MULTIVARIATE ANALYSIS AND THE SEARCH FOR NEW PARTICLES* **

Kyle Cranmer

University of Wisconsin
1150 University Av., Madison WI 53 706, USA

(Received October 7, 2003)

A general setting is presented for the evaluation of several common multivariate analysis techniques including neural networks, support vector machines, and genetic programming. Various theoretical results from pure mathematics and statistical learning theory are presented. Special attention is placed on the optimization criterion for the search for new particles.

PACS numbers: 14.80.Bn, 02.40.–k, 02.60.–x, 07.05.Mh

1. Introduction

With the rise of computing power, physicists are employing numerical techniques to transcend those calculations which were intractable analytically. The empowerment gained in the predictive half of physics is concomitant with the revolution in data analysis. Corresponding to the advances in theory, such as Lattice Gauge Theory and the Monte Carlo techniques used in particle physics, are the advances in data analysis, such as neural networks and maximum likelihood techniques.

Ironically, the complexity which make these algorithms so powerful also makes them difficult to understand. Multivariate Algorithms (MAs), which began as pure mathematics, have flourished under the high art of engineering. Unfortunately, the process has obfuscated the fundamental theory of MAs. The recent advent of statistical learning theory has returned to the fundamentals and motivated a significant reconsideration of common techniques. In this note I will attempt to reconsider the use of multivariate analysis in the search for new particles using the tools of statistical learning

* Presented at the Workshop on Applications of Neural Networks, Zakopane, Poland, May 30–June 8, 2003.

** This work was supported by a graduate research fellowship from the National Science Foundation and US Department of Energy Grant DE-FG0295-ER40896.

theory. To provide context, several approaches to multivariate analysis will be reviewed and specific algorithms will be introduced.

The goals of this text and the corresponding presentation are:

1. to provide an overview for MAs suited for theoretical physicists,
2. to establish a general and coherent formalism for MAs,
3. to distinguish the different applications of MAs,
4. to clarify the relationship between the application and the appropriate notion of performance of the algorithm,
5. to introduce the different settings in which MAs have been developed.

2. Formalism

Formally a Learning Machine is a family of functions $\mathcal{F}$ with domain I and range O parametrized by $\alpha \in \Lambda$. The domain can usually be thought of as, or at least embedded in, $\mathbb{R}^d$ and we generically denote points in the domain as x. The points x can be referred to in many ways (*e.g.* patterns, events, inputs, examples, ...). The range is most commonly $\mathbb{R}$, $[0,1]$, or just $\{0,1\}$. Elements of the range are denoted by y and can be referred to in many ways (*e.g.* classes, target values, outputs, ...). The parameters α specify a particular function $f_\alpha \in \mathcal{F}$ and the structure of $\alpha \in \Lambda$ depends upon the learning machine [1, 2].

Typically, the use of a learning machine is broken into three phases: a training phase, a testing phase, and a processing phase. The training phase is responsible for choosing a particular $\alpha \in \Lambda$, an independent testing phase is used to assess the performance of the resulting f_α, and the processing phase represents the intended use of f_α.

In the training phase, one has some training data in the form of pairs $(x,y)_i$. These training data are presented to the learning machine and a *learning algorithm* adjusts the parameters α as to maximize some notion of performance or, equivalently, minimize some notion of *risk*. This model of learning is called *supervised learning*, because the target values y_i are known.

The pairs $(x,y)_i$ are example associations collected through experience or derived from a theoretical model. As a toy example, consider x to contain the height and age of individuals and y to be their weight. What is important to realize is that the associations between x and y can be probabilistic and summarized by a joint probability distribution $p(x,y)$. For instance, two different individuals may have the same height and age x but different weights $y_i \neq y_j$. Thus, while the training data include both (x,y_i) and

(x, y_j), the learning machine will never be able to satisfy both requirements simultaneously.

In the modern theory of machine learning, the performance of a learning machine is usually cast in the more pessimistic setting of *risk*. In general, the risk, R, of a learning machine is written as

$$R(\alpha) = \int L(y, f_\alpha(x))\ p(x,y)dxdy = \int Q(x,y;\alpha)\ p(x,y)dxdy\,, \qquad (1)$$

where L measures some notion of *loss* between $f_\alpha(x)$ and the target value y. Often $L(y, f_\alpha(x))$ is written in the more compact form $Q(x, y; \alpha)$. As we will see in the next section, most of the classic applications of learning machines can be cast into this formalism; however, searches for new particles place some strain on the notion of risk.

To clarify the nomenclature, I will attempt to refer to the abstract family of functions $\mathcal{F}$ as a learning machine, a pair $(\mathcal{F}, Q)$ as a multivariate algorithm, and a particular function f_α as a function or (with a slight abuse) a trained learning machine.

3. Applications of multivariate algorithms

Rarely does one encounter a task that can be fully encapsulated by a multivariate algorithm. Instead, it is much more common that a multivariate algorithm performs an intermediate task and both its input (output) incur (result from) external processing. What is often overlooked is that the goal of the multivariate algorithm may not be the most appropriate for the task at hand. This problem is exacerbated by the fact that operationally the use of different algorithms may be quite similar or even indistinguishable.

Consider a common post-processing step using learning machines with a range $[0, 1]$ in which the function f_α is composed with a step function $\Theta(y - y_0)$[1]. In this case the resulting function $\tilde{f}(x) = \Theta(f_\alpha(x) - y_0)$ has a range $\{0, 1\}$ and is operationally identical to a learning machine $g_\beta(x)$ with binary output. The subtlety is that f_α may have been trained as to minimize a different risk functional that g_β and the resulting partition of the domain will not coincide in general.

