1. Introduction

Inspired by the algorithmic notion of complexity, Solomonoff (1964), Kolmogorov (1965), Chaitin (1966), as well as Akaike's work, Akaike (1977), I some ten years ago proposed the shortest code length for the observed data as a criterion for model selection, Rissanen (1978), which in the subsequent papers, Rissanen (1983), (1984), (1986a), (1987), gradually evolved into stochastic complexity. The word "stochastic" was meant to suggest that the models, relative to which the coding ought to be done, were probabilistic rather than being defined by programs in a universal computer as in the algorithmic theory. Having presented the material to numerous audiences I frequently was asked the question to the effect that "why should we be interested in the code length as a measure of model's performance if the models are not used for coding purposes". Although this measure in itself has a strong intuitive appeal and its success can be supported both by applications and theoretical analysis, a deeper answer would clearly be desirable. It turns out that a search for such an answer, which is the main topic in this talk, will force us to look at the fundamental process of learning by statistical inference. This, incidentally, is quite different from the customary statistical thinking, in which one makes an arbitrary assumption about the data, namely, that they form a sample from some unknown distribution. This, then, will be estimated and the work is done. To quench any lingering doubts the estimation procedure may further be supported by an analysis of its optimality in the light of the assumed distribution. But because in current statistics there is no rational means to compare two distinct models, a critical step required for learning is lacking, and nothing beyond the initial guess is learned from the data.

The objective in any modeling is to learn the behavior of the machinery generating the observations. This machinery is physical rather than mathematical, and we perceive it only through the observations of its behavior. This means that our description of the machinery will have to be in the same terms, namely, some properties of the observed data. Modeling, then,
brings together two important ideas: coding and learning, coding being synonymous with des-
cription, and learning being virtually synonymous with extraction of properties from data. The
very word description presupposes the existence of a language, in which the properties can be
expressed. If by a language we mean a set of appropriately formed strings of recognizable
symbols, we may talk about the length of a description. The short descriptions, and, in partic-
ular, the very shortest ones, play a special role for reasons that are not entirely obvious; after
having set up a suitable formalism we discuss this important issue in Section 4. The often cited
Occam's razor: "entities should not be multiplied beyond necessity" while appealing to intui-
tion is too vague to be of much help in this regard. For a general discussion of the same issue
we refer to Kemeny (1965) and the fascinating paper, Chaitin (1979).

An important formalization of the just sketched process is the theory of algorithmic com-
plexity together with the subsequent developments to define the notion of a random string, the
construction of probabilities from the complexity, and to found inductive inference on it, Sol-
Martin-Lov (1974); see also the expository paper Li and Vitanyi (1988). In this theory, a de-
scription is a program in a universal computer such that the computer, when executing the
program, delivers a binary string representing the object. The shortest programs for an object,
then, embody the non-redundant descriptions in the agreed language of the universal com-
puter. Moreover, the common length of the shortest programs for long strings is relatively un-
affected by the choice of the universal computer itself in that the difference is just a constant,
namely, the length of the compilation program needed to simulate one universal computer in
another. Still, a complete independence of the complexity from the particular computer has not
been achieved, and, in fact, cannot be achieved.

However, for the purposes of learning properties of an object from its observed data, com-
puter programs are about the last thing to resort to. In fact, once we know the properties we
can write a program to describe the data; not the other way around. A much better way to
learn is to propose suitable models of behavior of the data, suggested by our preconceived ideas
about the object, and then form a judgement of the best performing model. Such models may
consist of a mixture of distributions, equations, and more general mathematical relations, and
they need by no means be computable; in fact, some of the most powerful models and theories
ever constructed deal with non-computable objects. That every estimate and evaluation we
carry out with such models must evidently be computable is of little consequence and should
not discourage us from using our imagination to create models of any kind, both computable
and non-computable alike. It seems therefore worthwhile to examine models and modeling
problems from a somewhat more general point of view, which we do in Section 2 in terms of
coding systems. Adapting the arguments in Solomonoff (1974) we construct a family of prob-