Part II

Experiments with MDL

Introduction to Part II

There are several theorems which say that, under reasonable conditions, using MDL leads to good or even optimal results [128, 14]. But such theorems are not sufficient to guarantee that it will work in the real world! First of all, while the mathematical conditions under which our optimality theorems hold may be reasonable, in practice they will never be *completely* met (is data ever generated by independent sampling from a fixed probability distribution?) Furthermore, when applying a theoretically optimal procedure in a practical setting, one always encounters additional problems. For example, parts of the data are missing; or data has to be discretized and it is not clear how to do this in an optimal manner; or the data turns out to have outliers of a completely unexpected kind; or the model class used is structured in a way that makes it computationally too hard to find the optimal model etc. In practice one always has to compromise, and it is not at all clear whether the theoretically optimal procedures will work well in practice.

Experiments with an Overly Simple Model

If one bases practical inductive inference on model classes that are *sufficiently simple*, then at least one of the problems mentioned above disappears: it becomes possible to efficiently compute, for arbitrary D, the single optimal model for D. For most sophisticated model classes this is not possible (or at least, no algorithm is known for doing this), and one has to search for a good model using heuristic search strategies that do not necessarily converge to the global optimum. Examples of such more sophisticated model classes are feed-forward neural networks with hidden layers, finite mixture distributions in statistics, hidden Markov models etc.

In the following two chapters, we base predictions on the so-called *Naive Bayes* model class which is simple enough so that we can efficiently find the optimal model. This class has been repeatedly shown to lead to results competitive with (sometimes even better than) those obtained by using much more sophisticated model classes [50, 90]. Yet it is evidently too simple ('naive') to provide realistic models for the domains of real-world data we will consider. We will use domains from areas as diverse as finance, forensic science, biology and medicine. We give a concrete example of the latter. The 'heart disease database' (see page 146) contains 270 observations. Each observation represents data concerning a single *case* (patient) and contains 14 *attribute entries*. Some of these attributes are age, sex, type of chest pain the patient suffers from (there are 4 possibilities), blood pressure, blood sugar level, 'maximum heart rate achieved' etc. The 14th attribute has a special status and is called the *class attribute*. It is either 1 (presence of heart disease) or 2 (absence of heart disease). All datasets we use have a similar decomposition in several attributes (either discrete or continuous-valued) one of which is a class attribute (always discrete-valued).

Throughout part II of this thesis, we use several methods to learn from a subset of the observations, the *training set* in order to *classify* new cases that are randomly picked from the *test set*. This is the subset of observations that are not contained in the training set. 'Classification' means that we are given all attribute values of a new case except for the class value. We then have to come up with a good prediction of the class value. In our example, we would be given age, sex etc. of a patient and we would be asked to predict whether or not that patient has heart disease.

When given enough training data from the heart disease database, the optimal Naive Bayes model class can predict whether or not a patient has a heart disease with about 84 % accuracy. (whereas, by simply predicting in accordance with the majority of cases would give us an accuracy of only 55 %). Similar accuracies are reached for the other data sets considered.

The key idea of the Naive Bayes model class is that all the attributes are considered *independent* of each other once the class value has been given. To give a concrete example, according to a Naive Bayes model, *given* the fact that a patient does not have a heart disease, the fact whether or not he has high blood pressure is independent of his age. According to a general practitioner we asked, this independence assumption is contrary to general medical wisdom. But note that we are only interested in predicting *whether or not a patient has heart disease*: since we are not asking about other attributes (like whether or not the patient has high blood pressure), we do not really care so much that the interrelations between other attributes are wrongly modeled.

The Power of Naive Bayes Of course, there may exist more sophisticated model classes, making less independence assumptions, that lead to better classifications of the heart disease data. But for these model classes, it is often not possible to efficiently compute the globally optimal model and/or prediction for the data at hand. We then have to settle for a suboptimal solution. It is perfectly possible that such a suboptimal solution is worse than the optimal Naive Bayes solution, which *can* be efficiently computed. The good overall behaviour of Naive Bayes suggests that this phenomenon frequently occurs in practice.

Comparing MDL to Bayesian Statistics and Maximum Likelihood

Specifically, we will use the Naive Bayes model class to compare the classification accuracies obtained by MDL to those obtained by several other methods of inductive inference. In Chapter 6, we compare MDL to some Bayesian methods and to the Maximum Likelihood method of classical statistics. This also involves some theoretical work: we have to instantiate the stochastic complexity to the Naive Bayes model class, which involves computing *Jeffrey's prior* for this model class. To make our results more widely applicable, we instantiate stochastic complexity to the class of Bayesian networks with arbitrary but fixed structure. This widely used model class [116, 137, 92] contains the Naive Bayes class as a special case.

Comparing MDL to MML

The Minimum *Message* Length Principle [161, 162] is a method for inductive inference that is very close in spirit, yet different from, the Minimum *Description* Length Principle. When investigating the relations between MDL and MML, we found a significant oversight in the derivation of MML estimators as given in [162]. We give a theoretical analysis of MML estimators and present a revised version of them. The original MML estimators differ from MDL estimators in an essential manner; but the revised MML estimators turn out to be closely related to Rissanen's 1996 refinement of the MDL Principle. Since all our theoretical results are asymptotic, it is not clear whether they lead to any significant differences in a practical setting. We therefore tested the original MML estimators, our revised MML estimators and an MDL-based prediction method on real world data, using once more the Naive Bayes model class. Our practical results turn out to be in agreement with our theoretical ones: MDL and our revised MML estimators perform slightly better than the original MML estimators.

Acknowledgements

The work reported in Part II of this thesis has been carried out jointly with the CoSCo group of the University of Helsinki, in particular P. Kontkanen, P. Myllymäki, T. Silander and H. Tirri. Two of the domains used in our experiments (the breast cancer and the lymphography domains; page 146) were obtained from the University Medical Centre, Institute of Oncology, Ljubljana, Slovenia. Thanks go to M. Zwitter and M. Soklič for providing the data.

Chapter 6

Predictive Distributions for Bayesian Networks

6.1 Introduction

This chapter is about discrete prediction problems where the task is to select one outcome from a finite set of possible alternatives. This is done by determining a *predictive distribution* over all the possible outcomes. A predictive distribution is simply a probability distribution over the sample space, conditioned on the *training data* D. The purpose of this chapter is to compare different alternatives for computing such a predictive distribution. We will consider two Bayesian and two MDL-based methods to arrive at a predictive distribution. All four methods satisfy some optimality criterion, and all four of them are, under reasonable conditions, guaranteed to work well for very large training samples. The question is then how well they work for small sample sizes – the theoretical results about the four distributions considered do not say too much about this situation.

More specifically, our predictive distributions will be based on a class of *candidate models* $\mathcal{M}$, which throughout this chapter we assume to be a finitely parameterizable class of probabilistic models over a finite sample space. Given some sample data (the *training data*), and an incomplete *query vector*, where the values of some of the problem domain variables are not given, the task is to compute the predictive distribution for the missing part of the query vector.

In the Bayesian *maximum a posteriori (MAP)* approach, the predictive distribution is determined by using the model in $\mathcal{M}$ with the highest posterior probability, given the training data and a prior distribution for the parameters. In the Bayesian *evidence* approach, the predictive distribution is obtained by averaging over all the distributions representable by the chosen model form. In the third and fourth approaches considered here, we define the predictive distribution by using Rissanen's recent [129] definition of stochastic complexity based on minimizing worst-case regret (Chapter 2, Definition 2.8). In the third approach, a form of the stochastic complexity is used that cannot be interpreted as a probabilistic model (Chapter 1, Definition 1.5). In the fourth approach, a probabilistic-model version of stochastic complexity is used. This form happens to coincide with the Bayesian evidence approach with the prior instantiated to Jeffreys' prior.

6.1.1 Overview of this chapter

In the theoretical part of this chapter (sections 6.2-6.3), we derive explicit formulas for the four distributions considered so as to make them applicable in a practical setting. In the practical part of the chapter (sections 6.5-6.6, we compare the performance of the distributions on several publicly available data sets coming from 'real-world' problem domains.

The details of the discrete prediction problem considered here, together with the MAP, the evidence and the stochastic complexity predictive distributions for solving this problem are described formally in Section 6.2. In Section 6.3, we introduce the *Bayesian network model class* [116, 92]. In Section 6.4, we instantiate the general results of Section 6.2 to this class. Specifically, we show how to define each of the above mentioned predictive distributions for a given Bayesian network structure and we show how to compute Jeffreys' prior for such structures.

We then (Section 6.5) use these formulas in experiments on real-world data sets. To avoid computational difficulties, in the experiments we restrict ourselves to simple *classification* problems, using the 'Naive Bayes' model class [42], which is a special case of a Bayesian network model class.

The experiments indicate that the evidence approaches produce the most accurate predictions in the log-score sense, outperforming both the MAP method and the version of stochastic complexity that cannot be seen as a probabilistic model. The evidence approaches are also quite robust in the sense that they predict surprisingly well even when only a small fraction of the full training set is used. We analyze the reasons for this behaviour in Section 6.5.3.

Our Contributions To enhance readability of this chapter, we have not kept a strict separation between results that were already known and our own, original contributions. We therefore indicate briefly what our contributions are, apart from the very idea of applying stochastic complexity to Bayesian networks: on the theoretical side, there are two contributions: first, we show how the stochastic complexity can be used to arrive at a predictive distribution in a direct manner (sections 6.2.3 and 6.4.2). Second, we show how to compute Jeffreys' prior for Bayesian networks (Section 6.4.4). All the empirical work reported in this chapter (sections 6.5 and 6.6) is completely ours.

6.2 Predictive Distributions for Discrete Domains

6.2.1 The Prediction and Classification Problems

Let $\mathbf{E} = \mathbf{E}_1 \times \ldots \times \mathbf{E}_m$ be a sample space that can be decomposed into m subspaces. Each $\mathbf{E}_i$ contains a finite number k_i of elements: $\mathbf{E}_i = \{e_{i1}, \ldots, e_{ik_i}\}$. To emphasize the distinction between vectors and their components we denote throughout this chapter an outcome of $\mathbf{E}$ by $\vec{x}$. An outcome of $\mathbf{E}_i$ is denoted by x_i. Hence $\vec{x} = (x_1, \ldots, x_m)^T$. Our training data will consist of a sample D of n outcomes: $D = (\vec{x}_1, \ldots, \vec{x}_n)$. Here $\vec{x}_i = (x_{i1}, \ldots, x_{im})$.

In the simplest setting, we want to use the training data D to arrive at a good prediction of a single new observation $\vec{x} \in \mathbf{E}$. We will call this observation the *test vector*. We do all our predictions on the basis of a *predictive distribution* $\mathcal{P}(\cdot|D)$ where $\mathcal{P}(\vec{X} = \vec{x}|D)$ is to be read as 'the probability that the test vector is $\vec{x}$ given that the training data was D'.

In most realistic settings some of the constituent variables of the test vector are given to us and are to be used for prediction of the remaining variables. We call the variables of the test vector that are given *clamped* and the variables that are to be predicted *free*. Specifically, let $\vec{x} = (x_1, \ldots, x_m)$ be the test vector. Without loss of generality we may assume that the vector $\vec{u}$ of clamped variables coincides with $(x_1, \ldots, x_u)$ for some $u < m$ while the vector $\vec{v}$ of free variables coincides with $(x_{u+1}, \ldots, x_m)$. The *classification problem* is a special case of our prediction problem. In the classification problem we are given a vector of 'feature variables' $(x_1, \ldots, x_{m'})$ which has to be classified into one of K categories or *classes*. We can represent this by setting $m = m' + 1$ and letting $\mathbf{E}_m = \{e_{m1}, \ldots, e_{mK}\}$ where each e_{mi} stands for a class. The 'clamped' variables are then $(x_1, \ldots, x_{m'})$ and the free variable, also called the *class variable*, is x_m.

We can now state our aim more precisely as follows: we wish to compute, for all possible $\vec{v}$, the probabilities

$$\mathcal{P}(\vec{v}|\vec{u}, D) = \frac{\mathcal{P}(\vec{v}, \vec{u}|D)}{\mathcal{P}(\vec{u}|D)} = \frac{\mathcal{P}((\vec{v}, \vec{u})|D)}{\sum_{\vec{v}} \mathcal{P}((\vec{v}, \vec{u})|D)}, \tag{6.1}$$

where the summation is over all $\vec{v} \in \mathbf{E}_{u+1} \times \ldots \times \mathbf{E}_m$.

Consequently, we see that the conditional distribution for the free variables can be computed using the complete data vector predictive probabilities $\mathcal{P}(\vec{x}|D)$ for each of the possible complete vectors $\vec{x} = (\vec{v}, \vec{u})$. It should be noted that the number of terms in the summation over possible $\vec{v}$ grows exponentially with the number of free variables. Therefore, predicting the values of many variables given the values of only a few may be difficult. However, in many cases of practical interest we only want to predict the values of a very small number (in classification problems just one) of the variables and then such difficulties do not arise.

We model our data using a model class $\mathcal{M}$ that is finitely parameterized by some $\Gamma \subset \mathbf{R}^k$ (Definition 2.7 of page 35). Hence $\mathcal{M}$ can be written as $\mathcal{M} = \{P(\cdot|\theta) \mid \theta \in \Gamma\}$. We further assume that all models in $\mathcal{M}$ render the data i.i.d. Hence for every $\theta \in \Gamma$, $P(\vec{x}^n|\theta) = \Pi_{i=1}^n P(\vec{x}_i|\theta)$.

In the next section we consider four different ways to compute predictive distributions $\mathcal{P}(\cdot|D)$ based on the model class $\mathcal{M}$.

6.2.2 The Bayesian Predictive Distributions $\mathcal{P}_{map}$ and $\mathcal{P}_{av}$

Given a prior density $P(\theta)$ defined for all $\theta \in \Gamma$, we can arrive at a posterior distribution $P(\theta|D)$ by using Bayes' rule:

$$P(\theta|D) \propto P(D|\theta)P(\theta). \tag{6.2}$$

In the *maximum a posteriori (MAP) probability* approach, the predictive distribution $\mathcal{P}$ is identified with the distribution corresponding to the single model $\breve{\theta}(D)$ maximizing the posterior distribution (6.2),

$$\breve{\theta}(D) = \arg\max_{\theta \in \Gamma} P(\theta|D).$$

The corresponding predictive distribution is

$$\mathcal{P}_{map}(\vec{x} \mid D) := P(\vec{x} \mid \breve{\theta}(D)). \tag{6.3}$$

If we assume the prior distribution $P(\theta)$ to be uniform, then (6.2) becomes $P(\theta|D) \propto P(D|\theta)$ and the MAP model becomes equal to the *Maximum Likelihood (ML) model*; see Chapter 1, page 18.

We can get more sophisticated predictions by averaging (integrating) over all the models in $\mathcal{M}$ instead of using a single model $\breve{\theta}$. The resulting distribution is called the *marginal* or *evidence* distribution in the Bayesian literature; we denote it by P_{av}. For $\vec{x}^n \in \mathbf{E}^n$, $P_{av}(\vec{x}^n)$ is given by:

$$P_{av}(\vec{x}^n) = \int P(\vec{x}^n|\theta)P(\theta)d\theta. \tag{6.4}$$

where the integration goes over all the models θ in $\mathcal{M}$. The resulting predictive distribution then coincides with the evidence distribution conditioned on training data D:

$$\mathcal{P}_{av}(\vec{x} \mid D) := P_{av}(\vec{x} \mid D) = \frac{P_{av}(\vec{x}, D)}{P_{av}(D)} \tag{6.5}$$

For more details about the Bayesian MAP and evidence approaches we refer to Chapter 2, Section 2.8.

6.2.3 The 'direct' Stochastic Complexity Predictive Distribution $\mathcal{P}_{sc}$

In Chapter 2 we introduced the stochastic complexity distribution $P_{sc}(\cdot) = P_{sc}(\cdot|\mathcal{M})$ as the distribution that minimizes the worst case excess code length, or equivalently, the *regret* (see equations 2.5-2.7 of Section 2.3). In Section 2.7 we further showed that sequential proportional betting on the basis of P_{sc} minimizes the worst-case excess loss in capital. Here the 'excess' is over the loss we would have made if we had bet on the basis of the model in the class that, with hindsight, would have given us the least loss of capital; see page 43. This property of P_{sc} provides a strong reason for using it in actual prediction tasks.