3.1. Classification

Classification was one of the first applications of multivariate algorithms. This task was addressed by Fisher in the 1930's using discriminant analysis. Fisher's approach assumed knowledge of the probability distribution

[1] This is typically the case with neural networks and y_0 is called the "cut" on the neural network output.

functions for the two categories. In the 1960's Rosenblatt introduced the perceptron, a hallmark of both neural networks and learning theory in general.

Classification can be thought of as simply partitioning the domain into categories. The most simple case is binary classification in which $y = 0$ represents one class and $y = 1$ represents the other. For classification, only learning machines with range $\{0, 1\}$ are considered[2]. In that case, the relevant notion of risk is the rate of misclassification and thus,

$$Q(x, y; \alpha) = |y - f_\alpha(x)| \,. \tag{2}$$

3.2. Regression & prediction

Regression is the most common setting of selecting a particular function $f(x; \alpha_0)$ from a parametrized set of options $f(x; \alpha)$ based on empirical data. Regression is often solved within the context of Maximal Likelihood and substantial progress has been made for complex problems with the use of the EM algorithm [3]. There are a number of techniques, but with standard assumptions the solution to the problem of regression is the familiar least-squares method. That is the motivation for the loss functional

$$Q(x, y; \alpha) = (y - f_\alpha(x))^2 \tag{3}$$

used in regression problems.

Regression is different from classification, most notably because the range is continuous, as opposed to binary. While neural networks are not often referred to as regression techniques, they usually attempt to minimize some error functional equivalent to regression risk functional.

3.3. Searches for new particles

The search for a new particle is a practical application of multivariate algorithms, and one with an intuitive notion of performance. Unfortunately, the notion of performance is so nebulous that many of the "optimizations" performed in the analysis chain are irrelevant to the final conclusion. The conclusion of this type of experiment is a statistical statement — usually a declaration of discovery or a limit on the mass of the hypothetical particle. Thus, the appropriate notion of performance for a multivariate algorithm used in a search for a new particle is that performance measure which will maximize the chance of declaring a discovery or provide the tightest limits on the hypothetical particle.

[2] Sometimes the classes are considered as $\{-1, 1\}$ so that functions with range $\mathbb{R}$ can be implicitly composed with the sign () function.

In principle, it should be a fairly straight-forward procedure to use the formal statistical statements to derive the most appropriate performance measure. The first difficulty in this process is that there are many interesting statistical statements from which to choose. To make matters worse, experimentalists and statisticians cannot even settle on a formalism to use. Within the statistics literature there are two factions commonly referred to as "Bayesians" and "Frequentists". While both formalisms adhere to Kolmogorov's axioms for a probability measure, they disagree on how to define probability. In short, Bayesians define probability to be *a degree of belief* while Frequentist define probability to be *a limiting frequency*. Avoiding a detailed discussion, it should be known that Bayesian methods are considered to be powerful because they can incorporate prior knowledge in a more natural way, but they are also criticized as being subjective due to the introduction of *a priori* probability for the physics parameters being studied by the experiment. This philosophical divide is deep, and recently being confronted by experimental physicists.

As an example, let us consider the Frequentist theory developed by Neyman and Pearson. This was the basis for the results of the search for the Standard Model Higgs boson at LEP [4].

4. The Neyman–Pearson theory

The Neyman–Pearson theory [5] begins with two Hypotheses: the null hypothesis H_0 and the alternate hypothesis H_1. In the case of a new particle search H_0 is identified with the currently accepted theory (*i.e.* the Standard Model) and is usually referred to as the "background-only" hypothesis. Similarly, H_1 is identified with the theory being tested (*i.e.* Standard Model with Higgs boson at some specified mass m_H) usually referred to as the "signal-plus-background" hypothesis[3]. With these two hypotheses one is able, through theory, to describe the probability distribution of physical observables $x \in I$ written as $p(x|H_0)$ and $p(x|H_1)$. Next, one defines a region $W \in I$ such that if the data fall in W we accept the null hypothesis (and reject the alternate hypothesis)[4]. Similarly, if the data fall in $I - W$ we reject the null hypothesis and accept the alternate hypothesis. Recognize that if the null hypothesis is true, then there exists a chance that the data could fall in $I - W$ and we reject H_0 even though it is true — we commit a *Type I error*. The probability to commit a Type I error is called the *size* of

[3] An attentive reader might question the meaning of the Standard Model without the presence of the Higgs boson. Furthermore, one must be careful that additional particles are included in a way consistent with quantum mechanics and not blindly resort to the addition of probabilities.

[4] With m measurements, we should actually consider the data as $(x_1, \ldots, x_m) \in I^m$, but, for ease of notation, let us only consider $m = 1$.

the test and is given by (note alternate use of α)

$$\alpha = \int_{I-W} p(x|H_0)dx\,. \tag{4}$$

Similarly, if the alternate hypothesis is true the data could fall in W, in which case we accept H_0 even though it is false — we commit a *Type II error*. The probability to commit a Type II error is given by

$$\beta = \int_{W} p(x|H_1)dx\,. \tag{5}$$

Also of importance is the notion of *power* $= 1-\beta$, which can be interpreted as the chance that one accepts H_1 when it is true.

In particle physics, the discovery criterion is often referred to as the 5σ requirement. This requirement is related to the probability of Type I error and, depending on convention, corresponds to $\alpha = 5.8 \times 10^{-7}$ or $\alpha = 2.9 \times 10^{-7}$. Thus, what particle physics control with the 5σ requirement is the rate of false discovery.

The central result of the Neyman–Pearson theory is the Neyman–Pearson Lemma, which tells us how to chose an acceptance region W. The Neyman–Pearson Lemma states that holding α fixed, the region W that maximizes the power is bounded by a contour of the Likelihood ratio

$$W = \left\{ x \;\middle|\; \frac{p(x|H_1)}{p(x|H_0)} > k_\alpha \right\}, \tag{6}$$

where k_α is a constant chosen to satisfy equation (4).

