev. Nucl. Part. Sci.* 46:149 (1996); Campagnari C, Franklin M. *Rev. Mod. Phys.* 69:137(1997); Bhat PC, Prosper HB, Snyder SS. *Int. J. Mod. Phys. A* 13: 5113 (1998)
15. Bhat PC, Spalding WJ. *Proc. 15th Topical Conf. on Hadron Collider Physics*, East Lansing, Michigan, 2004, *AIP Conf.Proc.*753:30 (2005)
16. DØ Collaboration (Abazov VM et al.), *Phys. Rev. Lett.* 103:092001 (2009)
17. CDF Collaboration (Aaltonen T et al.), *Phys. Rev. Lett.* 103:092002 (2009)

18. *The Large Hadron Collider*, LHC Design Report, v.3 CERN-2004-003-V-3. - Geneva : CERN, 356 p (2004)

19. Denby B. *Comp. Phys. Comm.* 49:429 (1988); Denby B, et al., FERMILAB-CONF-92-269-E (1992)

20. Lönnblad L, Peterson C, Rögnvaldsson T. *Phys. Rev. Lett*. 65:1321 (1990)

21. Bhat PC, Lonnblad L, Meier K, Sugano K. *Research Directions for the Decade: Proc. of 1990 Summer Study on High Energy Physics*: Snowmass, CO, p. 168 (1990)

22. Peterson C, in *Proc. Computing in High Energy Physics* (CHEP '92), Annecy, France, CERN 92-07 (1992) and references therein; Also see, http://neuralnets.web.cern.ch/NeuralNets/nnwInHepRefSoft.html

23. Bhat PC (for the DØ Collaboration), *Proc. Meeting of the American Physical Society, Division of Particles and Fields,* Albuquerque, NM, p. 705 (1994); *ibid Proc. pbar-p Collider Workshop 1995, AIP Conf. Proc*.357:308, (1996)

24. G. Moneti (CLEO Collaboration), *Nuclear Physics B (Proc. Suppl.)* 59:17 (1997)

25. DØ Collaboration (Abbott B, et al.), *Phys. Rev. Lett.* 79:1197 (1997); *ibid Phys.Rev.* D58: 052001 (1998); Bhat PC et al. DØ Note 3061, unpublished (1997)

26. DØ Collaboration (Abbott B, et al.), *Phys. Rev. Lett.* 79:4321 (1997); *ibid Phys. Rev. Lett.,* 80:2051 (1998)

27. DØ Collaboration (Abbott B, et al.), *Phys. Rev. Lett*. 83:1908 (1999)

28. Bhat PC. *Proc. Mtg. American Physical Society, Division of Particles and Fields,* Columbus, OH, 2000, *Int. J. Mod. Phys*.A16S1C:1122 (2001); *Proc. Int. Workshop on Adv. Comp. Anal. Tech, in Phys. Res.* (ACAT 2000), Batavia, IL, 2000, *AIP Conf. Proc*. 583, p. 22 (2001)

29. See e.g., McNamara PA III, Wu SL. *Rep. Prog. Phys*. 65:465, (2002), and references therein.

30. Fisher R. *Annals of Eugenics*, 7:179 (1936)

31. Rosenblatt F. *Psych. Rev.* 65:386 (1958)

32. Levenberg K. *The Qtrly. Appl. Math.* 2:164 (1944); Marquardt D. *SIAM J. Appl. Math.* 11:431 (1963)

33. Kirkpatrick S, Gelatt CD, Vecchi MP. *Science*. *New Series* 220 (4598):671 (1983)

34. Goldberg DG. *Genetic Algorithms in Search Optimization and Machine Learning.* Addison Wesley, (1989)

35. O'Hagan A. *Kendall's Advance Theory of Statistics: Volume 2B, Bayesian Inference*, Oxford University Press, New York (2002)

36. Neyman J, Pearson E. *Philosophical Transactions o the Royal Society of London, Series A,. Series A,* **231**: 289 (1933)

37. Bishop CM. *Neural Networks for Pattern Recognition*, Oxford University Press (1995);

38. Bishop CM, *Pattern Recognition and Machine Learning*, pp. 738. New York: Springer Science+Business Media (2007)

39. Vapnik VN. *Statistical Learning Theory*, York: Springer-Verlag (2000)

40. Duda RD, Hart PE, Stork DG. *Pattern Recognition*, pp. 654. United States of America: Wiley Interscience (2000)

41. Hastie T, Tibshirani R, Friedman JH. *The elements of statistical learning : Data mining, inference, and prediction*, pp. 533, New York: Springer Verlag (2001)

42. Amos NA, et al. *Proc. Int. Conf. Computing in High Energy Physics* (CHEP 95), Rio de Janeiro, Brazil, p. 215 (1995)

43. Engelman R, et al. *Nucl. Inst. Meth*. 216:45 (1983); Raja R, DØ Note 1192, unpublished (1991)

44. Holmstrom L, Sain R, Miettinen HE. *Comput. Phys. Commun*. 88:195 (1995); DØ Collaboration, Phys.Rev.D60:052001 (1999).