For the model classes $\mathcal{M}$ we will consider in our experiments, a unique ML estimator $\hat{\theta}(D)$ exists for all $D \in \mathbf{E}^*$. In addition $\mathbf{E}$ is finite, and this means (Chapter 2, Equation (2.10), page 37) that the stochastic complexity distribution can be explicitly written as follows:

$$P_{sc}(D) = \frac{P(D|\hat{\theta}(D))}{\sum_{D \in \mathbf{E}^n} P(D|\hat{\theta}(D))} = (F(n)^{-1})P(D|\hat{\theta}(D)) \tag{6.6}$$

Here $F(n)$ is defined as $F(n) = \sum_{D \in \mathbf{E}^n} P(D|\hat{\theta}(D))$. At first sight, using (6.6) as the predictive distribution may seem infeasible in practice since computing the normalizing sum $F(n)$ involves summing over a number of terms exponential in n. The problem disappears if one computes a predictive distribution for a data set D of length n in the following straightforward manner:

$$\begin{aligned}
P_{sc}(\vec{x} \mid D) &= \frac{P_{sc}(\vec{x}, D)}{\sum_{\vec{x}'} P_{sc}(\vec{x}', D)} \\
&= \frac{P(\vec{x}, D \mid \hat{\theta}(\vec{x}, D)) \cdot (F(n+1))^{-1}}{\sum_{\vec{x}'} P(\vec{x}', D \mid \hat{\theta}(\vec{x}', D)) \cdot (F(n+1))^{-1}} \\
&= \frac{P(\vec{x}, D \mid \hat{\theta}(\vec{x}, D))}{\sum_{\vec{x}'} P(\vec{x}', D \mid \hat{\theta}(\vec{x}', D))} \\
&\overset{\text{i.i.d.}}{=} \frac{P(\vec{x} \mid \hat{\theta}(\vec{x}, D))P(D \mid \hat{\theta}(\vec{x}, D))}{\sum_{\vec{x}'} P(\vec{x}' \mid \hat{\theta}(\vec{x}', D))P(D \mid \hat{\theta}(\vec{x}', D))}.
\end{aligned} \tag{6.7}$$

The *stochastic complexity predictive distribution* $\mathcal{P}_{sc}$ can now be identified with the conditional stochastic complexity distribution $P_{sc}(\vec{x} \mid D)$ as given in Equation (6.7):

$$\mathcal{P}_{sc}(\vec{x}|D) := P_{sc}(\vec{x}|D). \tag{6.8}$$

6.2.4 Connecting P_{av} and P_{sc}: the $\mathcal{P}_{jef}$ Predictive Distribution

P_{sc} is not a probabilistic model

Though $P_{sc}(\cdot)$ is defined with respect to a class of probabilistic models $\mathcal{M}$ (Chapter 1, Definition 1.5), it is not *itself* a probabilistic model: in general,

$$\sum_{z \in \mathbf{E}} P_{sc}(D, z) \neq P_{sc}(D).$$

Let us contrast this with a P that *does* conform to the definition of probabilistic model: let P be a probabilistic model and let $n_1 < n_2$. Let P_{n_i} be the restriction of P to sequences of length n_i. Both P_{n_1} and P_{n_2} are probability distributions (over $\mathbf{E}^{n_1}$ and $\mathbf{E}^{n_2}$ respectively). The fundamental property of a probabilistic model is that, for all $D \in \mathbf{E}^{n_1}$, $P_{n_1}(D) = \sum_{D' \in \mathbf{E}^{n_2 - n_1}} P_{n_2}(D, D') = P_{n_2}(D)$: P_{n_2} can be seen as an extension of P_{n_1}.

SC has been defined so as to minimize the worst-case regret. The distribution which achieves this for us for sample size n_2 can*not* be interpreted as an extension of the distribution which achieves it for sample size n_1. Therefore, the stochastic complexity distributions defined for data sequences of length 1, of length 2 etc. are really all different: perhaps a better notation would be to write P^n_{sc} for the stochastic complexity distribution over sequences of length n.

Let us now once more consider the game we introduced in Chapter 2, Section 2.7, page 43: we are given a model class $\mathcal{M}$ and we sequentially predict x_{i+1} on the basis of x^i. We saw on page 43 that sequentially predicting on the basis of P_{sc} minimizes the worst-case regret. However, the P_{sc} we used there was really the stochastic complexity distribution as defined *for data of length n*. So we really predicted x_1 on the basis of $P^n_{sc}(x_1)$, x_2 on the basis of $P^n_{sc}(x_2|x_1)$ etc. In practical settings, the total number n of predictions we want to make is often unknown. In that case, we do not know whether to use P^1_{sc}, P^2_{sc} or any other P^i_{sc} for prediction. Using a different n for different predictions will certainly not make our predictions optimal! But this is what will happen if we use $\mathcal{P}_{sc}$ as defined in Equation 6.8 for prediction. To see this, note that in Section 6.2.3 we implicitly used P^{n+1}_{sc} to define the predictive distribution (6.8): $\mathcal{P}_{sc}(\vec{x}|D)$ was defined in terms of $P_{sc}(D, \vec{x})$ and the sample $(D, \vec{x})$ is of length $n+1$. So if we would sequentially predict on the basis of (6.8), this would amount to predicting x_1 on the basis of P^1_{sc}, x_2 on the basis of $P^2_{sc}(\cdot|x_1)$, x_3 on the basis of $P^3_{sc}(\cdot|x_1, x_2)$ etc., which is not optimal in general.

Things are not necessarily as bad as they seem, however. Rissanen [129] proved that under suitable regularity conditions on the class of models $\mathcal{M}$, as n grows larger, P^n_{sc} starts to behave more and more like a probabilistic model - the restriction of distribution P^{n+1}_{sc} to outcomes of length n will be almost identical to the distribution P^n_{sc}. At least for the class of Bernoulli models, this 'convergence' to a probabilistic model seems to take place very fast [129]. For this reason we decided to use the predictive distribution P_{sc} in our experiments after all.

Nevertheless, we could ask ourselves whether there exists a probabilistic model that achieves a worst-case regret that is almost as small as that of the stochastic complexity. We present such a model below.

A Probabilistic-Model Version of P_{sc}

The evidence distribution P_{av} can be seen as a probabilistic model irrespective of the prior w used, since for all $D \in \mathbf{E}^*$ we have

$$\sum_{z \in \mathbf{E}} \int P(D, z|\theta) w(\theta) d\theta = \int P(D|\theta) w(\theta) d\theta.$$

Under certain regularity conditions on the class of models $\mathcal{M}$, one can prove that the prior over the models in $\mathcal{M}$ which yields the lowest regret $\mathcal{R}_{L_{av,w}}(D) = -\log P_{av}(D) - [-\log P(D|\hat{\theta}(D))]$ is the so-called *Jeffreys' prior* $\pi(\theta)$ [129] which we define below. We will denote the marginal distribution P_{av} with the prior instantiated to Jeffreys' prior by $P_{jef} = P_{jef}(\cdot|\mathcal{M})$.

The specific form of Jeffreys' prior depends on the model class $\mathcal{M}$ that is used. For the model classes we will consider in our experiments, Jeffreys' prior is proper[1]. In that case it is given by (see for example [129] or [17]):

$$\pi(\theta) = \frac{|I(\theta)|^{1/2}}{\int |I(\eta)|^{1/2} d\eta}. \tag{6.9}$$

Here $|I(\theta)|$ is the determinant of the *Fisher information matrix* $I(\theta)$. Denoting $\theta = (\theta_1, \ldots, \theta_k)$, entry (i, j) of matrix $I(\theta)$ is defined as

$$[I(\theta)]_{i,j} = -E_\theta \left[\frac{\partial^2 \log P(\vec{X}|\theta)}{\partial\theta_i \partial\theta_j} \right].$$

Originally, Jeffreys' prior was derived by invariance arguments [78, 17]: the value of $\pi(\theta)$ is invariant under 1-1 transformations of the parameter space. We will show below that if the model class $\mathcal{M}$ is an *exponential family* (Chapter 3, Section 3.4), then P_{jef} essentially minimizes the worst-case regret. In the next section we introduce the model class of Bayesian networks which we will use in our experiments. We show there that this class is an exponential family. This means that we can use the predictive distribution based on P_{jef} as an alternative version of the stochastic complexity predictive distribution P_{sc}. In Section 6.4, we derive an analytic expression for Jeffreys' prior $\pi(\theta)$ for the case where θ indexes a Bayesian network, and show how to calculate P_{jef} for the class of Bayesian networks with a fixed but arbitrary structure.

To show that P_{jef} minimizes regret we need the following fundamental result (a special case of a theorem to be found in [26]).

Theorem 6.1 (Clarke & Barron) *Let $\mathcal{M}$ be an exponential family of irreducible dimension k (as defined in Chapter 3, Section 3.4) over sample space $\mathbf{E}$. Let $\mathcal{M}$ be parameterized by some $\Gamma \subset \mathbf{R}^l$ such that (1) there exists a bijection $g : \Gamma \to \mathcal{M}$ and (2) $\ln P(x|\theta)$ as a function of θ is differentiable infinitely often for all $x \in \mathbf{E}$ at all $\theta \in \Gamma$. Let $w(\theta)$ be a proper prior density over Γ with $w(\theta) > 0$ for all $\theta \in \Gamma$. Let $x^n \in \mathbf{E}^n$ be generated by repeated sampling from $P(\cdot|\theta^*)$ where θ^* is an arbitrary member of Γ. Then, with θ^*- probability 1,*

$$-\ln P_{av}(x^n) = -\ln P(x^n|\hat{\theta}(x^n)) + \frac{k}{2} \ln \frac{n}{2\pi} + \ln \frac{\sqrt{|I(\theta^*)|}}{w(\theta^*)} + o(1). \tag{6.10}$$

where P_{av} is the evidence distribution (6.4) *with prior w.*

If we substitute Jeffreys' prior $\pi(\theta)$ (Equation 6.9) for the prior $w(\theta)$, then the term $\ln(\sqrt{|I(\theta^*)|}/w(\theta^*))$ in Equation 6.10 becomes independent of the data generating model θ^*. This means that whatever this 'true' θ^* is, we obtain, with probability one, the same asymptotic expansion

$$-\ln P_{jef}(x^n) = -\ln P(x^n|\hat{\theta}(x^n)) + \frac{k}{2} \ln \frac{n}{2\pi} + \ln \int \sqrt{|I(\theta)|} d\theta + o(1). \tag{6.11}$$

[1] A prior w is *proper* if $\int w(\theta) d\theta = 1$.

Let w be a proper prior such that within the parameter space there exists a region R of non-zero volume with $\int_R w(\theta)d\theta \neq \int_R \pi(\theta)d\theta$. Let P_{av} be the marginal distribution with such a prior w. Since both w and π are proper the parameter space must also contain a region R' of non-zero volume such that $w(\theta) < \pi(\theta)$ for all $\theta \in R'$. By Theorem 6.1 above it follows that for every $\theta^* \in \mathrm{int}(R')$, it holds with θ^*-probability 1 that $-\ln P_{av}(x^n) = -\ln P_{jef}(x^n) + K + o(1)$ for some $K > 0$. Hence for all P_{av} based on proper priors w as defined above, there exists θ^* for which they perform worse than π, while π itself reaches (with probability 1) the same asymptotic expansion (6.11) for all $\theta^* \in \Gamma$. It follows that, among all proper priors, Jeffreys' prior minimizes the worst-case regret that is reached with probability 1. Here 'worst-case' is with respect to generating distributions instead of data sequences.

This provides a justification for basing predictions on P_{jef} instead of the stochastic complexity whenever the model class is an exponential family: First, P_{jef} is a probabilistic model (so we can use it without knowing the number of predictions to be made). Second, within the class of models that can be written as a Bayesian marginal distribution $P_{av}(\cdot|\mathcal{M})$, it is the one that minimizes worst-case regret (where worst-case regret is defined with respect to a worst-case generating distribution instead of a worst-case data sequence). Third, all such marginal distributions with priors $w(\theta) > 0$ for all $\theta \in \Gamma$ are asymptotically good approximations of the stochastic complexity; see [128].

Formally, we define the *predictive distribution based on P_{jef}* by equating it to P_{jef} conditioned on D:

$$\mathcal{P}_{jef}(\vec{x}|D) := P_{jef}(\vec{x}|D) \tag{6.12}$$

Discussion The line of reasoning just presented suffers from two weaknesses: first, Equation 6.10 has only been proven to hold (with probability 1) under the assumption that there is some true model in $\mathcal{M}$ generating the data. Second, it is not yet clear how close $-\ln P_{jef}(x^n)$ really is to $-\ln P_{sc}(x^n)$, or whether there does not exist a different probabilistic model that can*not* be written as a Bayesian marginal distribution, but that nevertheless achieves a smaller regret.

Both weaknesses can most likely be removed: *very* recently (a week before this thesis had to go to the printer's), Takeuchi and Barron [149] announced that they have proven theorems stating that for all exponential families the following holds: First, (6.11) is the case not just with probability 1 but, independent of any underlying distribution, *uniformly* for all $x^n \in \mathbf{E}^n$ which are such that $\hat{\theta}(x^n)$ lies in the interior of the parameterspace. Second, $-\ln P_{sc}(x^n) = -\ln P_{jef}(x^n) + o(1)$ also uniformly for all $x^n \in \mathbf{E}^n$ which are such that $\hat{\theta}(x^n)$ lies in the interior of the parameter space. Therefore, using P_{jef} should lead to approximately the same (and hence by definition minimal in the worst-case) regret as P_{sc}. This would resolve both weaknesses. No proofs of these theorems have yet been published however.

6.3 Bayesian Networks

A Bayesian (belief) network [116, 79, 92] is a graphical high-level representation of a probability distribution over a finite set of discrete random variables $X_1, \ldots, X_m$. Bayesian networks have their roots in the notion of 'conditional independence' in statistics [35] and the method of 'path analysis' in the social sciences [165]. Nowadays, they are used extensively in the field of 'reasoning under uncertainty' [116, 114]. Before defining Bayesian networks themselves we need to say something about the sample space over which they are defined.

The Sample Space The variables $X_1, \ldots, X_m$ take on values in $\mathbf{E}_1, \ldots, \mathbf{E}_m$ respectively. In our setting, we assume the $\mathbf{E}_i$ to be finite. Previously we denoted elements in $\mathbf{E}_i$ as follows: $\mathbf{E}_i = \{e_{i1}, \ldots, e_{ik_i}\}$. Whenever in the sequel we refer to an outcome x of one of the random variables X_1, X_2, ... or X_m, it will be clear from the context to which of these random variables x belongs. We can therefore denote outcome e_{ij} by the number j without introducing any ambiguity. When defining Bayesian networks this will allow for a considerable simplification of notation. Hence, from now on we write $\mathbf{E}_i = \{1, \ldots, k_i\}$.

6.3.1 Definition of Bayesian Networks

A Bayesian network consists of a *structure* $\mathcal{G}$ and a *parameter set* θ. We discuss both in turn.

The Bayesian Network Structure The Bayesian network structure is a directed acyclic graph (DAG), where the nodes correspond to the domain variables $X_1, \ldots, X_m$. An example Bayesian network structure is shown in Figure 6.1. Below we first introduce the concepts involved an a Bayesian network structure; we then give an example based on Figure 6.1.

The graph $\mathcal{G}$ is represented by a set of $m-1$ *Parent variable sets* $Pa_i \subseteq \{X_{i+1}, \ldots, X_m\}$ where $1 \leq i < m$. For each node (corresponding to) X_i, the set Pa_i represents the nodes in the graph that are the parents of X_i. We define $\mathbf{E}_{\mathrm{pa}_i}$ to be the set of all possible configurations of the values of the parents Pa_i of variable X_i. Each $q_i \in Pa_i$ stands for a vector of values of random variables. We write $P(Pa_i = q_i)$ to denote the probability that the parent variables of X_i take on the values corresponding to configuration q_i. Hence $P(Pa_i = q_i)$ is an abbreviation of $P(X_{j_1} = x_{j_1}, \ldots, X_{j_{m'}} = x_{j_{m'}})$ where $X_{j_1}, \ldots, X_{j_{m'}}$ are the elements of Pa_i and $q_i = (x_{j_1}, \ldots, x_{j_{m'}})$.

From the definition of conditional probability one sees that each probability distri-

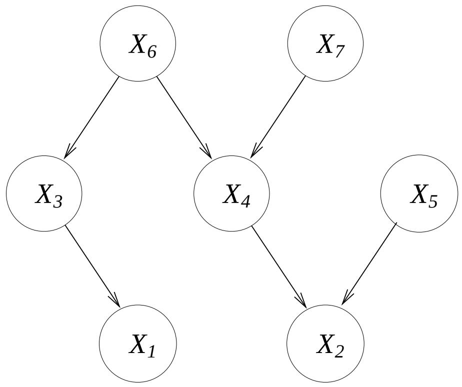


Figure 6.1: An example Bayesian network.

bution over $\mathbf{E} = \mathbf{E}_1 \times \ldots \times \mathbf{E}_m$ can be written such that, for all $\vec{x} \in \mathbf{E}$,

$$\begin{aligned}
&P(\vec{x}) = P(X_1 = x_1, \ldots, X_m = x_m) = \\
&P(X_1 = x_1 \mid X_2 = x_2, \ldots, X_m = x_m) \times P(X_2 = x_2 \mid X_3 = x_3, \ldots, X_m = x_m) \times \\
&\qquad \cdots \times P(X_m = x_m) = \\
&\qquad \prod_{i=1}^{m} P(X_i = x_i | X_{i+1} = x_{i+1}, \ldots, X_m = x_m) \qquad (6.13)
\end{aligned}$$

The key idea behind Bayesian Networks is to use a given network structure $\mathcal{G}$ to define a set of *independence assumptions* such that, in the decomposition (6.13), the distribution of each X_i depends not on *all* of the variables $X_{i+1}, \ldots, X_m$, but only on the subset of variables given by Pa_i. Hence a structure $\mathcal{G}$ *restricts* the set of possible distributions over $\mathbf{E}$ to the set of distributions that can be written as follows:

$$\begin{aligned}
&P(\vec{x}) = P(X_1 = x_1, \ldots, X_m = x_m) = \\
&\qquad P(X_1 = x_1 \mid Pa_1 = q_1) \times P(X_2 = x_2 \mid Pa_2 = q_2) \times \\
&\qquad \cdots \times P(X_m = x_m) = \\
&\qquad \prod_{i=1}^{m} P(X_i = x_i | Pa_i = q_i), \qquad (6.14)
\end{aligned}$$

where for notational convenience we write $P(X_m = x_m | Pa_m = q_m)$ instead of $P(X_m = x_m)$ (q_m can be thought of as always taking on a single value, independent of the outcome of any of the X_i).