Once one specifies the size, α, of the test the power of the test is determined from $p(x|H_0)$ and $p(x|H_1)$. How one chooses the size of the test, however, transcends the Neyman–Pearson theory. Typically, scientists retreat to conventional values such as $\alpha = 0.05$ (which corresponds to a 95% confidence) or 5σ in the case of particle physics. These choices are essentially arbitrary, but that need not be the case. It is possible that one can define a utility function $U(\alpha, \beta)$ and optimize the utility as a function of α (remembering that β can be determined from α with $p(x|H_0)$ and $p(x|H_1)$ fixed).

It is worth emphasizing the role of the alternate hypothesis H_1. Prior to the Neyman–Pearson theory, Fisher had developed a similar theory based on what he called a *pure significance test*. The test only considered one hypothesis H_0, in contrast to the Neyman–Pearson theory. The expected value of Fisher's p-value has the same form as α; however, there is no unique

notion of W. There could be infinitely many W with the same size, but Fisher left the choice of W to the experimenter. In the Neyman–Pearson theory, this symmetry is broken by H_1. In this context it is easy to see how fundamentally different the typical search scenario is from the model-independent searches that have been suggested recently [6].

4.1. The Neyman–Pearson theory in the context of risk

In Section 3 we provided the loss functional appropriate for the classification and regression tasks; however, we did not provide a loss functional for searches for new particles. The first reason for delaying the presentation of the loss functional was the lack of consensus within the experimental community on a statistical formalism. Having chosen the Neyman–Pearson theory as an explicit example, it is possible to develop a formal notion of risk.

Once the size of the test, α, has been agreed upon, the notion of risk is the probability of Type II error β. In order to return to the formalism outlined in Section 2, identify H_1 with $y = 1$ and H_0 with $y = 0$. Let us consider learning machines that have a range $\mathbb{R}$ which we will compose with a step function $\tilde{f}(x) = \Theta(f_\alpha(x) - k_\alpha)$ so that by adjusting k_α we insure that the acceptance region W has the appropriate size. The region W is the acceptance region for H_0, thus it corresponds to $W = \{x|\tilde{f}(x) = 0\}$ and $I - W = \{x|\tilde{f}(x) = 1\}$. We can also translate the quantities $p(x|H_0)$ and $p(x|H_1)$ into their learning-theory equivalents $p(x|0) = p(x,0)/p(0) = \delta(y)p(x,y)/\int p(x,0)dx$ and $\delta(1-y)p(x,y)/\int p(x,1)dx$, respectively. With these substitutions we can rewrite the Neyman–Pearson theory as follows. A fixed size gives us the global constraint

$$\alpha = \frac{\int \Theta(f_\alpha(x) - k_\alpha)\ \delta(y)\ p(x,y))dxdy}{\int p(x,0)dx} \tag{7}$$

and the risk is given by

$$\begin{aligned} \beta &= \int\limits_W p(x|H_1)dx \\ &= \frac{\int[1 - \Theta(f_\alpha(x) - k_\alpha)]\ p(x,1)dx}{\int p(x,1)dx} \\ &\propto \int \Theta(f_\alpha(x) + k_\alpha)\delta(1-y)p(x,y)dxdy\,. \end{aligned} \tag{8}$$

Extracting the integrand we can write the loss functional as

$$Q(x,y;\alpha) = \Theta(f_\alpha(x) + k_\alpha)\delta(1-y)\,. \tag{9}$$

Unfortunately, equation (1) does not allow for the global constraint imposed by k_α (which is implicitly a functional of f_α), but this could be accommodated by the methods of Euler and Lagrange.

4.2. The case of no interference

When there is no (or negligible) interference between the signal process and the background processes one can avoid the complications imposed by quantum mechanics and simply add probabilities. This is often the case with searches for new particles. For instance, the width of a Higgs boson with mass less than about 300 GeV/c^2 is so narrow that the interference with standard model processes is usually considered negligible. In that case the signal-plus-background hypothesis can be rewritten $p(x,|H_1) = n_{\rm s}p_{\rm s}(x) + n_0 p(x|H_0)$, where $n_{\rm s}$ and n_0 are normalization constants that sum to unity. Thus, the Likelihood Ratio can be rewritten

$$\frac{p(x|H_1)}{p(x|H_0)} = \frac{n_{\rm s}p_{\rm s}(x) + n_0 p(x|H_0)}{p(x|H_0)} \tag{10}$$

and the contours of the likelihood ratio $\frac{p(x|H_1)}{p(x|H_0)} = k_\alpha$ can be simplified to $\frac{p_{\rm s}(x)}{p(x|H_0)} = (k_\alpha - n_0)/n_{\rm s} = k'_\alpha$.

4.3. Indirect methods

The loss functional defined in equation (9) is derived from a minimization on the rate of Type II error. This is logically distinct from, but asymptotically equivalent to, approximating the Likelihood Ratio. In the case of no interference, this is logically distinct from, but asymptotically equivalent to, approximating the signal to background ratio. As we will see most multivariate algorithms are concerned with approximating an auxiliary function that is one-to-one with the Likelihood Ratio. Because the methods are not directly concerned with minimizing the rate of Type II error, they should be considered indirect methods. Furthermore, the asymptotic equivalence breaks down in most applications, and the indirect methods are no longer optimal. Neural Networks, Kernel Estimation Techniques, and Support Vector Machines all represent indirect solutions to the search for new particles. The Genetic Programming approach is the only direct approach considered in this note.