45. Neal RM. *Bayesian Learning for Neural Networks* (Lecture Notes in Statistics), New York: Springer Verlag (1996)

46. Bhat PC, Prosper HB. *Proc. PHYSTATO5: Statistical Problems in Particle Physics, Astrophysics and Cosmology*, Oxford, England, UK, Imperial College Press, p. 151 (2005); Prosper HB, *Proc. Adv. Comput. Anal. Tech.*, Erice, Italy, *PoS (ACAT08) 010* (2008)

47. Brieman L, Friedman J, Olshen R, Stone C. *Classification and Regression Trees*, Wadsworth, (1984)

48. Roe BP et al. *Nucl. Inst. Meth. in Physics Research A* 543:577(2005)

49. Brieman L, *Machine Learning*, 26:123(1996); *ibid* 45:5 (2001)

50. Freund Y, Schapire RE. *J. Comput. Sys. Sci.* 55:119 (1997); Friedman J, Hastie T, Tibshirani R. *Annals of Stat.* 28:337 (2000)

51. Yao X. *Proc. IEEE* 87:1423 (1999)

52. DØ Collaboration (Abazov VM, et al.) *Nature* 429:638 (2004).

53. Peterson C, Rögnvaldsson T. *JETNET 3.0—A Versatile Artifical Neural Network Package*, Rep. No. CERNTH. 7135/94 (1994)

54. Feindt M, Kerzel U *Nucl. Instrum. Methods in Phys. Res. Sect. A* 559:190 (2006)

55. Stanley KO, Miikkulainen R *Evolutionary Comp.* 10:99 (2002)

56. Friedman JH, Popescu BE *Annals of Applied Statistics*, 2:916 (2008)

57. Hoecker A, et al., *TMVA - Toolkit for Multivariate Data Analysis*, PoS ACAT 040 (2007); arXiv:Physics/0703039 (2007)

58. Narsky I. arXiv:physics/0507143v1 (2005)

59. Bhat PC, Prosper HB, Snyder SS. *Phys. Lett.* B407:73 (1997)

60. The Tevatron Electroweak Group, http://arxiv.org/pdf/1007:3178 (2010)

61. DØ Collaboration (V.M. Abazov, et al.) *Phys. Rev. Lett.*103:092001 (2009); *ibid*, *Phys. Rev.* D78:012005 (2008)

62. CDF collaboration (Aaltonen T et al.) *Phys. Rev. Lett.* 103:092002 (2009)

63. Harris BW et al. *Phys. Rev. D.* 66:054024 (2002)

64. Heinson A, Junk T, *Ann. Rev. Nucl. Part. Sci.* 61:xxx (2011)

65. The ALEPH, DELPHI, L3, OPAL Collaborations and the LEP Higgs Working Group (Barate et al.), *Phys. Lett*. B565:61 (2003)

66. ALEPH Collaboration (Barate R et al.) *Phys. Lett.* 495:1 (2000); DELPHI Collaboration, (Abreu P) *Eur. Phys. J. C*17:187 (2000); OPAL Collaboration, (Abbiendi G) *Phys. Lett. B*499:38 (2001); L3 Collaboration, (Achard P) *Phys. Lett. B*517:319 (2001)

67. Kado MM, Tully CG. *Annu. Rev. Nucl. Part. Sci*. 52:65 (2002)

68. Bhat PC, Gilmartin R, Prosper HB. *Phys. Rev. D*62:074022 (2000)

69. Carena M et al., e-Print: hep-ph/0010338 (2000)

70. DØ Collaboration (Abazov VM et al.) *Nucl. Instrum. Methods Phys. Res. A* 620:490 (2010)

71. Scanlon T. *Ph.D. Thesis*, Imperial College, London, UK (2006)

72. CDF Collaboration (Aaltonen T et al.) *Phys. Rev. Lett.* 105:251802 (2010)

73. CDF and the DØ Collaborations (Aaltonen T et al.) *Phys. Rev. Lett*. 104:061802 (2010)

74. CDF Collaboration (Aaltonen T et al.), *Phys. Rev. Lett*. 104:061803 (2010)

75. DØ Collaboration (Abazov VM et al.) *Phys. Rev. Lett*. 104:061804 (2010), http://www-d0.fnal.gov/ DØ Note 6008-CONF (2009)

76. NNPDF Collaboration, (Ball RD, et al.) *Nucl. Phys. B* 809:1 (2009)

77. Neal R, *Proc.PHYSTAT LHC Workshop on Statistical Issues for LHC Physics*, CERN, Switzerland, p. 111 (2007).

78. Straessner A, *Springer Tracts Mod. Phys.* 235:1-211 (2010)

79. Parker GJ, *Springer Tracts Mod. Phys.* 236:1-170 (2010)

80. Kiesling C et al. *Proc. Int. Workshop on Adv. Comp. Anal. Tech, in Phys. Res.* (ACAT 2000), Batavia, IL, 2000, *AIP Conf. Proc*. 583, p. 36 (2001)

81. Babar Collaboration, (Aubert B, et al.) *Phys. Rev. Lett.* 99:021603 (2007); *ibid*, 87:091801 (2001)

82. Shaevitz MH, et al. (for MiniBooNE Collaboration) *J. Phys.: Conf. Ser*. 120:052003 (2008); Yang HJ, Roe BP, Zhu J. *Nucl. Inst. Meth A*555:370 (2005); MINOS Collaboration (Adamson P et al.) *Phys.Rev.Lett*. 103:261802 (2009)