Example 6.2 Consider Figure 6.1 again. Suppose that in this figure, $\mathbf{E}_4 = \{1, 2\}$ and $\mathbf{E}_5 = \{1, 2, 3\}$. Then $Pa_2 = \{X_4, X_5\}, \mathbf{E}_{\mathrm{pa}_2} = \{(1,1), (1,2), (1,3), (2,1), (2,2), (2,3)\}$. Let now $q_2 \in \mathbf{E}_{\mathrm{pa}_2}$; as an example, take $q_2 = (1, 2)$. Then $P(Pa_2 = q_2) = P(X_4 = 1, X_5 = 2)$.

Intuitively, the independence assumption encoded in a Bayesian network structure $\mathcal{G}$ is the following: given the values of its parents, a variable X_i becomes independent of all variables in the graph except its descendants. To get some (quite rough) idea of what this means, consider variables X_3 and X_4 in the figure. Let $j \in \{2,4,5,7\}$. The fact that the probability distribution over $X_1, \ldots, X_7$ can be written in the form (6.14) implies that , for all $x_3 \in \mathbf{E}_3, x_6 \in \mathbf{E}_6$ and all $x_j \in \mathbf{E}_j$, we have $P(X_3 = x_3 | X_6 = x_6, X_j = x_j) = P(X_3 = x_3 | X_6 = x_6)$: once the value of X_6 is known (given), the distribution of X_3 becomes independent of any of the other outcomes except its descendants (which, in this case, consist only of X_1). However, if the value of X_6 is not known, then the distribution of X_3 may very well depend on other outcomes. For example, it is easy to come up with a distribution of the form (6.14) for which $P(X_3 = x_3 | X_4 = 1) \neq P(X_3 = x_3 | X_4 = 2)$.

The Bayesian Network Parameter Set A Bayesian network structure $\mathcal{G} = \{Pa_1, \ldots, Pa_{m-1}\}$ represents the class of all probability distributions P on variables $X_1, \ldots, X_m$ such that $P(\vec{X} = \vec{x})$ can, for all $\vec{x}$, be written as in (6.14). Consequently, in the Bayesian network model class induced by a graph $\mathcal{G}$, a single distribution P is uniquely determined by fixing the values of the probabilities (6.14). We will therefore parameterize such a class by using one parameter for each of these probabilities. The parameter corresponding to probability $P(X_i = x_i | Pa_i = q_i)$ will be indicated by $\theta^i_{x_i|q_i}$. Hence for all $1 \leq i \leq m$, all $q_i \in \mathbf{E}_{\mathrm{pa}_i}$ and all $x_i \in \mathbf{E}_i$:

$$P(X_i = x_i | Pa_i = q_i) = \theta^i_{x_i|q_i} \tag{6.15}$$

The parameter vector consisting of the values $\theta^i_{x_i|q_i}$ for all $x_i \in \mathbf{E}_i$ and $q_i \in Pa_i$ will be denoted by θ^i (we assume the parameters $\theta^i_{x_i|q_i}$ to be ordered in some fixed manner). Summarizing:

Definition 6.3 (Bayesian Network) *A Bayesian network $(\theta, \mathcal{G})$ consists of a structure $\mathcal{G} = \{Pa_1, \ldots, Pa_{m-1}\}$ and a parameter vector $\theta = (\theta^1, \ldots, \theta^m)$. Here the θ^i are themselves vectors with components $\theta^i_{x_i|q_i}$ for all $x_i \in \mathbf{E}_i$ and $q_i \in Pa_i$. The Bayesian network probability distribution P_θ corresponding to Bayesian network $(\theta, \mathcal{G})$ is defined such that for all $\vec{x} \in \mathbf{E}$, all $1 \leq i \leq m$, all $q_i \in \mathbf{E}_{pa_i}$ and all $x_i \in \mathbf{E}_i$, Equations 6.14 and 6.15 hold.*

Definition 6.4 (Bayesian Network Model Class) *The class $\Gamma_\mathcal{G}$ of parameter values for Bayesian networks with structure $\mathcal{G}$ contains exactly those θ with, for all $1 \leq i \leq m$ and all $q_i \in \mathbf{E}_{pa_i}$:*

1. *for all $x_i \in \mathbf{E}_i$: $\theta^i_{x_i|q_i} > 0$*

2. $\sum_{x_i \in \mathbf{E}_i} \theta^i_{x_i|q_i} = 1$.

The class of Bayesian networks with structure $\mathcal{G}$, denoted by $\mathcal{M}_\mathcal{G}$, is given by $\mathcal{M}_\mathcal{G} = \{P_\theta | \theta \in \Gamma_\mathcal{G}\}$.

In the following we assume an arbitrary but fixed structure $\mathcal{G}$ and we consider the class of probability distributions $\mathcal{M}_{\mathcal{G}}$. Note that our definition of $\mathcal{M}_{\mathcal{G}}$ excludes points at the boundaries of the parameter space. Note further that, by (6.14) conditional distributions of variables given values for their parent values are multinomial: $X_i|q_i \sim \text{Multi}(1; \theta^i_{1|q_i}, \ldots, \theta^i_{k_i|q_i})$.

In the sequel, we will sometimes use the notation $P(\cdot|\theta)$ to denote $P_\theta(\cdot)$.

6.3.2 Parameter Priors for Bayesian Networks

The family of Dirichlet distributions is *conjugate* (see e.g. [39] or [17]) to the family of multinomials: when a Dirichlet density is used as a prior for the family of multinomials, then the functional form of the parameter distribution remains invariant in the prior-to-posterior transformation. It is therefore convenient to use prior distributions from this family. Moreover, as we will see, all theoretically motivated prior distributions for Bayesian network model classes that we consider in our experiments in this and the next chapter turn out to be Dirichlet. We now give the definition of the Dirichlet distributions in terms of their densities (adapted from [17]):

Definition 6.5 *Let $\theta_1, \ldots, \theta_k$ be such that $\sum_{i=1}^k \theta_i = 1$ and $0 \leq \theta_i \leq 1$ for all i. The* Dirichlet density *corresponding to* hyperparameters $(\mu_1, \ldots, \mu_k)$ *is given by*

$$P(\theta_1, \ldots, \theta_k|\mu_1, \ldots, \mu_k) = c \cdot \prod_{i=1}^{k} (\theta_i)^{(\mu_i - 1)} \tag{6.16}$$

where c is a constant needed for normalization. If $\theta_1, \ldots, \theta_k$ are distributed according to (6.16), we write

$$(\theta_1, \ldots, \theta_k) \sim \text{Di}(\mu_1, \ldots, \mu_k)$$

For our Bayesian network distributions, we will assume that for each $1 \leq i \leq m$, for each $q_i \in \mathbf{E}_{\text{pa}_i}$,

$$(\theta^i_{1|q_i}, \ldots, \theta^i_{k_i|q_i}) \sim \text{Di}(\mu^i_{1|q_i}, \ldots, \mu^i_{k_i|q_i})$$

for some $(\mu^i_{1|q_i}, \ldots, \mu^i_{k_i|q_i})$. We further assume that the parameter vectors $(\theta^i_{1|q_i}, \ldots, \theta^i_{k_i|q_i})$ are independent, so that the joint prior distribution of all the parameters θ becomes

$$P(\theta) \propto \prod_{i=1}^{m} \prod_{q_i \in \mathbf{E}_{\text{pa}_i}} \prod_{x_i \in \mathbf{E}_i} (\theta^i_{x_i|q_i})^{(\mu^i_{x_i|q_i} - 1)}, \tag{6.17}$$

which is itself a Dirichlet distribution.

Observe that the *uniform prior* is a Dirichlet density. It is obtained when the values of all hyperparameters $\mu^i_{x_i|q_i}$ are set to 1, since in that case all the exponents in 6.17 become 0.

6.3.3 Bayesian network families $\mathcal{M}_{\mathcal{G}}$ are exponential families

As explained in Section 6.2.4, for model classes that satisfy the regularity conditions of Theorem 6.1, the Bayesian evidence (6.4) with the prior instantiated to Jeffreys' prior can serve as an approximation of stochastic complexity. The following theorem demonstrates that Bayesian network model classes $\mathcal{M}_{\mathcal{G}}$ satisfy these conditions:

Proposition 6.6 *Let $\mathcal{M}_{\mathcal{G}}$ be the class of probability distributions corresponding to an arbitrary Bayesian network structure $\mathcal{G}$ and let $\Gamma_{\mathcal{G}}$ be the class of associated parameters. Then (1) the class $\mathcal{M}_{\mathcal{G}}$ is an exponential family of irreducible dimension k for some k, and (2) there exists a bijection $g : \Gamma_{\mathcal{G}} \to \mathcal{M}_{\mathcal{G}}$ such that $P(\vec{x}|\theta) = g(\theta)$ as a function of θ is differentiable infinitely often for all $\vec{x} \in \mathbf{E}$ at all $\theta \in \Gamma_{\mathcal{G}}$.*

Proof: To see that $\mathcal{M}_{\mathcal{G}}$ is an exponential family let $a = |\mathbf{E}|$, order the elements of $\mathbf{E}$ from 1 to a and, for $1 \leq i \leq a$, let $\vec{x}_i$ be the element in $\mathbf{E}$ corresponding to index i in this ordering. Now write:

$$P(\vec{X} = \vec{x}|\beta) = \frac{1}{Z(\beta)} \exp(-\sum_{i=1}^{a} \beta_i \mathcal{I}(\vec{x} = \vec{x}_i)) \tag{6.18}$$

where $\mathcal{I}$ is the indicator function (Chapter 3, page 51). Clearly, this distribution is of the form (3.11) required for the exponential family (page 56). One easily checks that each model $P \in \mathcal{M}_{\mathcal{G}}$ can be written in the form (6.18). One can show [82] that for each exponential family there exists some minimal k such that it can be written in the form (3.11) with $\beta = (\beta_1, \ldots, \beta_k)$. Hence $\mathcal{M}_{\mathcal{G}}$ must be irreducible for some k, which proves (1).

We now prove (2). The only part of (2) that is not immediate is the condition that g must map each parameter vector to a different model. We prove this by showing that for all parameter values $\theta, \theta' \in \Gamma$ with $\theta \neq \theta'$ we have $P(\cdot|\theta) \neq P(\cdot|\theta')$. By renaming random variable X_i to X_{m-i+1} and renaming all the elements in Pa_i correspondingly, we see that the following definition of a Bayesian network structure is equivalent to the one we gave in Section 6.3.1: a Bayesian network structure over a set of m discrete random variables $X_1, \ldots X_m$ is defined by a set of parent variable sets $\{Pa_1, \ldots, Pa_m\}$ where $Pa_i \subseteq \{X_1, \ldots, X_{i-1}\}$. This alternative way of ordering variables will be adopted in the proof.

Clearly, for $1 \leq l \leq m$, the set $\mathcal{G}_l = \{Pa_1, \ldots, Pa_l\}$ forms a Bayesian network structure for the set of variables $X_1, \ldots, X_l$. For any set of parameter values $\theta = (\theta^1, \ldots, \theta^m) \in \Gamma$ we let $\theta\rfloor_l$ stand for the restriction of θ to $(\theta^1, \ldots, \theta^l)$

Let Γ_l be the set of parameters corresponding to $\mathcal{M}_{\mathcal{G}_l}$. We now prove by induction on l that for $1 \leq l \leq m$, for all $\theta_1, \theta_2 \in \Gamma_l$, $P(\cdot|\theta_1) \neq P(\cdot|\theta_2)$, from which our desired result follows as a special case. The base case, $l = 1$, is immediate: if $\theta\rfloor_1 \neq \theta'\rfloor_1$, then for some $x_1 \in \mathbf{E}_1$, $P(x_1|\theta\rfloor_1) \neq P(x_1|\theta'\rfloor_1)$.

Now for the induction step. Suppose for every $\theta\rfloor_l$ and $\theta'\rfloor_l$ such that $\theta\rfloor_l \neq \theta'\rfloor_l$ we have $P(x_1, \ldots, x_l|\theta\rfloor_l) \neq P(x_1, \ldots, x_l|\theta'\rfloor_l)$ for at least one $(x_1, \ldots, x_l) \in \mathbf{E}_1 \times \ldots \times \mathbf{E}_l$. Now consider any θ and θ' such that $\theta\rfloor_{l+1} \neq \theta'\rfloor_{l+1}$. For these θ and θ' we either (first case) have $\theta\rfloor_l \neq \theta'\rfloor_l$ or (second case) $\theta\rfloor_l = \theta'\rfloor_l$ and $\theta^{l+1} \neq (\theta')^{l+1}$. In the first case, suppose by means of contradiction that

$$P(x_1, \ldots, x_{l+1}|\theta\rfloor_{l+1}) = P(x_1, \ldots, x_{l+1}|\theta'\rfloor_{l+1}) \text{ for all } x_1, \ldots, x_{l+1} \tag{6.19}$$

It follows that for all $x_1, \ldots, x_l$

$$P(x_1, \ldots, x_l|\theta\rfloor_l) = \sum_{x_{l+1}} P(x_1, \ldots, x_{l+1}|\theta\rfloor_{l+1}) = P(x_1, \ldots, x_l|\theta'\rfloor_l)$$

where the second equality follows from Equation (6.19). But this contradicts our induction hypothesis.

In the second case, there must be a q_{l+1} and an x_{l+1} such that $\theta^{l+1}_{x_{l+1}|q_{l+1}} \neq (\theta')^{l+1}_{x_{l+1}|q_{l+1}}$. There must also be an assignment $X_1 = x_1, \ldots, X_l = x_l$ such that $Pa_{l+1} = q_{l+1}$. Consider the data vector $\vec{x} = (x_1, \ldots, x_l, x_{l+1})$ for the $x_1, \ldots, x_l$ and x_{l+1} just defined. It is easy to see that $P(\vec{x}|\theta\rfloor_{l+1}) = c \cdot \theta^{l+1}_{x_{l+1}|q_{l+1}}$ while $P(\vec{x}|\theta'\rfloor_{l+1}) = c \cdot (\theta')^{l+1}_{x_{l+1}|q_{l+1}}$ for the same constant c, so $P(\vec{x}|\theta\rfloor_{l+1}) \neq P(\vec{x}|\theta'\rfloor_{l+1})$ which is what we had to prove. $\Box$

6.4 Predictive Distributions for Bayesian Networks

In the previous section we defined the model class of Bayesian networks together with the functional form of the prior distributions we will use for this model class. This allows us to write the predictive distributions $\mathcal{P}_{map}$ (6.3) and $\mathcal{P}_{av}$ (6.5) more explicitly, as will be shown in the next two sections. The direct stochastic complexity predictive distribution $\mathcal{P}_{sc}$ (6.8) is instantiated for the Bayesian network case in Section 6.4.3. In order to determine the $\mathcal{P}_{jef}$ predictive distribution (6.12), we show in Section 6.4.4 how to compute Jeffreys' prior for Bayesian network model classes.

6.4.1 The $\mathcal{P}_{map}$ Predictive Distribution for Bayesian Networks

Let $\mathcal{M}_\mathcal{G}$ be a Bayesian network family for arbitrary network structure $\mathcal{G}$. We now interpret the elements in $\mathcal{M}_\mathcal{G}$ as i.i.d. probabilistic models: $P(\cdot|\theta) \in \mathcal{M}_\mathcal{G}$ stands for the hypothesis that the data are i.i.d., each single observation being distributed according to $P(\cdot|\theta)$. We denote by $X_1, \ldots, X_m$ the random variables in $\mathcal{G}$, such that $(X_1, \ldots, X_m)$ takes on values in $\mathbf{E} = \mathbf{E}_1 \times \ldots \times \mathbf{E}_m$. Let $D = (\vec{x}_1, \ldots, \vec{x}_n) \in \mathbf{E}^n$. We denote by X_{ji} the instance of random variable X_i for the j-th outcome. We denote by Pa_{ji} the instance of random variable Pa_i for the j-th outcome. Let $x_i \in \mathbf{E}_i$ and let $q_i \in \mathbf{E}_{\mathrm{pa}_i}$. We will use indicator functions (see Chapter 3, page 51) $\mathcal{I}(X_{ji} = x_i)$ (abbreviated to $\mathcal{I}_j(x_i)$) and $\mathcal{I}(Pa_{ji} = q_i)$ (abbreviated to $\mathcal{I}_j(q_i)$):

$$\mathcal{I}_j(x_i) = \begin{cases} 1, & \text{if } X_{ji} = x_i, \\ 0, & \text{otherwise.} \end{cases}, \text{ and } \mathcal{I}_j(q_i) = \begin{cases} 1, & \text{if } Pa_{ji} = q_i, \\ 0, & \text{otherwise.} \end{cases} \tag{6.20}$$

For a test vector $\vec{x}$ we simply omit the subscript j and write $\mathcal{I}(x_i)$ and $\mathcal{I}(q_i)$.