5. Different approaches to multivariate analysis

For those familiar with multivariate analysis, the presentation given in this text may seem unfamiliar. The field of multivariate analysis is so large and so heterogeneous that disciples of one approach may not understand the formalism of another. In this text, multivariate analysis has been cast in the formalism of Statistical Learning Theory (or Machine Learning) due largely to Vapnik. The different approaches, some might say paradigms, of multivariate analysis can be roughly grouped into three different classes: statistical approaches, machine learning, and information theoretical approaches. Recently there have been a number of methods which cross these boundaries and connect with very deep branches of mathematics. The differences between these approaches is not simply their formalism, but their emphasis. Statistical approaches are primarily based on asymptotic properties (*i.e.* in the limit of infinite training data, $l \to \infty$), machine learning stresses the lack of these asymptotic properties for discrete data sets, and the information theoretical approaches lie somewhere in between with a distinctly intuitive feel.

5.1. Statistical approaches

Statistical approaches to learning theory leverage the many powerful theorems of statistics assuming one can explicitly refer to $p(x, y)$, the joint probability distribution function of the input–output pairs. This dependence on $p(x, y)$ places a great deal of stress on the asymptotic ability to estimate $p(x, y)$ from a finite set of samples $\{(x, y)_i\}$. There are many such techniques for estimating a multivariate density function $p(x, y)$ given the samples [7,8]. Unfortunately, for high dimensional domains, the number of samples needed to enjoy the asymptotic properties grows very rapidly; this is known as the *curse of dimensionality*. The kernel estimation techniques described in Section 7.3 represent a particular statistical approach.

5.2. Machine learning

The starting point for machine learning is to accept that we might not know $p(x, y)$ in any analytic or numerical form. This is, indeed, the case for particle physics, because only $\{(x, y)_i\}$ can be obtained from the Monte Carlo convolution of a well-known theoretical prediction and complex numerical description of the detector. In this case, the learning problem is based entirely on the training samples $\{(x, y)_i\}$ with l elements. The risk functional is thus replaced by the *empirical risk functional*

$$R_{\text{emp}}(\alpha) = \frac{1}{l} \sum_{i=1}^{l} Q(x_i, y_i; \alpha) \,. \tag{11}$$

One then must try to approximate $f_{\alpha_0} \in \mathcal{F}$, that minimizes the true risk, by the function f_{α_l}, that minimizes the empirical risk. This is approach is called the *empirical risk minimization* (ERM) inductive principle.

Vapnik outlines the four parts of learning theory in [2]:

(1) What are the (necessary and sufficient) conditions for consistency of a learning process based on the ERM principle?

(2) How fast is the rate of convergence of the learning process?

(3) How can one control the rate of convergence (the generalization ability) of the learning process?

(4) How can one construct algorithms that can control the generalization ability?

Answering question (1) is achieved by considering the notion of non-trivial consistency. The details of the discussion are beyond the scope of this note, but consistency is essentially a guarantee that with an infinite amount of training data ($l \to \infty$) the ERM principle will produce a function with equal risk to f_{α_0}. Interestingly, the necessary and sufficient conditions for non-trivial consistency are analogous to Popper's theory of non-falsifiability in the philosophy of science. In particular, Vapnik introduces a quantity h which is a property of a learning machine $\mathcal{F}$ and called the Vapnik–Chervonenkis (VC) dimension. Simply put, the conditions for (1) are that h is finite.

The VC dimension of $\mathcal{F}$ is defined as the maximal cardinality of a set which can be shattered by $\mathcal{F}$. "A set $\{x_i\}$ can be shattered by $\mathcal{F}$" means that for each of the 2^h binary classifications of the points $\{x_i\}$, there exists a $f_\alpha \in \mathcal{F}$ which satisfies $y_i = f_\alpha(x_i)$. A set of three points can be shattered by an oriented line as illustrated in figure 1. Note that for a learning machine with VC dimension h, not every set of h elements must be shattered by $\mathcal{F}$, but at least one.

The answer to question (2) is the surprising result that there are bounds on the true risk $R(\alpha)$, which are independent of the distribution $p(x, y)$. In particular, for $0 \leq Q(x, y; \alpha) \leq 1$

$$\begin{aligned} R(\alpha) &= \int Q(x, y; \alpha)\ p(x, y) dx dy \\ &\leq R_{\text{emp}}(\alpha) + \sqrt{\left(\frac{h(\log(2l/h) + 1) - \log(\eta/4)}{l}\right)}, \end{aligned} \tag{12}$$

where η is the probability that the bound is violated. As $\eta \to 0$, $h \to \infty$, or $l \to 0$ the bound becomes trivial. In figure 2 the shape of the second term in the bound, known as the VC confidence, is shown.

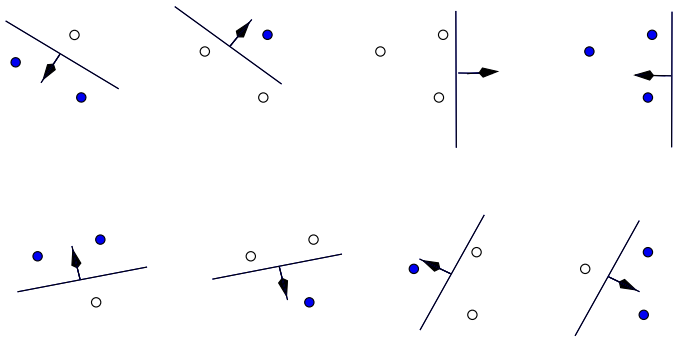


Fig. 1. Example of an oriented line shattering 3 points. Solid and empty dots represent the two classes for y and each of the 2^3 permutations are shown.