As already noted in Section 6.2.2, the MAP predictive distribution can be determined by computing the likelihood of a test vector $\vec{x} = (\vec{u}, \vec{v})$:

$$\mathcal{P}_{map}(\vec{x} \mid D) = P(\vec{x} \mid \breve{\theta}(D)) = \prod_{i=1}^{m} \prod_{q_i \in \mathbf{E}_{\mathrm{pa}_i}} \prod_{x_i \in \mathbf{E}_i} (\breve{\theta}^i_{x_i|q_i})^{\mathcal{I}(x_i)\mathcal{I}(q_i)}, \tag{6.21}$$

where $\mathcal{I}(x_i)$ and $\mathcal{I}(q_i)$ are indicator variables for the test vector $\vec{x}$ and $\breve{\theta}^i_{x_i|q_i}$ is the MAP model given by (see, for example, [71])

$$\breve{\theta}^i_{x_i|q_i} = \frac{f^i_{x_i|q_i} + \mu^i_{x_i|q_i} - 1}{\sum_{l=1}^{k_i} \left(f^i_{l|q_i} + \mu^i_{l|q_i} \right) - k_i},$$

Here $\mu^i_{x_i|q_i}$ are the hyperparameters as defined in Section 6.3.2 and $f^i_{x_i|q_i}$ are the sufficient statistics of the training data D: $f^i_{x_i|q_i}$ is the number of data instantiations where random variable X_i has taken on value x_i and the parents of X_i have configuration q_i:

$$f^i_{x_i|q_i} = \sum_{j=1}^{n} \mathcal{I}_j(x_i)\mathcal{I}_j(q_i)$$

With the uniform prior all the hyperparameters $\mu^i_{x_i|q_i}$ are set to 1 (Section 6.3.2), in which case we get the standard maximum likelihood estimator,

$$\hat{\theta}^i_{x_i|q_i} = \frac{f^i_{x_i|q_i}}{\sum_{l=1}^{k_i} f^i_{l|q_i}}.$$

6.4.2 The $\mathcal{P}_{av}$ Predictive Distribution for Bayesian Networks

The evidence predictive distribution (6.5) is defined as an integral over the parameter space. As shown in [29, 71], with Bayesian networks this integral can be solved analytically, yielding

$$\mathcal{P}_{av}(\vec{x} \mid D) = \prod_{i=1}^{m} \prod_{q_i \in \mathbf{E}_{\mathrm{pa}_i}} \prod_{x_i \in \mathbf{E}_i} (\bar{\theta}^i_{x_i|q_i})^{\mathcal{I}(x_i)\mathcal{I}(q_i)} \tag{6.22}$$

where

$$\bar{\theta}^i_{x_i|q_i} = \frac{f^i_{x_i|q_i} + \mu^i_{x_i|q_i}}{\sum_{l=1}^{k_i} \left(f^i_{l|q_i} + \mu^i_{l|q_i} \right)}.$$

From (6.22) we see that (perhaps somewhat surprisingly) similarly to the $\mathcal{P}_{map}$ predictive distribution case, the resulting predictive distribution can be regarded as a likelihood of the test vector $\vec{x}_t$, but now taken at the mean of the posterior rather than at the mode (maximum).

6.4.3 The $\mathcal{P}_{sc}$ Predictive Distribution for Bayesian Networks

The stochastic complexity predictive distribution is proportional to the likelihood of the combined data set $D^+ = D \cup \vec{x}$ at the maximum likelihood point:

$$\mathcal{P}_{sc}(\vec{x} \mid D) \propto P(D^+ \mid \hat{\theta}(D^+)) = \prod_{i=1}^{m} \prod_{q_i \in \mathbf{E}_{\mathrm{pa}_i}} \prod_{x_i \in \mathbf{E}_i} (\hat{\theta}^i_{x_i|q_i})^{(f^i_{x_i|q_i})^+},$$

where

$$\hat{\theta}^i_{x_i|q_i} = \frac{(f^i_{x_i|q_i})^+}{\sum_{l=1}^{k_i} (f^i_{l|q_i})^+},$$

and $(f^i_{l|q_i})^+$ are the sufficient statistics of D^+. Consequently, the predictive distribution can also in this case be regarded as a likelihood of the test vector $\vec{x}$, but the maximum likelihood estimator is now computed from the extended data set, consisting of the original data set together with the test vector itself.

6.4.4 The $\mathcal{P}_{jef}$ Predictive Distribution for Bayesian Networks

As can be seen from (6.9), Jeffreys' prior is proportional to the square root of the determinant of the Fisher information matrix $I(\theta)$. We proceed to compute this matrix. Observe that, for a Bayesian network distribution $P(\cdot|\theta) \in \mathcal{M}_{\mathcal{G}}$, the log-likelihood of a single data vector $\vec{x}$ can be written as

$$\log P(\vec{x} \mid \theta) = \sum_{i=1}^{m} \sum_{q_i \in \mathbf{E}_{\mathrm{pa}_i}} \mathcal{I}(q_i) \left(\sum_{x_i=1}^{k_i-1} \left(\mathcal{I}(x_i) \log \theta^i_{x_i|q_i} \right) + \mathcal{I}(k_i) \log \theta^i_{k_i|q_i} \right). \tag{6.23}$$

where $\mathcal{I}(q_i)$ and $\mathcal{I}(x_i)$ are defined as in (6.20).

Let us consider the element $(\theta^{i_1}_{l_1|q_{i_1}}, \theta^{i_2}_{l_2|q_{i_2}})$ of the second derivative (Hessian) matrix of (6.23). If either the variable indices i_1, i_2 or the parent configurations q_{i_1}, q_{i_2} are different, then clearly the second derivative is zero, and thus also the corresponding element of the information matrix is zero. It follows that the only non-zero elements of the information matrix are in sub-matrices where both parameters in question have the same variable and configuration index. Consider one of these sub-matrices $I^i_{q_i}(\theta)$, where i is the variable index and q_i the parent configuration. After some simple calculus we get

$$-\frac{\partial^2 \log P(\vec{x} \mid \theta)}{\partial \theta^i_{l_1|q_i} \partial \theta^i_{l_2|q_i}} = \begin{cases} \frac{\mathcal{I}(q_i)\mathcal{I}(k_i)}{(\theta^i_{k_i|q_i})^2}, \text{ if } l_1 \neq l_2, \\ \\ \frac{\mathcal{I}(q_i)\mathcal{I}(l_i)}{(\theta^i_{l_i|q_i})^2} + \frac{\mathcal{I}(q_i)\mathcal{I}(k_i)}{(\theta^i_{k_i|q_i})^2}, \text{ if } l_1 = l_2. \end{cases} \tag{6.24}$$

The elements of the Fisher information matrix are now the expectations of (6.24) over the set of all possible data vectors $\vec{x} \in \mathbf{E}$. For the case where $l_1 \neq l_2$ we get

$$\mathrm{E}_\theta \left[\frac{-\partial^2 \log P(\vec{x} \mid \theta)}{\partial \theta^i_{l_1|q_i} \partial \theta^i_{l_2|q_i}} \right] = \sum_{\vec{x} \in \mathbf{E}} P(\vec{x} \mid \theta) \frac{\mathcal{I}(q_i)\mathcal{I}(k_i)}{(\theta^i_{k_i|q_i})^2} = \tag{6.25}$$

$$= \frac{P(Pa_i = q_i, X_i = k_i \mid \theta)}{(\theta^i_{k_i|q_i})^2} = \frac{P(Pa_i = q_i \mid \theta)}{\theta^i_{k_i|q_i}}. \tag{6.26}$$

Similarly, with $l_1 = l_2$ we get

$$\mathrm{E}_\theta \left[\frac{-\partial^2 \log P(\vec{x} \mid \theta)}{\partial (\theta^i_{l|q_i})^2} \right] = \frac{P(Pa_i = q_i \mid \theta)}{\theta^i_{l|q_i}} + \frac{P(Pa_i = q_i \mid \theta)}{\theta^i_{k_i|q_i}}. \tag{6.27}$$

which gives the $(k_i - 1) \times (k_i - 1)$ sub-matrix $I^i_{q_i}(\theta)$ of the Fisher information matrix:

$$I^i_{q_i}(\theta) = \begin{pmatrix} (\frac{P^i_{q_i}}{\theta^i_{1|q_i}} + \frac{P^i_{q_i}}{\theta^i_{k_i|q_i}}) & (\frac{P^i_{q_i}}{\theta^i_{k_i|q_i}}) & \cdots & (\frac{P^i_{q_i}}{\theta^i_{k_i|q_i}}) \\ (\frac{P^i_{q_i}}{\theta^i_{k_i|q_i}}) & (\frac{P^i_{q_i}}{\theta^i_{2|q_i}} + \frac{P^i_{q_i}}{\theta^i_{k_i|q_i}}) & \cdots & (\frac{P^i_{q_i}}{\theta^i_{k_i|q_i}}) \\ \vdots & \vdots & \ddots & \vdots \\ (\frac{P^i_{q_i}}{\theta^i_{k_i|q_i}}) & (\frac{P^i_{q_i}}{\theta^i_{k_i|q_i}}) & \cdots & (\frac{P^i_{q_i}}{\theta^i_{k_i-1|q_i}} + \frac{P^i_{q_i}}{\theta^i_{k_i|q_i}}) \end{pmatrix}, \tag{6.28}$$

where $P^i_{q_i} = P(Pa_i = q_i|\theta)$. One can show by performing a sequence of Gaussian elimination steps on this matrix (see [18]) that its determinant is given by

$$|I^i_{q_i}(\theta)| = \frac{(P^i_{q_i})^{k_i-1}}{\prod_{l \in \mathbf{E}_i} \theta^i_{l|q_i}}. \tag{6.29}$$

The whole Fisher information matrix $I(\theta)$ is a block diagonal matrix, where the blocks are the sub-matrices $I^i_{q_i}(\theta)$. The determinant of a block diagonal matrix is the product of the determinants of the blocks, and thus

$$|I(\theta)| = \prod_{i=1}^{m} \prod_{q_i \in \mathbf{E}_{\mathrm{pa}_i}} \frac{(P^i_{q_i})^{k_i-1}}{\prod_{l \in \mathbf{E}_i} \theta^i_{l|q_i}}. \tag{6.30}$$

Finally, as noted in Section 6.2.4, $\pi(\theta) \propto \sqrt{|I(\theta)|}$, so we get

$$\pi(\theta) \propto \prod_{i=1}^{m} \prod_{q_i \in \mathbf{E}_{\mathrm{pa}_i}} (P^i_{q_i})^{\frac{k_i-1}{2}} \prod_{l \in \mathbf{E}_i} (\theta^i_{l|q_i})^{-\frac{1}{2}} \propto \prod_{i=1}^{m} \prod_{q_i \in \mathbf{E}_{\mathrm{pa}_i}} (P^i_{q_i})^{\frac{k_i-1}{2}} \prod_{l \in \mathbf{E}_i} (\theta^i_{l|q_i})^{-\frac{1}{2}}. \tag{6.31}$$

Computing Jeffreys' prior as formulated above requires computing for each variable the marginal distribution of its parents. Unfortunately, for multi-connected Bayesian networks, this problem is NP-hard [28]. In the experiments reported in Section 6.5, we used a simple tree-structured Bayesian network, in which case Jeffreys' prior is of a proper conjugate (Dirichlet) form, and it can be computed efficiently.

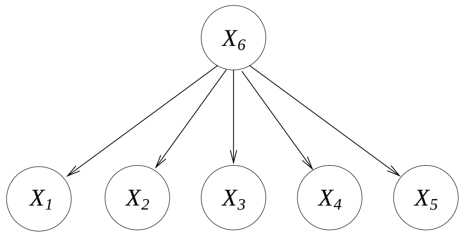


Figure 6.2: An example Naive Bayes structure. In this case, $m = 6$ and the feature variables are $X_1, \ldots, X_5$. *Given* the value of X_6 (the class value), the feature variables become independent.

6.5 Empirical Results

6.5.1 Experimental Setup

In our experiments, we concentrated on the standard classification problem, where the task is to predict the value of the single *classification variable* X_m, given the values of all the other variables. As explained in Section 6.2.1, this is achieved by computing probabilities of the form $\mathcal{P}(x_m \mid x_1, \ldots, x_{m-1}, D)$.

We used as our model class the class of Naive Bayes classifiers [42]. In a Naive Bayes model the clamped variables $X_1, \ldots, X_{m-1}$ are assumed to be independent, given the value of variable X_m. Consequently, we can regard the Naive Bayes model as a simple tree-structured Bayesian network, where variable X_m forms the root of the tree, and variables $X_1, \ldots, X_{m-1}$ are represented by the leaves. Figure 6.2 gives an example of a Naive Bayes structure. An informal introduction to the Naive Bayes model class was given on page 125.

Definition 6.7 (Naive Bayes) *Let* $\mathbf{E} = \mathbf{E}_1 \times \cdots \times \mathbf{E}_m$. *Let for* $1 \leq i \leq m-1$, $Pa_i = \{X_m\}$ *and let* $\mathcal{G} = \{Pa_1, \ldots, Pa_{m-1}\}$. *The Naive Bayes model class for feature variables* $X_1, \ldots, X_{m-1}$ *and class variable* X_m *is the class of Bayesian network distributions with structure* $\mathcal{G}$.

Let $\mathcal{M}_{nb}$ be the class of Naive Bayes models over $m-1$ feature variables and $K = k_m$ different class values. In this case the Jeffreys' prior formula (6.31) reduces to (writing θ^m_k as short for $\theta^m_{k|1} = P(X_m = k|\theta)$):

$$\begin{aligned}
\pi(\theta) &\propto \prod_{k=1}^{K} (\theta^m_k)^{-\frac{1}{2}} \prod_{i=1}^{m-1} \prod_{k'=1}^{K} (\theta^m_{k'})^{\frac{k_i-1}{2}} \prod_{l \in \mathbf{E}_i} (\theta^i_{l|k'})^{-\frac{1}{2}} \\
&= \prod_{k=1}^{K} (\theta^m_k)^{\frac{1}{2}(\sum_{i=1}^{m-1}(k_i-1)-1)} \prod_{i=1}^{m-1} \prod_{k'=1}^{K} \prod_{l \in \mathbf{E}_i} (\theta^i_{l|k'})^{-\frac{1}{2}}, \qquad (6.32)
\end{aligned}$$

which by Definition 6.5 can be seen to be a Dirichlet density.

For our experiments with the Naive Bayes classifier, eight public domain classification data sets of varying size were used (the data sets can be obtained from the UCI data repository[2]). Table 6.1 describes the size (n), the number of attributes (m), and the number of classes (K) for each of these data sets. All continuous attributes were discretized. To get an idea of how such a data set looks like we refer back to page 125 where the 'heart disease' set was discussed in some detail.

Dataset	Data vectors	Attributes	Classes	CV folds
Heart Disease (HD)	270	14	2	9
Iris (IR)	150	5	3	5
Lymphography (LY)	148	19	4	5
Australian (AU)	690	15	2	10
Breast Cancer (BC)	286	10	2	11
Diabetes (DB)	768	9	2	12
Glass (GL)	214	10	6	7
Hepatitis (HE)	150	20	2	5

Table 6.1: The datasets used in the experiments

For comparing the predictive accuracy of different predictive distributions, we used two different score functions: the *log-score* and the *0/1-score*. The log-score of a predictive distribution $\mathcal{P}(\cdot|\vec{u}, D)$ when the actual outcome is $X_m = k$, is defined as $-\log \mathcal{P}(X_m = k|\vec{u}, D)$. It thus coincides with the logarithmic loss; see Chapter 2, page 44 for several interpretations of this loss function.

For the 0/1-score, we simply first determine the k for which the probability $\mathcal{P}(X_m = k|\vec{u}, D)$ is maximized, and the 0/1-score is then defined to be 1, if the actual outcome indeed was k, otherwise it is defined to be 0.

Two separate sets of experiments were performed on each data set by using the following predictive inference methods with the Naive Bayes modelclass:

- ML: The $\mathcal{P}_{map}$ predictive distribution (6.3) with uniform prior (equivalent to the predictive distribution with the maximum likelihood model).
- EV: The $\mathcal{P}_{av}$ predictive distribution (6.5) with uniform prior.
- SC: The $\mathcal{P}_{sc}$ predictive distribution (6.8).
- EJ: The $\mathcal{P}_{jef}$ predictive distribution (6.12), which coincides with the $\mathcal{P}_{av}$ distribution (6.5) with Jeffreys' prior (6.32).

Note that all these predictive distributions are 'consistent' [164] in the following sense: in the hypothetical situation in which the data are truly generated by repeated sampling from a Naive Bayes model θ^*, then as D gets larger all four predictive distributions $\mathcal{P}(\cdot|D)$ will converge, with probability 1, to $P(\cdot|\theta^*)$. For the ML approach, this follows from the consistency of maximum likelihood estimators for exponential families (recall that Naive Bayes is an exponential family); see [164]. For the SC approach, we see from (6.8) and (6.7) that $\mathcal{P}_{sc}(\cdot|D)$ converges to $P(\cdot|\hat{\theta}(D))$ as the size of D increases.

[2] "http://www.ics.uci.edu/~mlearn/".

Consistency of the EJ and EV approach can be seen to follow from the consistency of the ML approach: compare the explicit formula for the ML predictive distribution $\mathcal{P}_{map}(\cdot \mid D)$ (Equation 6.21) with that of the EJ and EV approaches $\mathcal{P}_{av}(\cdot \mid D)$ (Equation 6.22) and observe that the estimates of the single parameters $\breve{\theta}^i_{x_i|q_i}$ and $\bar{\theta}^i_{x_i|q_i}$ occurring in these equations converge to the same value for increasing sample size.

We see that for very large samples all predictive distributions considered here are optimal in the sense that they are 'consistent'. Interesting differences will therefore only occur for samples of limited size. In that case, all four distributions have properties which make them an interesting choice to base predictions on: ML since it embodies the standard method of classical, 'Fisherian' statistics [48]; EV since the uniform prior is more or less the standard way of using the Bayesian evidence in discrete domains when there is no prior knowledge; SC since it is based on minimizing worst-case logarithmic loss; and EJ on the one hand since it is also based on minimizing worst-case logarithmic loss but is not dependent on knowledge of the sample size; and on the other hand because, according to some Bayesians, Jeffreys' prior, rather than the uniform one, is the sole prior expressing 'no prior knowledge' [17].

In the first set of experiments (Section 6.5.2) we measured the crossvalidated prediction performance by using the two score functions described above for training sets containing a substantial amount of data. In our second set of experiments (Section 6.5.3), we studied how the prediction quality of our various approaches depends on the size of the training set D.

6.5.2 Crossvalidation Results

In the machine learning literature, cross-validation has become a standard means of testing an algorithm's prediction and classification performance; here we briefly explain how k-fold cross-validation works; for more details about the procedure we refer to [22].

The k-fold cross-validation procedure for a sample D of size n starts by (randomly) partitioning D into k sets $D_1, \dots, D_k$ of size (as close as possible to) n/k. These sets are called the ***folds***. The idea is now to use the data in the union of $k-1$ of these sets as training data and to use the data in the remaining set as test data. We achieve this by setting $i := 1$ and determining the predictive distribution $\mathcal{P}(\vec{x}|D \setminus D_i)$ where $D \setminus D_i$ stands for the training sample D with the subsample D_i removed. We use this distribution to predict all items in D_i and we denote by $\textsc{score}(D_i)$ the accumulated score obtained in these predictions. For example, in case of the log-score, let $D_i = (\vec{x}_1, \dots, \vec{x}_{n_i})$. Then $\textsc{score}(D_i) := \textsc{loss}_{lg}(D_i) = -\sum_{1 \leq j \leq n_i} \log \mathcal{P}(\vec{x}_j|D \setminus D_i)$. We repeat this procedure to determine $\textsc{score}(D_i)$ for all $1 \leq i \leq k$. We then divide each score $\textsc{score}(D_i)$ by the number of items in D_i to obtain the average score for test-set D_i. Finally, we add the k average test-set scores and divide by k to obtain the *cross-validated prediction score.*

In our experiments, we initially used with each of the datasets the same number of folds as in the major experimental comparison performed in the Statlog project [111] (the number of folds used in each case can be found in Table 6.1). It should be noted though that although the result of one crossvalidation run is an average of n numbers, where n is the number of folds used, the result depends of course on how the n folds are selected from the sample data. To see how much the results vary with different fold partitionings, we performed 100 independent crossvalidation runs where the data was randomly partitioned into n folds, and computed the minimum, the average, and the maximum of the crossvalidated prediction accuracies obtained. As can be seen in Figures 6.3 (in the log-score case) and 6.4 (in the 0/1-score case), the crossvalidation results can vary quite a lot depending on the specific fold partitioning used.

Though the differences in crossvalidation results between different prediction methods are small, we see that for the log-score, evidence with uniform prior performs consistently better than the other methods, followed very closely by the evidence with Jeffreys' prior. The ML approach produces the worst results. For the 0/1-score, the picture is not as clear-cut. It should be noted that in this case the ML approach produces the best results with two of the data sets (GL and HE). A partial explanation for this fact may be that for the much coarser 0/1-score, it is in many cases not important exactly what probability we attach to a class value being k; all probability distributions over the class values for which k gets the maximum probability will lead to the same prediction. Thus it can very well happen that, while the ML prediction captures less well the regularities underlying the data (and hence performs worse with respect to log-score), it still captures them well enough to give maximum probability to the class value that should actually receive maximum probability.

In addition to comparison purposes between the different predictive distributions, the results in Figures 6.3 and 6.4 are interesting as they show surprisingly good performance of the Naive Bayes model when compared to the results reported in the machine learning literature. This highly competitive behaviour of 'Naive' Bayes has been noted before by several authors [50, 90].

The high variance of the results obtained indicate that one single n-fold crossvalidation run cannot be used as a reliable measure for comparing various predictive inference methods, unless the same specific fold partitioning is used in all cases. If, however, a number of independent runs is performed, then some statistical measure, such as the average, can be used for this purpose. Alternatively, the leave-one-out results[3] seem to follow the behavior of the averaged crossvalidation results quite accurately. For this reason, we decided to restrict ourselves to leave-one-out crossvalidation in the next section.

6.5.3 Results with Varying Amount of Training Data

To see how the prediction quality of our various approaches depends on the size of the training set D, we performed a set of experiments using only small fractions of the available training data. In these experiments, leave-one-out crossvalidation was used, but at each step, only the k first vectors from the training set were used in order to predict the single test vector that was "left out", and this procedure was repeated for $k = 1, \ldots, n-1$. As this setup is dependent on the ordering of the data vectors, the whole leave-one-out crossvalidation cycle was then repeated 100 times with 100 randomly generated permutations of the dataset. The averaged (over the 100 leave-one-out crossvalidation runs) results are plotted as a function of k in Figures 6.5–6.8

[3]'leave-one-out crossvalidation' over a data set of n observations is defined as n-fold crossvalidation over this data set.

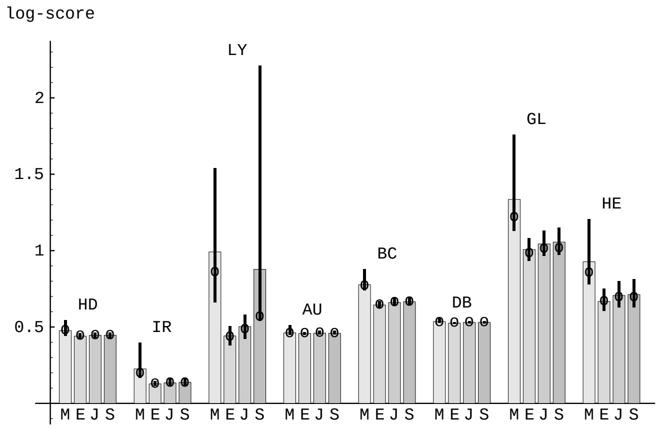


Figure 6.3: The minimum (lower end of the black line), the average (grey bar), and the maximum (upper end of the black line) of the crossvalidated log-score (= prediction error with logarithmic loss function) obtained by 100 independent crossvalidation runs. The corresponding leave-one-out crossvalidation results are marked with small circles. The y-axis represents the log-score, so the smaller the score, the better. The prediction methods used are ML (denoted here by M), EV (E), EJ (J), and SC (S).

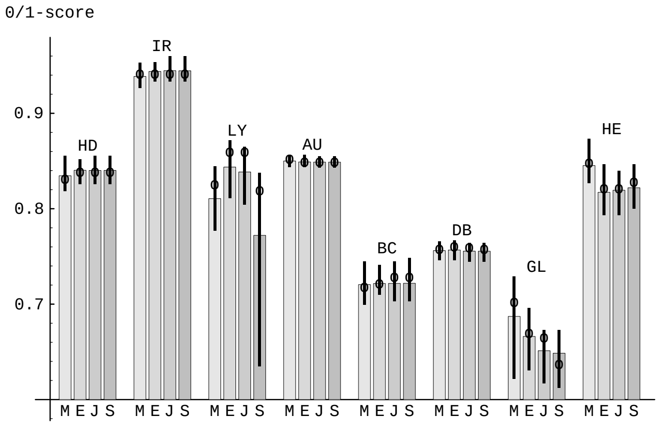


Figure 6.4: The minimum (lower end of the black line), the average (grey bar), and the maximum (upper end of the black line) of the crossvalidated 0/1-scores obtained by 100 independent crossvalidation runs. The corresponding leave-one-out crossvalidation results are marked with small circles. In this picture higher score is better. The prediction methods used are ML (denoted here by M), EV (E), EJ (J), and SC (S).

for the eight datasets used. These statistics of the behavior of different predictive distributions as a function of increasing amount of training data should now give us some idea as to the typical behavior of our prediction methods. For small sample sizes, the ML method will sometimes yield infinitely bad log-score. In order to prevent scaling problems when presenting the results graphically, the *pr-score* (the probability of the correct class, instead of its logarithm) was used in these tests as the alternative score for the 0/1-score, and not the log-score.

All in all, the results with all the eight data sets used show very similar behavior: the evidence-based EV and EJ approaches perform surprisingly well even in cases where the training data consists of only a few data vectors, which shows that the data sets used here are quite redundant, and when properly used, only a very small sample of these data sets is needed for constructing good models.

We now analyze three further interesting aspects of the results.

ML vs. the Rest It is a well-known fact that, for small sample sizes, the ML predictor is too dependent on the observed data and does not take into account that future data *may* turn out to be different. Our results support this observation and show that compared to the other methods, the ML predictive distribution appears to be much more sensitive to the amount of data available. This phenomenon can be explained by the fact that the EV and EJ approaches are more conservative methods as they base their predictions on averaging over all the parameter values, while ML makes more "eager" predictions based on the single maximum likelihood estimator. Let us consider a very simple example to illustrate this point. Suppose our data consists of a string of ones and zeros generated by some i.i.d. Bernoulli-process $p = P(X = 1)$. If we have seen an initial string consisting of one '1', and no zeros, then the ML predictor will determine that the probability of the second symbol being a '1' is unity. However, using the EV prediction, this probability is $\frac{2}{3}$. If the next data item turns out to be a '0', then the log-score of the ML (ML) predictor will be $-\infty$ while that of the EV will be $\log 2 - \log 3$. The behavior of the SC and EJ methods lies somewhere in between that of ML and EV. In our Bernoulli example, the probability of the second symbol being a '1' would be $\frac{3}{4}$ for EJ and $\frac{4}{5}$ for SC.

EJ vs. SC We see in the figures that for small sample sizes, EJ usually performs somewhat better than SC. Though this may have something to do with the fact that SC is not based on a probabilistic model (Section 6.2.4), we conjecture there is another reason: $\mathcal{P}_{sc}$ as we use it here is defined so that the method would give a maximally high probability for full unseen vectors $(x_1, \ldots, x_m)$, but the methods were tested in the restricted case, where only one variable (the class variable x_m) was actually predicted. The optimization of the SC method for the supervised classification case would require the use of a *conditional* maximum likelihood estimator of data D instead of the unconditional ML estimator used in equations 6.7 and 6.8. Using the notation of Section 6.2, this is the model θ in model class $\mathcal{M}$ that maximizes the *conditional* probability $P(\vec{v}_1, \ldots, \vec{v}_n | \vec{u}_1, \vec{u}_n, \theta)$. In other words, it is the model maximizing the probability of the sequence that is to be predicted conditioned on the sequence that is given. Apparently the evidence-based predictions are not so sensitive to exactly what

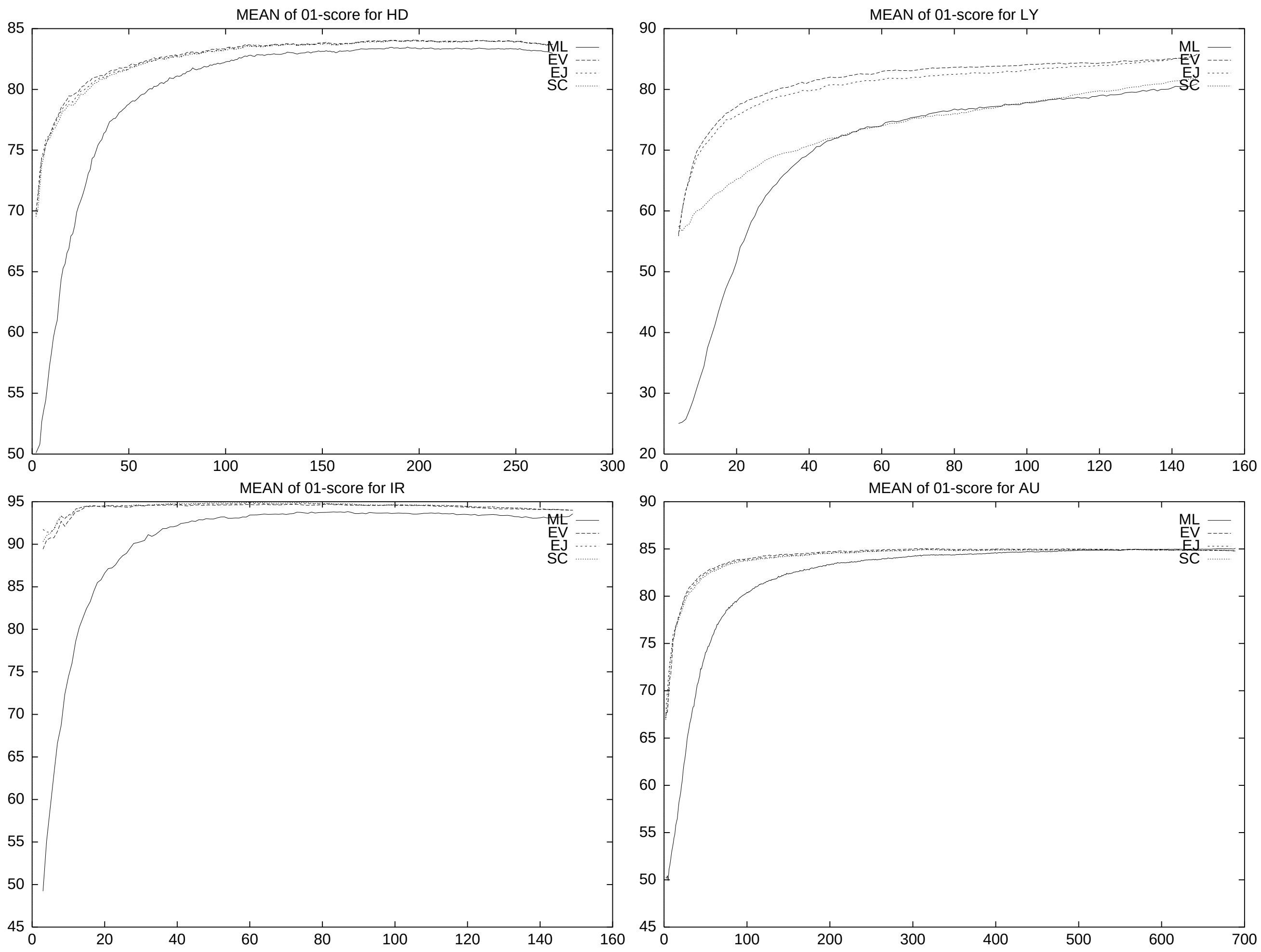


Figure 6.5: Average leave-one-out 0/1-scores obtained with different predictive distributions for the HD, IR, LY and AU dataset cases as a function of the number of the training examples used.

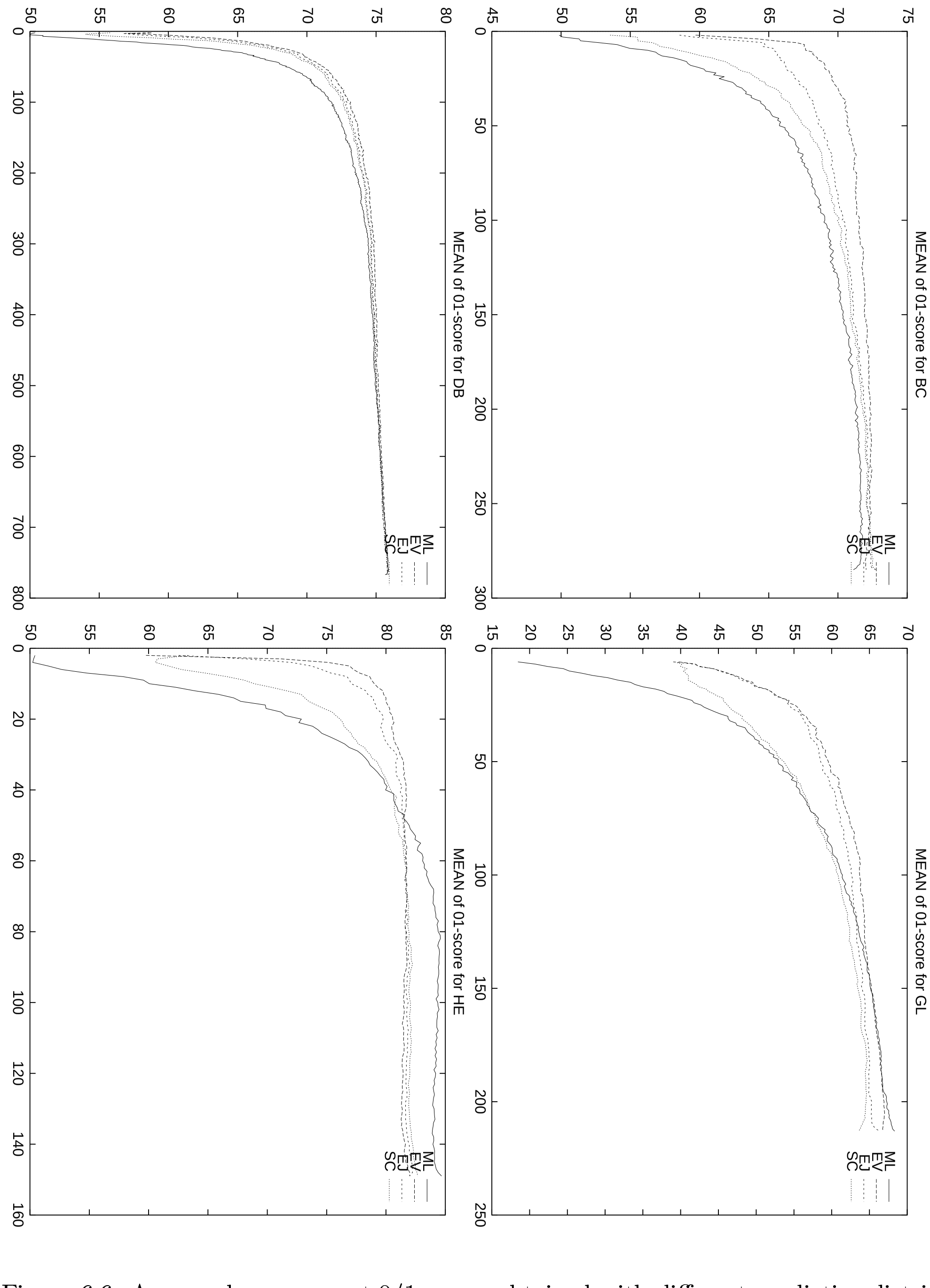


Figure 6.6: Average leave-one-out 0/1-scores obtained with different predictive distributions for the BC, DB, GL and HE dataset cases as a function of the number of the training examples used.