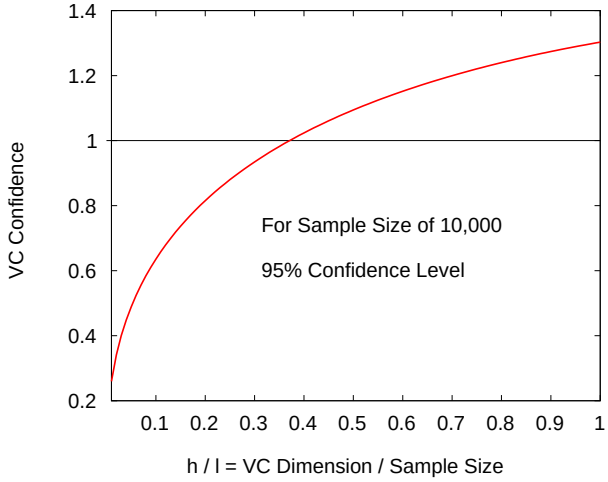


Fig. 2. The VC Confidence as a function of h/l for $l = 10,000$ and $\eta = 0.05$. Note that for $l < 3h$ the bound is non-trivial and for $l < 20h$ is quite tight.

Equation (12) is a remarkable result which relates the number of training examples l, the fundamental property of the learning machine h, and the risk R independent of the unknown distribution $p(x, y)$. The bounds provided by equation (12) are relatively weak due to their stunning generality. More important than their weakness, is the realization that with an independent testing sample one can evaluate the true risk arbitrarily well. This testing sample, by definition, is not known to the algorithm, so the bound is useful for the design of algorithms encountered in the 4^{th} part of Vapnik's theory. Neural Network and most other methods, however, rely on an independent testing sample to aid in their design.

5.3. Information theory

Information theory was born with Shannon's seminal paper *A Mathematical Theory of Communication* [9]. It would be difficult to over-estimate the profound impact this work had on the world and, in particular, the engineering fields. The key quantities that Shannon considered were the *a priori* probability p_i that a signal source could send a signal i and the entropy H of the source. The entropy defined as

$$H = -\sum_i p_i \log(p_i) \tag{13}$$

and the base of the logarithm defines the units of entropy (base 2 corresponds to bits).

It is beyond the scope of this note to discuss information theory in detail or the myriad of applications of Shannon's theory; however, it is worth pointing out a few common properties of information-based methods. First, they tend to be quite intuitive. For instance, when choosing variables x as input to a multivariate algorithm, those with the most *mutual information* with the targets y and the least mutual information among themselves should be used. Furthermore, leaving out variables or adding noise clearly results in information loss and should be avoided, or variables that are less sensitive to the noise should be chosen. Also, information theory has been used in the context of *unsupervised learning* in which the target values y are not known, but the network is still able to perform classification. Clearly unsupervised learning is very important for any biologically plausible model of cognition.

5.4. New directions

Clearly it is not possible to exhaustively discuss the new directions in multivariate analysis; however, let us consider two particularly examples that connect with deep fields of mathematics.

The first is the approach of Information Geometry due primarily to Amari. The idea is that one can consider a learning machine $\mathcal{F}$ as a manifold in which f_α (or just $\alpha \in \Lambda$) correspond to points, Λ corresponds to a coordinate system, and the metric tensor is provided by the Fisher information matrix

$$g_{ij}(\alpha) = \int dx f_\alpha(x) \left[\frac{\partial \log f_\alpha(x)}{\partial \alpha_i}\right] \left[\frac{\partial \log f_\alpha(x)}{\partial \alpha_j}\right]. \tag{14}$$

Geodesics on this manifold represent natural alternatives to gradient descent approaches and can result in exponentially faster rates of convergence [10].

The second approach is that of the *minimum description length* (MDL) principle. The MDL principle follows from the idea of algorithmic complexity. The work of Solomonoff, Kolmogorov, and Chaitin provided an

information theoretical link to the inductive approach of Vapnik. In [2], Vapnik shows that the MDL principle is roughly equivalent to his ERM principle, but more difficult in practice. It is interesting to see the theory of multivariate analysis touch so deeply with the fundamental limits of formal mathematics provided by Gödel, Turing, and Church. It is also interesting from an historical point of view that another of Hilbert's problems attacked by Kolmogorov is so deeply related to the theory of machine learning.

6. Motivation for neural networks

Neural Networks have a long history and their generality has given them utility in a broad range of fields. There are a number of theorems which describe the impact of the internal structure of a neural network on the class of functions to which it belongs. The most significant result is known as Kolmogorov's Superposition Theorem [11] which states:

THEOREM 1 (KOLMOGOROV'S SUPERPOSITION THEOREM)
For each $d \geq 2$ there exist continuous functions $\phi_q : [0,1] \to \mathbb{R}$, $q = 0, \ldots, 2d$ and constants $\lambda_p \in \mathbb{R}$, $p = 1, \ldots, d$ such that the following holds true: for each continuous function $F : [0,1]^d \to \mathbb{R}$ there exists a continuous function $g : [0,1] \to \mathbb{R}$ such that

$$F(x_1, \ldots, x_d) = \sum_{q=0}^{2d} g\left(\sum_{p=1}^{d} \lambda_p \phi_q(x_p)\right).$$

Note, ϕ_q and λ_p are independent of the represented function F.

Kolmogorov's paper, published in 1957, did not refer to neural networks directly; instead, it was in response to Hilbert's 13^{th} problem [12]. Exactly 30 years later Hecht-Nielsen noticed the application to the theory of neural networks [13]: each continuous function $F : [0,1]^n \to \mathbb{R}$ can be implemented by a feed-forward neural network with continuous activation functions t.