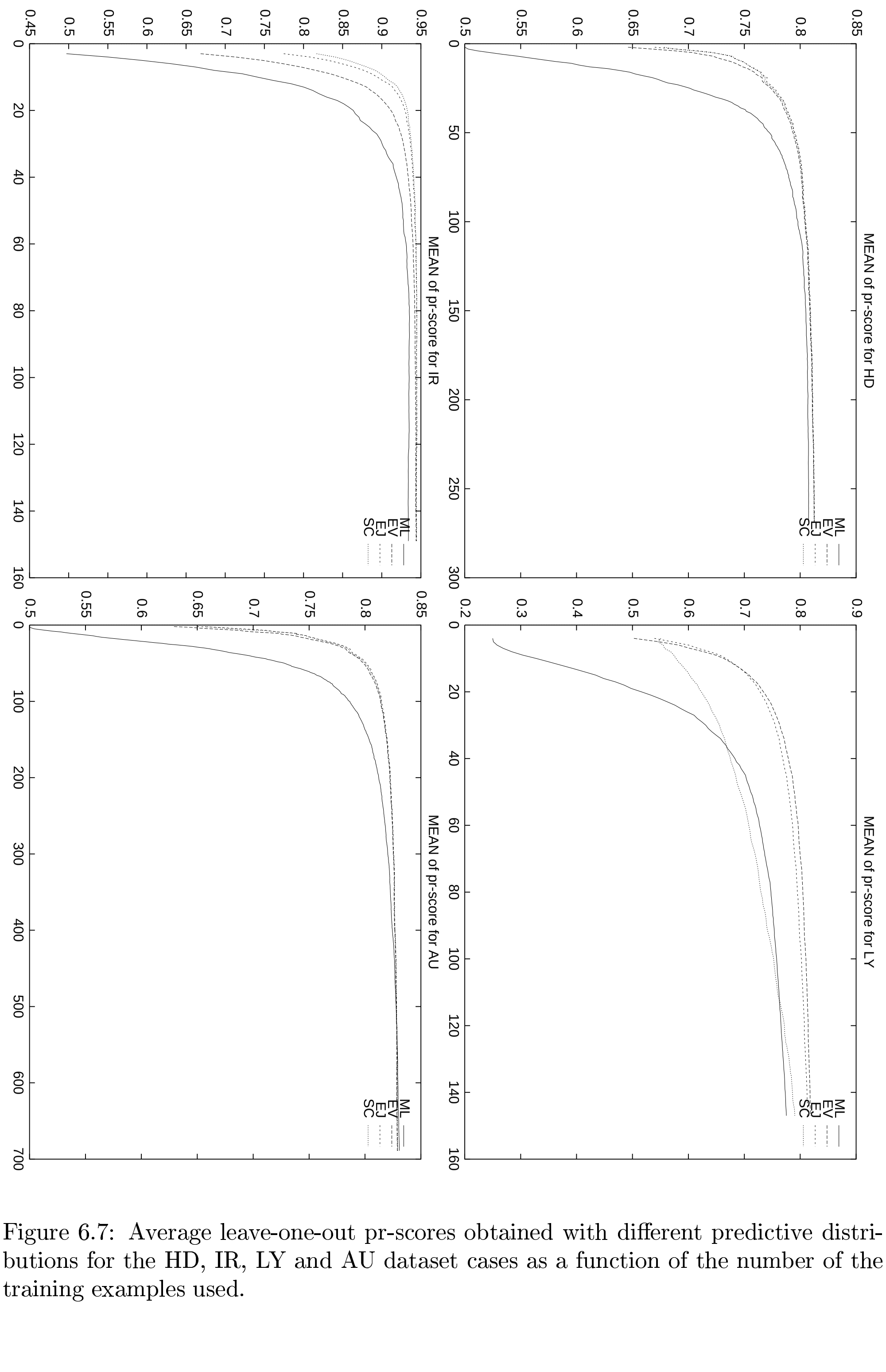


Figure 6.7: Average leave-one-out pr-scores obtained with different predictive distributions for the HD, IR, LY and AU dataset cases as a function of the number of the training examples used.

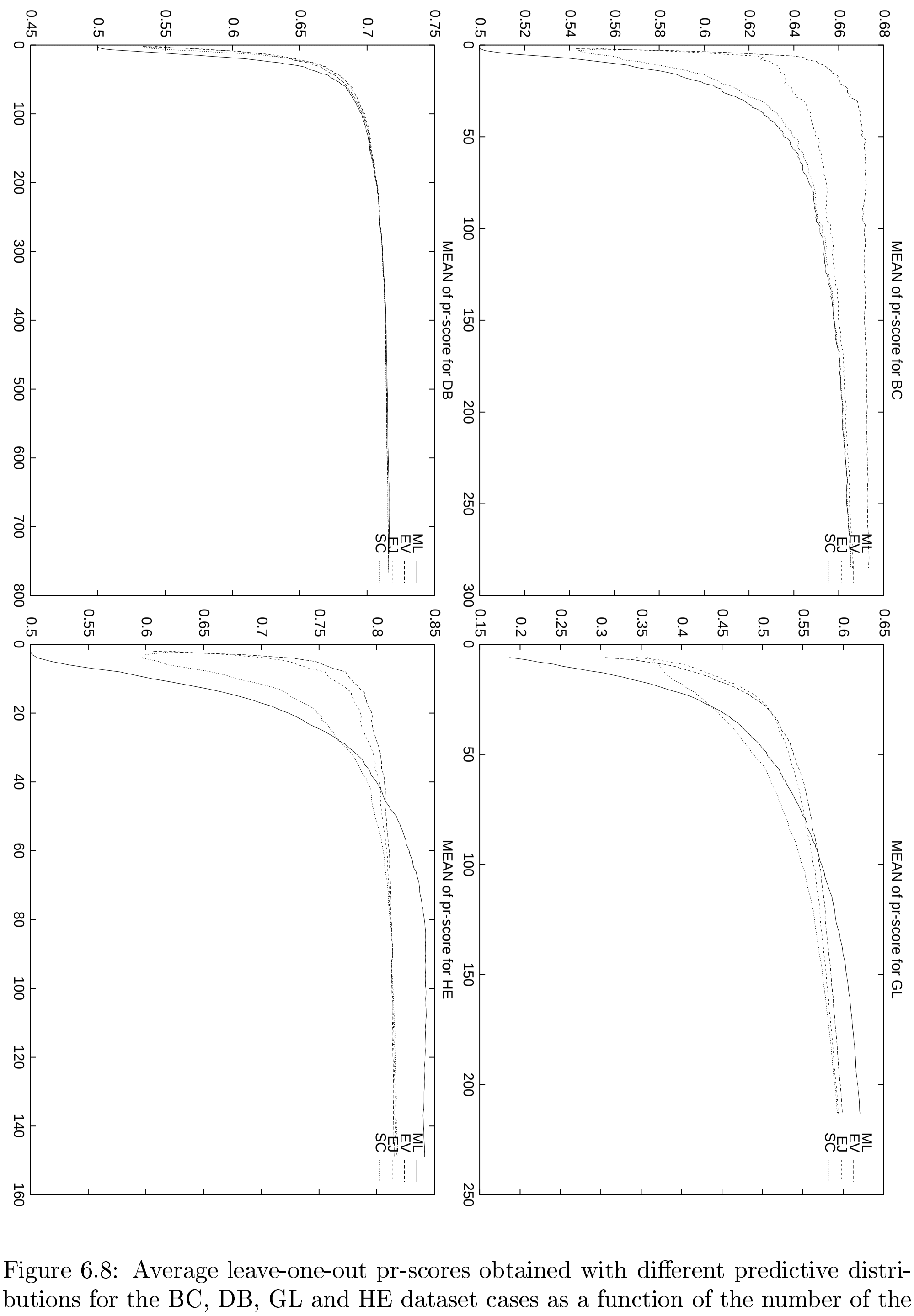


Figure 6.8: Average leave-one-out pr-scores obtained with different predictive distributions for the BC, DB, GL and HE dataset cases as a function of the number of the training examples used.

kind of predictive performance is being optimized. We have no proof of this; it is strongly suggested though by some additional experiments we performed but do not present here. In general, the non-asymptotic difference between EJ and SC seems an interesting topic for further research.

EV vs. EJ and SC Predictions based on EV usually give slightly better results than predictions based on EJ. This may be due to the fact that $-\log P_{jef}(D)$ approximates the stochastic complexity (Equation 2.7, page 2.7) and as such optimizes *worst-case* regret: *whatever* the data D is, $-\log P_{jef}(D)$ will be about equally close to $-\log P(D|\hat{\theta}(D))$, the best-fitting model in the class for D. For the model class of Bernoulli processes, Rissanen [129] showed that this leads to a considerable gain (in the sense of smaller number of bits needed to code the data and hence better predictions in the log-score sense) over $\mathcal{P}_{av}$ with the uniform prior for highly 'skewed' data sets. These are data sets for which the ML estimator lies near the boundary of the parameter space $\mathcal{M}$. The price to pay is that for all other, more 'average' data sets, using P_{jef} leads to a slightly larger code length than using P_{av} with the uniform prior. We conjecture that something similar is happening in our experiments in that the datasets are just not 'skewed' enough for EVP to outperform EV.

6.6 Conclusion

We have described how to obtain four different predictive distributions, one based on the Bayesian MAP estimator, one based on the Bayesian evidence distribution, one based on a direct but possibly non-optimal application of stochastic complexity and one based on a version of stochastic complexity that coincides with the evidence distribution using Jeffreys' prior. We have shown how to compute these predictive distributions for the model class of Bayesian networks.

In the experimental part of the chapter, the predictive accuracy of the ML predictive distribution (the MAP predictive distribution with uniform prior distribution), the evidence predictive distribution (with both uniform and Jeffreys' prior), and the stochastic complexity distribution was evaluated empirically by using publicly available classification data sets. For computational reasons, the specific model used in the tests was the structurally simple Naive Bayes model. In the experiments performed, in the 0/1-score case with substantial training data there was no clear winner. In the other three cases (0/1-score case with small amount of training data and log-score case with either substantial or small amount of data), the evidence-based predictive distributions outperformed the other methods. The effect of using either the uniform or Jeffreys' prior was extremely small, but fairly consistent in that it occurred in nearly every experiment we performed: the best results were obtained by using the uniform prior. One reason for the evidence predictive distribution performing just a little bit worse with Jeffreys' prior may be caused by the fact that Jeffreys' prior is in a sense a "worst-case" prior as we indicated in the last section.

The results with decreasing amount of training data show that the evidence-based approaches predict surprisingly well even with small training sets. The behavior of the SC method usually converges close to that of the EJ method as the training set size increases. As indicated in the last section, the somewhat worse results with small training sets may be explained by the fact that the SC predictive distribution was defined so that the method would give a maximally high probability for full unseen vectors, but the methods were tested in the restricted case, where only one variable (the class variable) was actually predicted.

As a final remark we repeat that the actual classification accuracies obtained with the Naive Bayes model are surprisingly high, when compared to the results obtained by alternative models. This has actually been noted before by several authors (see for example [90, 50]); the results reported here strengthen these previous conclusions in that they indicate that, when the evidence predictive distribution based on Naive Bayes is used, very good performance may already be achieved for quite small sample sizes. This suggests that in many natural domains, using the Naive Bayes model may not be so naive after all.

Chapter 7

MML vs. MDL

7.1 Introduction

In this chapter we compare MDL to the closely related *Minimum Message Length (MML)* principle. MML was introduced by Wallace and Boulton in 1968 [161], nine years before Rissanen independently introduced the MDL Principle [126]. Like MDL, MML is based on the idea that the more we are able to compress a given set of data, the more we have learned from that set of data. Nevertheless, as discussed in [15], there are subtle differences between these two approaches in both the underlying philosophy and the proposed formal criteria.

Recently [129], Rissanen has refined his MDL approach to incorporate effects on the description length of the data that are due to local geometrical properties of the hypothesis space. Wallace and Freeman (WF) already took these properties into account earlier, in their 1987 paper [162] on MML estimation. It has been informally claimed by several people at several conferences that a large part of Rissanen's 1996 work is already implicit in WF's 1987 paper. In this chapter we investigate this claim, and show that it does not hold: though superficially similar, the refinement of MDL proposed in [129] is quite different from the MML approach proposed in [162]. The difference is even quite dramatic in the sense that in the MDL approach the likelihood of the data, given a model θ, is multiplied by a factor correcting for the local structure of the model space near θ, while in the MML approach it is *divided* by the very same factor. Trying to account for this difference, we discovered an oversight in Wallace and Freeman's 1987 derivation of MML estimators. Based on our analysis, we present two revised versions of MML: a pointwise estimator which gives the MML-optimal single parameter model, and a volumewise estimator which gives the MML-optimal region in the parameter space. Of these estimators, the pointwise MML estimator turns out to coincide with the standard Bayesian MAP estimator, at least if a uniform 'subjective' prior is used. The volumewise MML estimator is found to be related to Rissanen's MDL estimator.

Our theoretical analysis indicates that both the revised MML estimators and the MDL approaches should lead to better predictive performance than the original MML estimator. However, these results (1) are asymptotic in nature (2) are strictly speaking only valid if the uniform prior is used as the subjective prior and (3) they are based on a worst-case analysis which may not necessarily be optimal for practical purposes. For these reasons it is not a priori clear what will happen in practical settings with small sample sizes. Therefore we studied this question empirically. The results suggest that with small data sets, the MDL approach yields more accurate predictions than both the MML estimators. The empirical results also demonstrate that the revised versions introduced here indeed perform better than the original Wallace and Freeman MML estimator.

We would like to emphasize at the outset that we do *not* claim that there is anything wrong with the MML principle per se: the problem we discovered concerns only what WF call 'MML estimators', which are an approximation of the theoretically optimal (but usually not efficiently computable) 'strict' MML estimators. Our results do not say anything about either the strict MML estimators or the MML principle in general.

7.1.1 Structure of this Chapter

Section 7.2 reviews the MML principle and discusses some differences and similarities between MML and MDL. Sections 7.3-7.5 contain our theoretical work. In Section 7.3, we review in detail how the MML estimators were derived in [162], and point out the oversight in the derivation. Based on this analysis, we present our two 'revised' versions of MML. In Section 7.4, we compare both the original and the revised MML estimators to Rissanen's recent [129] refinement of MDL.

In Section 7.5 we discuss how to construct different predictive distributions based on the MML and MDL estimators considered. In Section 7.6 we compare their performance on several real-world datasets. The empirical results obtained demonstrate differences in performance that are consistent with the theoretical analysis.

7.2 The MML Principle

7.2.1 Definitions

We assume throughout this chapter that all data is recorded to a finite accuracy, which implies that the set $\mathbf{E}$ of all possible data values is countable. Let the model class $\mathcal{M} = \{P(\cdot|\theta) \mid \theta \in \Gamma\}$ be finitely parameterized (Chapter 2, Definition 2.7) by Γ where Γ is a bounded region of $\mathbf{R}^k$ and k is a positive integer. For simplicity we assume that $\mathcal{M}$ consists only of i.i.d. probabilistic models.

Two-part Codes

Recall that a *two-part description method* (Chapter 1, page 17) consists of a code C_1 for parameter values ('hypotheses') and a set of codes $C_2(\cdot|\theta)$ for encoding data sequences *with the help of* those parameter values. Since the set of parameter values Γ is uncountable, the code C_1 cannot have codewords for all of them; rather, C_1 will be a function $C_1 : \ddot{\Gamma} \rightarrow \mathbf{B}^*$ where $\ddot{\Gamma}$ is some countable subset of Γ.

A data sequence x^n can be encoded in two steps by first encoding a parameter value $\ddot{\theta} \in \ddot{\Gamma}$, and then encoding x^n by the code $C_2(\cdot|\ddot{\theta})$. The number of bits needed to encode data x^n on the basis of model $\ddot{\theta}$ then becomes:

$$L_{1,2}(x^n; \ddot{\theta}) = L_{C_2}(x^n|\ddot{\theta}) + L_{C_1}(\ddot{\theta}) = -\log P(x^n|\ddot{\theta}) + L_{C_1}(\ddot{\theta}) \qquad (7.1)$$

7.2.2 The Minimum Message Length (MML) Principle

The basic idea behind MML modeling is to find a two-part coding system and an associated estimator minimizing the *expected* message length (number of bits needed to encode the data), where the expectation is taken over the Bayesian marginal distribution $P_{av}(\cdot|\mathcal{M}) = \int_{\theta \in \Gamma} P(x^n|\theta)w(\theta)d\theta$ (see Chapter 2, Section 2.8). Hence every MML analysis depends on a prior distribution w over the set of parameter values Γ. This prior is interpreted in a subjective Bayesian manner: it is taken to represent the prior knowledge one has about the parameter values [162].