We do not expect, nor do we desire, a function which exactly categorizes our signal and background training sample – a behavior referred to as *over-training*. We know that there are regions of phase space for which either a signal or background event could occur. In such regions we do not wish for our neural network to fluctuate wildly between 0 and 1 to accommodate the training points. Instead, we desire an approximate solution which smoothly varies and has good generalization properties. Neural networks which use the so-called "sigmoid function" $t(x) = 1/(1 + \mathrm{e}^{-x})$ are known to posses good generalization properties and are the most common type for our application [14]. The upper-left image in figure 3 shows an example 7–10–10–1 architecture. The first column of nodes represent 7 input variables x_i, the

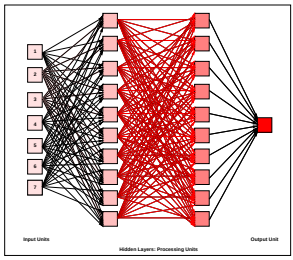
next two columns are the so-called hidden nodes which represent the bulk of the processing by calculating $t(W_{jk}x_k - \beta_j)$ (where W_{jk} is the j^{th} neuron's weight for the k^{th} node in the previous layer and β_j is the j^{th} neuron's bias), and the final node calculates a weighted sum of the penultimate layer's output.

What Kolmogorov and Hecht-Nielsen did not specify was how to find the weights W_{jk} and biases β_j given a function f_0 we wish to represent. Excluding neuroscience, the bulk of the literature focuses on so-called "learning algorithms" which attempt to find the optimal weights. The most widely used class of learning algorithm is called backpropagation, and is essentially a gradient-descent algorithm which aims to reduce an error function with respect to the network's weights [15]. The backpropogation algorithm is shown schematically in figure 3. The error function is usually the empirical risk functional 11 with the regression loss functional 3.

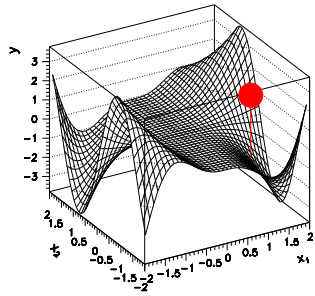
Because of their generalization properties, neural networks have become quite common within High Energy Physics — below we cite a few examples. They first appeared in the literature for their use in triggering and other online applications [16]. Quickly they were used for jet and track finding in an offline environment [17]. In addition to b-tagging, they were specifically used for Higgs searches in the early 1990s [18].

6.1. Overtraining

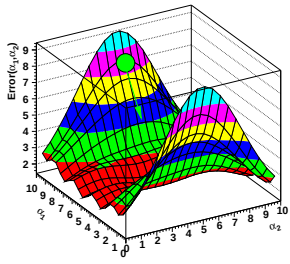
Because of neural networks remarkable representation capacity, they sometimes lose generalization performance. If a neural network can represent an incredibly complicated function, then it is in danger of learning the training samples. In that case, the empirical risk may be quite small, but the true risk may be large. Overtraining can be demonstrated by evaluating the risk with an independent testing sample. In figure 3, the bottom-right figure shows the evolution of the Error (or risk) as the network trains for three different architectures. It can be seen that the 3–20–1 architecture has such high capacity that around epoch 700, the true risk begins to grow. By construction, the backpropagation algorithm insures the empirical risk is a monotonically decreasing function, thus the network is loosing generalization performance. This is the phenomena of overtraining. For clarification, the training set may be presented to the network many times; the phenomena of overtraining is related to the repeated presentation of a fixed training set, *not* the addition of more training samples. This behavior is exactly what is described by equation (12).



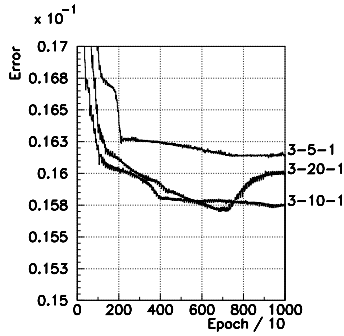
1) Start with some parameters α which correspond to some $f_\alpha(x)$



2) Calculate the Error $|y - f_\alpha(x)|$



3) Apply the Chain rule to find $\Delta\alpha$ of steepest decent



4) Repeat Many Times until Error is minimized

Fig. 3. A schematic of neural network training with backpropogation.

6.2. VC dimension of neural networks

In order to apply equation (12), one must determine the VC dimension of neural networks. This is a difficult problem in combinatorics and geometry aided by algebraic techniques. Eduardo Sontag has an excellent review of these techniques and shows that the VC dimension of neural networks can, thus far, only be bounded fairly weakly [19]. In particular, if we define ρ as the number of weights and biases in the network, then the best bounds are $\rho^2 < h < \rho^4$. In a typical particle physics neural network one can expect $100 < \rho < 1000$, which translates into a VC dimension as high as 10^{12}, which implies $l > 10^{13}$ for reasonable bounds on the risk. These bounds imply enormous numbers of training samples when compared to a typical training sample of 10^5. Sontag goes on to show that these shattered sets are incredibly special and that the set of all shattered sets of cardinality $\mu > 2\rho + 1$ is measure zero in general. Thus, perhaps a more relevant notion of the VC dimension of a neural network is given by μ.

7. Other multivariate methods

7.1. Support vector machines

The introduction of Support Vector Machines (SVM) is perhaps the most exciting development in machine learning in the last decade. Instead of trying to describe a non-linear acceptance region W in the input space I directly, SVMs use a non-linear map $\Phi(x)$ into a higher dimensional space and then perform a linear separation in that high dimensional space. The pull back of the acceptance region in the higher dimensional space can be quite non-linear due to the non-linearity of Φ. The benefits of this approach are three-fold. First, linear decision boundaries have simple properties, so the capacity of the learning machine can be controlled through Φ. Second, it is possible to form the optimization problem in the lower dimensional space by using a kernel $K(x_i, x_j)$, which implicitly describes the an inner product in the higher dimensional space specified by the non-linear map Φ. This technical restructuring of the problem allows for a computationally tractable inner product in the extremely high, possibly infinite, dimensional space. Thirdly, the solution to the optimization can be solved via quadratic programming techniques which insure a unique solution. This is very different from the case of neural networks, where the most serious practical problems are due to the presence of local minima and the non-uniqueness of the "solution". Lastly, because SVM are cast in the context of statistical learning theory, there are powerful results regarding consistency and risk.