MML thus seeks to find, for each n, the combination of (i) the subset $\ddot{\Gamma}$ of Γ, (ii) the code C_1 and (iii) the estimator $\ddot{\theta} : \mathbf{E}^n \to \ddot{\Gamma}$ minimizing the sum

$$\sum_{x^n \in \mathbf{E}^n} P_{av}(x^n)[-\log P(x^n|\ddot{\theta}(x^n)) + L_{C_1}(\ddot{\theta}(x^n))]. \qquad (7.2)$$

The estimator $\ddot{\theta}$ that is optimal in the above sense is called the *strict MML (SMML) estimator* [162]. In practice, it is very hard to find this SMML estimator. For this reason, Wallace and Freeman (WF) propose an approximation to the SMML estimator which they simply call the *MML estimator*. The derivation of the WF MML estimator can be found in Section 7.3.

7.2.3 Differences between MML and MDL

A crucial difference between the MDL and MML principles is that the latter is based on finding a code minimizing *expected* code lengths, while the former is based on finding a code that yields short code lengths for *all* datasets that are well-modeled by $\mathcal{M}$. Another important difference is that the goal of the MML approach is to find an efficient code together with the associated estimator, while MDL is not, in general, concerned with estimators. What is more, the MML approach uses for this purpose always two-part codes; for MDL there are several options, of which the two-part code MDL is only one special case.

It should also be noted that while MML is Bayesian in the sense that the approach is dependent on a subjective prior provided by an external observer, the MDL principle does not depend on any specific prior distribution. To be sure, priors do arise in MDL modeling, but they are merely used as technical tools and *not* as representing prior knowledge about the problem at hand.

7.3 MML Estimators

The strict MML estimator is usually very difficult to find, but there are several ways for constructing an MML estimator approximating SMML. In Section 7.3.1 we present the approximating estimator suggested by Wallace and Freeman in [162] which they simply call the 'MML estimator'. We will refer to it as the WF MML estimator of the WF estimator for short. Our critique towards the WF estimator concerns the code for encoding the parameter values, which is suboptimal. In Section 7.3.2 we present an alternative code that is optimal under worst-case assumptions. We show how using this code leads to a revised pointwise MML estimator. In Section 7.3.3 we go on to show that, using the revised MML estimator and looking at large sample sizes, the optimal MML *region* in the parameter space does not necessarily contain the single optimal revised MML estimator. Basing MML estimators on the optimal region in parameter space turns out to be closely related to Rissanen's 1996 formulation of MDL. Section 7.4 reviews this work and summarizes the differences between the WF MML estimator, the revised MML estimator, the region-based MML estimator and the recent form of MDL. The reader who only wants to get a quick look at the essential differences may wish to skip the remainder of this section and turn to Section 7.4 immediately.

7.3.1 The Wallace and Freeman MML Estimator

In this subsection we summarize the original derivation of the MML estimator as presented in [162]. We concentrate first on the case of a model class $\mathcal{M}$ containing models depending on a single parameter (hence $\Gamma \subset \mathbf{R}^1$). Rather than trying to find the strict-MML code optimizing (7.2), Wallace and Freeman [162] consider the following problem: given an observed data sequence x^n, we are asked to choose an estimate $\theta' \in \Gamma$ together with a *precision quantum* q, so that if x^n is encoded by first stating θ' with precision determined by q and then stating x^n using the code corresponding to the stated estimate, then the length of the encoding is minimized. This means that the estimate θ' is coded using only a limited number of binary places; in other words, a truncated value $\ddot{\theta}$ is obtained from θ' by selecting a value from a quantized scale in which adjacent values differ by q: $|\ddot{\theta} - \theta'| \leq q/2$. Note that the precision *quantum* q gives the *width* between adjacent parameter values, whereas the word 'precision' per se is used in this thesis to denote the number of *bits* needed to encode a parameter value.

While in the SMML setup the goal was to minimize (7.2), now we only ask for a 'target' estimate θ' together with a precision quantum q. Consequently, in contrast to SMML, here we do not require the detailed coding of the actually used estimate $\ddot{\theta}$ to be specified. This makes the approach feasible; the price we pay is that the exact effect of encoding the data using the quantized value $\ddot{\theta}$ instead of θ' cannot be predicted, and the code length can be minimized only in expectation.

We assume that the quantization has the following effect:

$$E[\theta' - \ddot{\theta}] = 0 \qquad \text{(the quantization is unbiased)} \tag{7.3}$$
$$E[(\theta' - \ddot{\theta})^2] = q^2/12 \qquad \text{(as for a uniform distribution)} \tag{7.4}$$

Let w be the prior density of the parameters $\theta \in \Gamma$. The prior probability that θ lies within $\pm q/2$ of a quantized value $\ddot{\theta}$ is approximately $q \cdot w(\ddot{\theta})$. Let $C_{\mathrm{wf}} : \ddot{\Gamma} \to \mathbf{B}^*$ be the code corresponding to this probability. Encoding the estimates $\ddot{\theta}$ using C_{wf}, the expected length of the first part of the message stating θ' to precision quantum

q is $-\log qw(\theta')$. In the second part of the message, we code x^n using the code corresponding to $\ddot{\theta}$, which requires $-\log P(x^n|\ddot{\theta})$ bits.

The length of the second part can be approximated by the following Taylor series expansion:

$$\begin{aligned}-\log P(x^n|\ddot{\theta}) &= -\log P(x^n|\theta') \\ &- (\theta'-\ddot{\theta})\frac{\partial}{\partial\theta}\log P(x^n|\theta') - \frac{1}{2}(\theta'-\ddot{\theta})^2\frac{\partial^2}{\partial\theta^2}\log P(x^n|\theta') + \dots \quad (7.5)\end{aligned}$$

Using the expectation of the effects of quantization as given by Equation (7.3) and (7.4), the total length is expected to be (to second order)

$$-\log qw(\theta') - \log P(x^n|\theta') + \frac{q^2}{24}I(x^n;\theta'), \tag{7.6}$$

where $I(x^n;\theta')$ is short for $-\partial^2 \log P(x^n|\theta')/\partial\theta^2$. The expected code length (7.6) is minimized by choosing

$$q^2 = \frac{12}{I(x^n;\theta')}. \tag{7.7}$$

Substituting this optimal precision quantum, we get the expected code length to be

$$-\log w(\theta') + \frac{1}{2}\log\frac{I(x^n,\theta')}{12} - \log P(x^n|\theta') + \frac{1}{2}. \tag{7.8}$$

The value θ' which minimizes this is called the *MML estimate*.

It is clear that in order to decode a two-part message as used here, one must first decode the parameter value $\ddot{\theta}$. For this, one must know the precision quantum q that was used to encode $\ddot{\theta}$. Since the optimal quantum depends on x^n, it is not constant and hence it seems that it must be made part of the code too (see Chapter 1, page 19). However, WF showed that the minimum of the expected message length reached for the optimal precision quantum $q^2 = 12/I(x^n,\theta')$ is very broad with respect to q. This implies that using a quantum q based on the *expectation* of $I(x^n,\theta')$, rather than $I(x^n,\theta')$ itself, will be reasonably efficient for most data values. Hence we can use $q^2 = 12/I_n(\theta')$, where $I_n(\theta')$ is the Fisher (expected) information for n observations:

$$I_n(\theta) = -E_\theta\left[\frac{d^2\log P(x^n|\theta)}{d\theta^2}\right]. \tag{7.9}$$

The advantage of using $I_n(\theta')$ is that now the optimal q is independent of the observed data and becomes a function of θ' only.

This means that there is only one set of possible truncated estimates which can be constructed without reference to the data. We can thus construct a code for the estimate which does not need a precision quantum preamble (for more details about the whole derivation, we refer to [162]). From (7.8) we see that the final definition of the WF MML estimator, from now on denoted by θ'_{wf}, becomes

$$\theta'_{\text{wf}} = \arg\max_{\theta\in\Gamma}\frac{P(x^n|\theta)w(\theta)}{\sqrt{I_n(\theta)}}. \tag{7.10}$$

7.3.2 The MAP/ML Estimator as a Revised Pointwise MML Estimator

In this subsection we point out an oversight in Wallace and Freeman's derivation of MML estimators as given in the previous subsection. For the special case where a uniform prior is used, we present an alternative derivation that leads to a revised MML estimator which coincides with the Bayesian MAP (maximum a posteriori; see Chapter 6, Section 6.2.2) estimator. Since the uniform prior is used the MAP estimator coincides with the ML estimator.

Let us define $q : \Gamma \to \mathbf{R}$ as a function which gives for each value θ' the corresponding optimal precision quantum $q(\theta')$. Using this notation, and substituting $I_n(\theta')$ for $I(x^n; \theta')$ (as prescribed at the end of the previous section), the expected total length (7.6) can be rewritten as

$$L_{\text{mml-wf}}(x^n; \theta') \approx -\log q(\theta') - \log w(\theta') - \log P(x^n|\theta') + \frac{q(\theta')^2}{24} I_n(\theta'). \tag{7.11}$$

We now make two assumptions. First, we assume that the value θ' which minimizes the expected code length may in principle lie anywhere in the interior of the parameter space. Second, we assume that the number of different possible truncated parameter values is finite, say N. Consequently, we can write $\ddot{\Gamma} = \ddot{\Gamma}_N = \{\ddot{\theta}_1, \dots, \ddot{\theta}_N\}$. Both assumptions are quite reasonable. For example, the first assumption follows from the requirement that θ' should be consistent, together with the (much stronger) assumption that there exists a true value θ according to which data are actually drawn, and which may lie at any point in the interior of Γ. The WF MML estimator as given in Equation (7.10) is indeed consistent for all combinations of priors and i.i.d. model classes $\mathcal{M}$ for which the Bayesian maximum posterior (MAP) estimator is consistent. In the case of the WF MML estimator, the reason is that the influence of the denominator $\sqrt{I_n(\theta)}$ vanishes as n gets larger. It will be seen that the two revised MML estimators we introduce below are also consistent whenever the MAP estimator is consistent.

The second assumption (finite number of truncated parameter values) is reasonable as long as we allow N to grow with the number of observations n, as we indeed do. We only assume that for each fixed n there is a finite number of candidates N.

Now the point that (in our view) has been overlooked by WF is that nobody *forces* us to code the parameters using the code C_{wf} with lengths $-\log q(\theta') - \log w(\theta')$. Another code may sometimes, or even under fairly general circumstances lead to shorter codelengths. Specifically, the expected extra number of bits you need to code the *data* given a truncated model $\ddot{\theta}$ instead of θ' necessarily depends on the precision quantum $q(\theta')$. But why should we let the encoding of the parameters themselves depend on the $q(\theta')$? Below we present an alternative code where $q(\theta')$ does not influence the codelength of the parameters. The code will be optimal under worst-case assumptions. In our derivation, we will only consider the special case that w is the uniform prior ($w(\theta) = c$ for some constant c). In this case, our alternative code coincides with the *uniform code* C_{uni} rather than the code C_{wf}. The uniform code is simply the code that codes each element of $\ddot{\Gamma}_N$ using the same number of bits $\log N$. It follows directly from the Kraft inequality (Chapter 1, Theorem 1.6) that for every other code $C' : \ddot{\Gamma}_N \rightarrow \mathbf{B}^*$, we have $\max_{\ddot{\theta}_i \in \ddot{\Gamma}_N} L_{C'}(\ddot{\theta}_i) > \max_{\ddot{\theta}_i \in \ddot{\Gamma}_N} L_{C_{\text{uni}}}(\ddot{\theta}_i)$. We say that C_{uni} has the *optimal worst-case code length* (here 'optimal' is used in the sense of 'shortest').

Using C_{uni} instead of C_{wf}, the code length to encode $\ddot{\theta}$ becomes $\log N$ instead of $-\log q(\theta) \cdot w(\theta) = -\log q(\theta) \cdot c$, and the right hand side of (7.11) becomes

$$\log N - \log P(x^n|\theta') + \frac{q(\theta')^2}{24} I_n(\theta'). \tag{7.12}$$

For some θ' this will yield shorter code lengths than (7.11) while for others it will yield larger ones. However, using our assumption that θ' may in principle lie everywhere in the interior of Γ, it makes sense to take a worst-case point of view. We will take this view for granted for the remainder of this section; whether it can always be justified will be discussed in Section 7.3.4.

For the worst-case θ', the expected code length (7.12) is clearly smaller than the expected length (7.11) with $w(\theta)$ instantiated to c. According to the worst-case viewpoint, C_{uni} should then be preferred over C_{wf}, and indeed over every other possible code over the set $\ddot{\Gamma}_N$.

Using C_{uni} and denoting the total length needed to code x^n by $L_{\text{mml-p}}(x^n, \theta')$, Equation 7.12 becomes

$$L_{\text{mml-p}}(x^n, \theta') \approx \log N - \log P(x^n|\theta') + \frac{q(\theta')^2}{24} I_n(\theta'). \tag{7.13}$$

We see from this equation that using C_{uni} for encoding the parameter values is worst-case optimal independently of the way $q(\theta')$ is instantiated. Hence we should base our two-part codes on (7.13) rather than (7.11), and furthermore instantiate (7.13) by using the function $q(\theta')$ that gives shortest expected code lengths. Since we assume that θ' may lie everywhere in Γ, the optimal $q(\theta')$ becomes the function that minimizes the maximum value of the last term in (7.13):

$$q = \arg\min_{q} \max_{\theta' \in \Gamma} q(\theta')^2 I_n(\theta'), \tag{7.14}$$

which is clearly attained for $q(\theta')^2 \propto I_n(\theta')^{-1}$ (choosing $q(\theta') = 0$ everywhere is not an option since we assume that there exist only N parameter values). By substituting this optimal q back into (7.13), we obtain

$$L_{\text{mml-p}}(x^n, \theta') \approx -\log P(x^n|\theta') + \log N + K, \tag{7.15}$$

where K depends only on N and n, and not on θ. The θ' which minimizes this, however, is simply the standard Bayes posterior mode (or equivalently, the ML estimator); from now on we denote it by θ'_{p}:

$$\theta'_{\text{p}} = \arg\max_{\theta \in \Gamma} P(x^n|\theta) \tag{7.16}$$

Thus, interestingly, we find that in our alternative derivation of MML estimators, if the uniform prior is used, then the optimal MML estimate is just the (Bayesian) MAP estimate. On the other hand, as we see from (7.14), the optimal precision quantum $q(\theta'_{\mathrm{p}})$ at point θ'_{p} remains inversely proportional to $\sqrt{I_n(\ddot{\theta})}$, just like in the original derivation by WF:

$$q(\theta'_{\mathrm{p}}) \propto \frac{1}{\sqrt{I_n(\theta'_{\mathrm{p}})}} \tag{7.17}$$

7.3.3 A Volumewise MML Estimator

Using C_{uni}, we can code the data by first stating a $\ddot{\theta}_i \in \ddot{\Gamma}_N$ using $\log N$ bits, and then stating x^n using $-\log P(x^n|\ddot{\theta}_i)$ bits. Using Bayes' rule, this can equivalently be recast as determining the posterior probability of $\ddot{\theta}_i$ given data x^n, using the uniform discrete prior $W(\ddot{\theta}_i) = 1/N$. We denote this posterior probability by P_W:

$$P_W(\ddot{\theta}_i|x^n) \propto P(x^n|\ddot{\theta}_i). \tag{7.18}$$

In this probabilistic formulation, (7.16) tells us that the *single* value θ'_{p} which maximizes the expected value of $P_W(\ddot{\theta}_i|x^n)$ (where $\ddot{\theta}_i$ is the truncated version of θ'_{p}), is given by the ML estimate. However, it tells us nothing about the width of the maximum attained at θ'_{p}. As we shall see, its width may depend on the region in the parameter space where θ'_{p} lies. It may therefore be more interesting to choose a small (but non-zero) width s, and look for the interval in Γ of width s with the maximal posterior probability mass (or, equivalently, the shortest code length) according to (7.18).