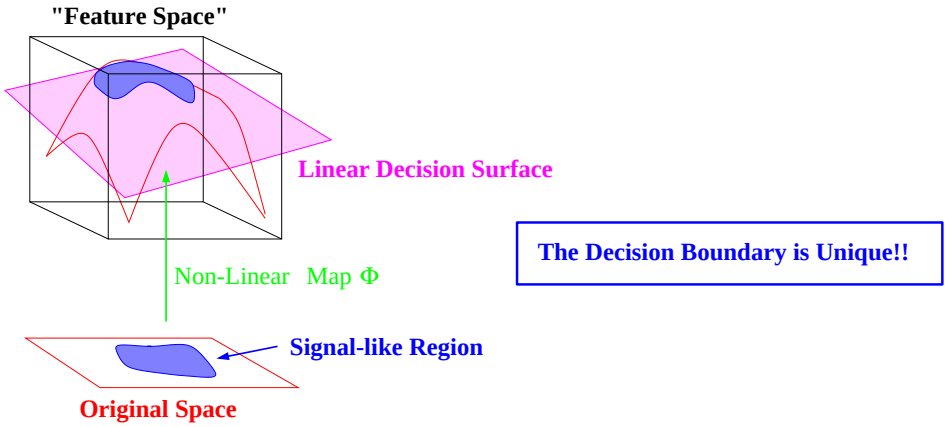


Fig. 4. A schematic representation of the implicit non-linear map Φ induced by the kernel $K(\cdot, \cdot)$.

7.2. Genetic programming

The use of Genetic Programming for the classification is fairly limited; however, it can be traced to the early works on the subject by Koza [20]. More recently, Kishore *et al.* extended Koza's work to the multicategory problem [21]. To the best of the authors' knowledge, the first application of Genetic Programming within particle physics will appear in [22].

In Genetic Programming (GP), a group of "individuals" evolve and compete with respect to a user-defined performance measure. The individuals represent potential solutions to the problem at hand, and evolution is the mechanism by which the algorithm optimizes the population. GP can be thought of as a stochastic sampling of a very high dimensional search space, where the sampling is related to the fitness evaluated in the previous generation (a Markov process), and stochastic perturbations to help avoid local extrema.

Genetic Programming is similar to, but distinct from, Genetic Algorithms (GAs), though both methods are based on a similar evolutionary metaphor. GAs evolve a bit string which typically encodes parameters to a pre-existing program, function, or class of cuts, while GP directly evolves the programs or functions. For example, Field and Kanev [23] used Genetic Algorithms to optimize the lower- and upper-bounds for six 1-dimensional cuts on Modified Fox–Wolfram "shape" variables. In that case, the phase-space region was a pre-defined 6-cube and the GA was simply evolving the parameters for the upper- and lower-bounds. On the other hand (*i.e.* an acceptance region W), GP algorithm is not constrained to a pre-defined shape or parametric form. Instead, the GP approach is concerned directly with the construction of an optimal, non-trivial phase space region with respect to a user-defined performance measure.

In the case at hand, the individuals that evolve are simple arithmetic expressions on the input variables. Without loss of generality, an event is classified as signal ($y = 1$) if the corresponding expression is evaluated to lie in the interval $(-1, 1)$. Furthermore, an individual may consist of one or more such cuts combined by the Boolean conjunctions `AND` and `OR`.

The genotype is an expression tree similar to an abstract syntax tree that might be generated by a compiler as an intermediate representation of a computer program. An example of such a tree is shown in figure 5. Leaves are either constants or one of the input variables. Nodes are simple arithmetic operators: addition, subtraction, multiplication, and safe division[5]. GPs which only produce polynomial expressions form a vector space, which allows for a quick approximation of their VC dimension [19].

[5] Safe division is used to avoid division by zero.

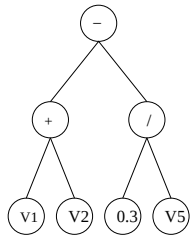


Fig. 5. The representation of an example expression $(V_1 + V_2) - (0.3/V_5)$.

7.3. Kernel estimation techniques

Kernel estimation (also known as probability estimation or density estimation) techniques belong to the class of statistical approaches to multivariate analysis. They can be thought of as the inverse of Monte Carlo techniques: from samples $\{x_i\}$ one attempts to reconstruct $p(x)$. A histogram is the simplest approach to density estimation, but more sophisticated methods have been developed [8]. The application of kernel estimation techniques to multivariate analysis is a three step process. In the first step, the probability densities $p_{\mathrm{s}}(x)$ and $p_{\mathrm{b}}(x)$ are constructed from signal and background training samples. Next, a discriminant function $D(x)$ is formed according to

$$D(x) = \frac{p_{\mathrm{s}}(x)}{p_{\mathrm{s}}(x) + p_{\mathrm{b}}(x)}\,. \tag{15}$$

Finally, events can be classified by composing $D(x)$ with a step function $\tilde{D}(x) = \Theta(D(x) - k''_\alpha)$. Through simple algebra $D(x)$ can be shown to be one-to-one with $p_{\mathrm{s}}(x)/p_{\mathrm{b}}(x)$ which is in turn one-to-one with the Likelihood Ratio so central to the Neyman–Pearson lemma. These correspondences are only valid asymptotically, and the ability to accurately approximate $p(x)$ from an empirical sample is often far from ideal. However, for particle physics applications, up to 5-dimensional multivariate analyses have shown good performance [24]. Furthermore, they have the added benefit that they can be easily understood.

8. Conclusions

Multivariate algorithms are obviously an extremely useful tool in data analysis. The more germane concern for physicists is what are the relevant properties of a multivariate algorithm for their particular application. In this note we have considered three common applications: classification, regression, and the search for new particles. For the three main approaches to multivariate analysis, we have distinguished between their asymptotic and non-asymptotic properties, established relationships among the approaches,

and presented the key theorems in their fundamental theories. Particular emphasis has been placed on the Neyman–Pearson setting for the interpretation of searches for new particles and the development of an appropriate notion of risk. We have considered several common multivariate algorithms and indicated their strengths and weaknesses. The final conclusions as to which multivariate algorithms are most appropriate for a given task will remain as much an experiment in human psychology as mathematical rigor.

REFERENCES

[1] V. Vapnik, A.J. Cervonenkis, The Uniform Convergence of Frequencies of the Appearance of Events to Their Probabilities, *Dokl. Akad. Nauk SSSR*, 1968, in Russian.

[2] V. Vapnik, *The Nature of Statistical Learning Theory*, 2nd edition Springer, New York 2000.

[3] Hartley, *Biometrics*, 1958, pp. 174–194; Demptster, Laird *et al.*, *JRSS*, B, 1977.

[4] LEP Higgs Working Group, Search for the Standard Model Higgs Boson at LEP, `LHWG Note/2002-01` (2002).

[5] J.K Stuart, A. Ord, S. Arnold, *Kendall's Advanced Theory of Statistics*, Vol. 2A, (6^{th} ed.), Oxford University Press, New York 1994.

[6] B. Abbott *et al.*, *Phys. Rev.* **D62**, 092004 (2000).

[7] D. Scott, *Multivariate Density Estimation: Theory, Practice, and Visualization*, John Wiley and Sons Inc., 1992.

[8] K. Cranmer, *Comput. Phys. Commun.* **136**, 198 (2001).

[9] C.E. Shannon, *Bell System Technical Journal* **27**, 623 (1948).

[10] Shun ichi Amari, *Neural Networks* **8**, 1379 (1995).

[11] A.N. Kolmogorov, *Dokl. Akad. Nauk USSR* **114**, 953 (1957) [translated in: *Am. Math. Soc. Transl.* **28**, 55 (1963)]; D. Sprecher, *Transactions American Mathematical Society* **115**, 340 (1965).

[12] D. Hilbert, *Nachrichten der Königlichen Gesellshaft der Wissenschaften zu Göttingen*, 1900, pp. 253–297.

[13] R. Hecht-Nielsen, *Proceedings IEEE International Conference On Neural Networks*, Vol. II, 1987, pp. 11–13; R. Hecht-Nielsen, *Neurocomputing*, Addison-Wesley, Reading 1990.

[14] G.G. Lorentz, *Approximation of Functions, Athena Series, Selected Topics in Mathematics*, Holt, Rinehardt and Winston, Inc., New York 1966.

[15] P.J. Werbos, *The Roots of Backpropagation*, John Wiley & Sons, New York 1974; H. Robbins, S. Monroe, *Ann. Math. Statistics* **22**, (1951); D.E. Rumelhart *et al.*, *Parallel Distributed Processing Explorations in the Microstructure of Cognition*, The MIT Press, Cambridge 1986; B.T. Polyak, *Z. Vycisl. Mat. i*

Mat. Fiz. **4**, 1 (1964); B.T. Polyak, *Introduction to Optimization. Optimization Software*, Inc., New York 1987.

[16] Denby, H. Bruce. Applications of Neural Networks and Cellular Automata in Experimental High-Energy Physics. In Trieste 1988, Proceedings, The Impact of Digital Microelectronics and Microprocessors on Particle Physics, pp. 150–153; C. Barter *et al.*, Neural Networks D0, and the SSC. Presented at the Workshop on Triggering and Data Acquisition for Experiments at the Supercollider, Toronto, Canada, January 1989; L. Lonnblad, C. Peterson, T. Rognvaldsson, *Phys. Rev. Lett.* **65**, 1321 (1990).

[17] L. Bellantoni *et al.*, *Nucl. Instrum. Meth.* **A310**, 618 (1991); T.D. Gottschalk, R. Nolty, Identification of Physics Processes Using Neural Network Classifiers, CALT-68-1680; B. Humpert, *Comput. Phys. Commun.* **56**, 299 (1990).

[18] D. Decamp *et al.*, *Phys. Lett.* **B265**, 475 (1991); D. Buskulic *et al.*, *Phys. Lett.* **B313**, 299 (1993); P. Chiappetta *et al.*, *Phys. Lett.* **B322**, 219 (1994); T. Maggipinto *et al.*, *Phys. Lett.* **B409**, 517 (1997).

[19] E. Sontag, *Neural Networks and Machine Learning*, ed. C.M. Bishop, Springer-Verlag, Berlin 1998, pp. 69–95.

[20] J.R. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection*, MIT Press, Cambridge, MA 1992.

[21] J.K. Kishore *et al.*, *IEEE Trans. Evolutionary Comput.* **4**, (2000).

[22] K. Cranmer, R.S. Bowman, PhysicsGP: A Genetic Programming Approach to Event Selection, submitted to *Comput. Phys. Commun.*

[23] R.D. Field, Y.A. Kanev, `hep-ph/9801318`.

[24] L. Hölmstrom *et al.*, *Comput. Phys. Commun.* **88**, 195 (1995).