To obtain this interval, let us adapt an idea introduced by Rissanen in [129] in a somewhat different context, and look at the MML two-part code in another manner. We partition the parameter space Γ into a set of adjacent regions $R_1, \ldots, R_M$, each of width s, where s is such that $M \ll N$. Let us now determine the region R_i with maximum posterior probability mass $P_W(R_i|x^n)$. We first associate with each region R_i the element θ_i that lies in the center of R_i, so $R_i = [\theta_i - s/2, \theta_i + s/2]$. We can now extend the density $P(x^n|\theta_i)$, determined by a single value θ_i, to a probability determined by a region in the parameter space R_i in the obvious way by defining $P(x^n|R_i) = \int_{R_i} P(x^n|\theta)\pi(\theta)d\theta$, where π is an arbitrary proper prior with support R_i. In the limit for small s, we have $P(x^n|R_i) = P(x^n|\theta_i)$. This implies

$$P_W(R_i|x^n) \propto P(x^n|R_i)W(R_i) \quad (\text{if } s \text{ is small}), \tag{7.19}$$

where W is the uniform prior. Marginalizing over the values $\ddot{\theta} \in \ddot{\Gamma}_N$ contained in R_i, we find that

$$W(R_i) = \frac{|R_i \cap \ddot{\Gamma}_N|}{N}. \tag{7.20}$$

In the limit for large N, we may select s as small as we want so that for all regions R_i, we can regard $P(x^n|\theta)$ and $\sqrt{I_n(\theta)}$ as approximately constant for all θ within the single interval R_i. For the optimal precision quantum $q(\theta)$, we have by Equation (7.17) that $q^2(\theta) \propto I_n(\theta)^{-1}$. Basing $\ddot{\Gamma}$ on this optimal precision quantum, it follows that in the limit for large N the density of parameter values $\ddot{\theta}$ in region R_i becomes proportional to $\sqrt{I_n(\theta_i)}$:

$$|R_i \cap \ddot{\Gamma}_N| \approx s\sqrt{I_n(\theta_i)} \cdot c \quad \text{for } N \text{ large}, \tag{7.21}$$

where c is a constant not depending on i. In the limit for large N, we have from (7.20) and (7.21) that

$$W(R_i) \propto s\sqrt{I_n(\theta_i)}. \tag{7.22}$$

We now conclude from (7.19), together with the fact that s does not depend on i, that

$$P_W(R_i|x^n) \propto P(x^n|\theta_i)\sqrt{I_n(\theta_i)}.$$

The region R_i which maximizes this is the most probable posterior region if the uniform prior is chosen. Assuming that I_n is a continuous function of θ, the $\theta \in \Gamma$ yielding the shortest expected code length *in its neighborhood* is thus given by

$$\theta'_{\mathrm{v}} = \arg\max_{\theta \in \Gamma}\{P(x^n|\theta)\sqrt{I_n(\theta)}\}. \tag{7.23}$$

We call this estimator the *revised volumewise MML estimator*. We see that in our alternative MML derivation, choosing the parameter value with the highest probability content in its neighborhood (7.23) gives us an estimate which maximizes the likelihood *times* $\sqrt{I_n(\theta)}$. This is in sharp contrast with the original MML estimate (7.10) which maximizes the likelihood *divided* by $\sqrt{I_n(\theta)}$! There is a caveat here; see the Discussion below. For simplicity, our derivations above have been only for the case where $\Gamma \subset \mathbf{R}^1$. The generalization to the multiple-parameter case is completely straightforward: it suffices to replace the intervals R_i of width s by *rectangles* R_i of *volume* s. In this case, the square root of the Fisher information $\sqrt{I_n(\theta)}$ becomes $\sqrt{|I_n(\theta)|}$, the square root of the determinant of the Fisher information matrix.

7.3.4 Discussion

Let us summarize what our foregoing derivations show and what they do not show. We *did* show in Section 7.3.2 that in general there is no reason to do as Wallace and Freeman and code parameter values using the code C_{wf} with expected lengths $-\log q(\theta') - \log w(\theta')$: another way of coding the parameter values may be better.

We provided an alternative coding scheme, but we did *not* show that it is in general better than Wallace and Freeman's code: we only showed that it leads to optimal estimators under *worst-case* assumptions. We should mention that there is a weakness[1] in the derivation of θ'_{v} as given by (7.23): in the derivation, we assumed that the optimal precision quantum was given by (7.17). However, (7.17) was derived in the derivation of the pointwise, not the volumewise MML estimator. It is not clear whether this remains the optimal precision quantum for the volumewise analysis. Hence, θ'_{v} has been proven to be 'volumewise optimal' *only* under the condition that the precision quantum (7.17) is used.

[1]Thanks to Kenji Yamanishi for pointing this out.

7.3.5 A Bold Assumption

We have derived our revised MML estimators θ'_{p} and θ'_{v} only for the special case of the uniform prior $w(\theta) = c$. In the following we shall simply *assume* that the derivations can also be done for the case of non-uniform priors. The definitions of θ'_{p} and θ'_{v} are therefore extended in the obvious way:

$$\theta'_{\mathrm{p}} = \arg\max_{\theta \in \Gamma} P(x^n|\theta)w(\theta) \tag{7.24}$$

$$\theta'_{\mathrm{v}} = \arg\max_{\theta \in \Gamma} \{P(x^n|\theta)w(\theta)\sqrt{I_n(\theta)}\}. \tag{7.25}$$

We should note that we have not proven the validity of these two formulas. However, all priors we shall use in our experiments are ESS-priors (see Section 7.5.2). As we shall see, using ESS-priors for data D is *always* equivalent to using the uniform prior for an extended data set $D \cup D^+$ (hence the ESS-priors embody some 'virtual data' on top of the actually given data). This gives an extra indication that the formulas (7.24) and (7.25) are theoretically valid for use in our experiments. Nevertheless, this is a very weak spot in our analysis, to which we shall return in the Conclusion.

7.4 Refined MDL Two-part Codes and their Relation to the Three MML Estimators

7.4.1 1996 MDL Two-part Codes

In Chapter 2, Section 2.1 we showed that the 2-part code $L_{2\text{-}p}$ is redundant. We used this fact as a motivation for introducing the stochastic complexity code as the code with the minimal worst-case regret (see Definition 2.8). The stochastic complexity code is 'one-part': no single parameter value is encoded explicitly. In his 1996 paper Rissanen [129] showed that one can revise the two-part code in such a way that the inherent redundancy gets removed, but the coding is still two-stage: first a parameter value is encoded, then the data is encoded with the help of that parameter. While the resulting code still has larger regret than the stochastic complexity code C_{sc} for all finite n, in the limit for large n the code lengths coincide.

We give a brief sketch of Rissanen's argument. For simplicity we assume that $\ddot{\Gamma}$ contains a finite number of parameter values, say N. We can thus write $\ddot{\Gamma} = \ddot{\Gamma}_N = \{\ddot{\theta}_1, \ldots \ddot{\theta}_N\}$. We also assume uniform code lengths for the models: $L_{C_1}(\ddot{\theta}_i) = \log N$ for all $\ddot{\theta}_i \in \ddot{\Gamma}_N$. Rissanen observed that a decoder, after having decoded $\ddot{\theta}_i$, already knows something about the data x^n whose description will follow. Namely, she knows that x^n must be a member of a proper subset $\mathbf{D}_i$ of the set of all possible data $\mathbf{E}^n$. This $\mathbf{D}_i$ is the set of all data x^n for which $\ddot{\theta}_i$ gives the shortest two-part code length:

$$\begin{aligned}
\mathbf{D}_i &= \{x^n \in \mathbf{E}^n \mid \ddot{\theta}_i = \arg\min_{\theta \in \ddot{\Gamma}_N}\{L_{C_1}(\theta) - \log P(x^n|\theta)\}\} \\
&= \{x^n \in \mathbf{E}^n \mid \ddot{\theta}_i = \arg\max_{\theta \in \ddot{\Gamma}_N}\{P(x^n|\theta)\}\}.
\end{aligned}$$

The reason for the decoder knowing that $x^n \in \mathbf{D}_i$ after decoding $\ddot{\theta}_i$ is the following: looking at Equation (7.1), we see that if $x^n \notin \mathbf{D}_i$, then the decoder would not have decoded $\ddot{\theta}_i$, but rather some other $\ddot{\theta}_j \neq \ddot{\theta}_i$. This fact can be exploited to change the code $C_2(\cdot|\theta)$ that was used in the original two-part code (7.1), to a code $C'_2(\cdot|\theta)$ with strictly shorter lengths. Using $C'_{2,\theta}$, we code x^n not by the code corresponding to probability distribution $P(x^n|\ddot{\theta}_i)$, but rather by the code based on the normalized probability distribution

$$\frac{P(x^n|\ddot{\theta}_i)}{\sum_{x^n \in \mathbf{D}_i} P(x^n|\ddot{\theta}_i)}.$$

In this case the total description length becomes

$$\log N - \log P(x^n|\ddot{\theta}_i) + \log \sum_{x^n \in \mathbf{D}_i} P(x^n|\ddot{\theta}_i)$$

rather than just $\log N - \log P(x^n|\ddot{\theta}_i)$ bits. In general, $\sum_{x^n \in \mathbf{D}_i} P(x^n|\ddot{\theta}_i) < 1$, which implies that the revised two-part code has a strictly shorter code length than the original one.

Rissanen showed [129] that the normalization trick described above can be optimally exploited (for large N) if, for every $\theta \in \Gamma$, the density of parameter values in $\ddot{\Gamma}_N$ in the neighborhood of θ is proportional to $\sqrt{|I(\theta)|}$, where $|I(\theta)|$ is the determinant of the Fisher information matrix $I(\theta) = I_1(\theta)$ as given by Equation (7.9) . This means that either the spacing between any two adjacent values $\ddot{\theta}_i$ and $\ddot{\theta}_{i+1}$ in $\ddot{\Gamma}_N$ should be made proportional to $1/\sqrt{|I(\theta)|}$, or the code giving code lengths $\log N$ to every $\ddot{\theta}_i \in \ddot{\Gamma}_N$ should be changed. Rissanen chooses the second option, but explicitly mentions that the first one is possible too [129, page 43].

7.4.2 Relation between the Four Approaches

We mentioned in the introduction that it has been – wrongly – claimed that Rissanen's considerations in [129] are already implicit in Wallace and Freeman's work on MML estimators. We now discern a possible reason for this confusion: according to Wallace and Freeman, the optimal precision quantum (width between adjacent parameter values in $\ddot{\Gamma}$) for encoding parameter values in the neighborhood of a point $\theta' \in \Gamma$ is given by $q(\theta') \propto 1/\sqrt{|I_n(\theta')|}$ (Equation 7.17). For i.i.d. model classes, $|I_n(\theta)| = n|I_1(\theta)|$ for all $\theta \in \Gamma$ [17]. This implies that for such model classes, both according to Rissanen (see above) and according to Wallace and Freeman one should pick the width between two adjacent parameter values proportional to $1/\sqrt{|I_n(\theta')|}$. In this sense they seem to reach the same conclusion. However, this same width was chosen for a very different reason. What is more, as we shall see in Sections 7.5 and 7.6, making predictions of future data on the basis of Wallace and Freeman's (1987) MML-estimators can be quite different from making predictions on the basis of Rissanen's 1996 refinement of MDL.

To understand how the same optimal precision can lead to different estimators and predictions, it is instructive to study the graph of $\sqrt{|I(\theta)|}$. As an example, in Figure 7.1 we depict this graph for the case of the Bernoulli model class. Observe that the optimal

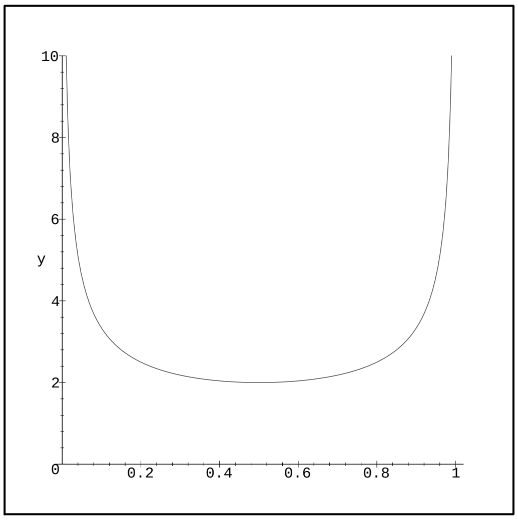


Figure 7.1: $\sqrt{|I(\theta)|}$ as a function of θ for the Bernoulli model defined by $P(X = 1) = \theta$. This graph can be interpreted as giving for each θ the optimal density of quantized parameter values $\ddot{\theta}$ in the region around θ.

width between two truncated parameter values at point θ is inversely proportional to $\sqrt{|I(\theta')|}$. Therefore, the optimal *density* of truncated parameter values near θ is proportional to $\sqrt{|I_n(\theta')|}$. The graph of $\sqrt{|I(\theta)|}$ as a function of θ can therefore be interpreted as giving, for each θ, the density of the parameters in $\ddot{\Gamma}_N$ in the region around θ. In Wallace and Freeman's derivation of MML estimators, a 'subjective' prior density w defined for all θ is implicitly transformed into a probability mass function W defined for all $\ddot{\theta} \in \ddot{\Gamma}_N$. In our analysis we restrict ourselves to the situation where the subjective prior density w is chosen to be uniform (which is the only situation for which we have actually proven anything). In this case, in Wallace and Freeman's original derivation, it is implicitly assumed that regions R in the parameter space of equal volume should obtain equal prior probability mass $\sum_{\ddot{\theta} \in R \cap \ddot{\Gamma}_N} W(\ddot{\theta})$. If we choose $\ddot{\Gamma}_N$ with the optimal widths between the parameter values, then in regions with high $\sqrt{|I(\theta)|}$, there will be many truncated values $\ddot{\theta} \in \ddot{\Gamma}_N$. To obtain equal prior densities in regions of equal volume, the prior probability mass assigned to a *single* $\ddot{\theta} \in \ddot{\Gamma}_N$ must be made *inversely* proportional to $\sqrt{|I(\theta)|}$.

In contrast, both in our volumewise MML-estimator (with uniform subjective prior) and in Rissanen's revised two-part code all *single* truncated parameter values $\ddot{\theta} \in \ddot{\Gamma}_N$ receive *equal* prior probability $1/N$. In this case, for large N, the graph of $\sqrt{|I(\theta)|}$ can also be interpreted as giving, for each θ, the prior density of the region around θ: the prior probability becomes proportional, instead of inversely proportional, to $\sqrt{|I(\theta)|}$.

7.5 MML and MDL in Practice

According to both the MML and MDL *Principles*, it should be the case that the more we compress the data at hand, the better we are able to predict future data coming from the same source. We therefore expect the following: θ'_{v} gives better predictions than θ'_{p}, which in turn should give better predictions than θ'_{wf}. From an MDL (not MML) point of view we additionally expect that prediction based on P_{jef}, the evidence distribution with Jeffreys' prior will work even better, since P_{jef} can be interpreted as a form of stochastic complexity, and hence leads to maximal compression of the data; see Chapter 6, Section 6.2.4.

However, there are two possible caveats: first, as discussed in Section 7.3.4, θ'_{v} and θ'_{p} will only give shorter codelengths than θ'_{wf} under worst-case assumptions regarding the value of θ'_{p}, and it is not clear whether these are justified. Second, for the i.i.d. model classes we have $|I_n(\theta)| = n|I_1(\theta)|$. This implies that the influence of the 'correction factors' $\sqrt{|I_n(\theta')|}$ and $1/\sqrt{|I_n(\theta')|}$ in equations 7.23 and 7.10, respectively, becomes negligible as n grows to infinity. Therefore, asymptotically predictions based on θ'_{wf}, θ'_{p} and θ'_{v} all coincide with the Bayesian MAP prediction, which in turn asymptotically coincides with Bayesian evidence prediction for all reasonable priors w; see Chapter 6, Section 6.5.1. Consequently, the differences between the three MML estimators and the MDL (stochastic complexity) approach are relevant only for small sample sizes. Unfortunately, since both Rissanen's [129] and our derivations are asymptotic in nature, they do not say very much about this situation. It is therefore an interesting empirical question, whether either Rissanen's MDL approach or our revised MML estimators lead to a more accurate predictive distribution than the WF estimator in cases where only a limited amount of data is available. Below we describe how the four methods considered can be used to arrive at predictive distributions. In the next section, we study the predictive performance of these different predictive distributions empirically by using small, real-world datasets.

7.5.1 Predictive Distributions based on MML and MDL

Henceforth we assume the setting of Chapter 6, Section 6.2.1: our outcome space $\mathbf{E}$ can be written $\mathbf{E} = \mathbf{E}_1 \times \ldots \times \mathbf{E}_n$ and outcomes are written as *vectors* $\vec{x} \in \mathbf{E}$. Recall from Chapter 6 that in prediction and classification problems for discrete data, we are given some training data D which we use to arrive at predictions of a single *test vector* $\vec{x}$. The predictions are done using some *predictive distribution* $\mathcal{P}$. We now define the predictive distributions based on the three different MML estimators:

$$\mathcal{P}_{\text{mml-wf}}(\vec{x}|D) := P(\vec{x}|\theta'_{\mathrm{wf}}(D)) \tag{7.26}$$

$$\mathcal{P}_{\text{mml-p}}(\vec{x}|D) := P(\vec{x}|\theta'_{\mathrm{p}}(D)) = \mathcal{P}_{map}(\vec{x}|D) \tag{7.27}$$

$$\mathcal{P}_{\text{mml-v}}(\vec{x}|D) := P(\vec{x}|\theta'_{\mathrm{v}}(D)) \tag{7.28}$$

The MDL-based predictive distribution $\mathcal{P}_{\mathrm{mdl}}$ is defined to be the evidence predictive distribution with the prior set to Jeffreys' prior $\pi(\theta) \propto \sqrt{|I(\theta)|}$. In Chapter 6, Section 6.2.4, Equation 6.12 this distribution was introduced under the name $\mathcal{P}_{jef}$. Hence:

$$\mathcal{P}_{\mathrm{mdl}}(\vec{x}|D) := \mathcal{P}_{jef}(\vec{x}|D) = \frac{P_{jef}(\vec{x}, D)}{P_{jef}(D)} \tag{7.29}$$

Keeping D fixed and regarding $\mathcal{P}_{\mathrm{mdl}}(\cdot|D)$ as a function of $\vec{x}$, the denominator in (7.29) becomes a constant and we see that

$$\mathcal{P}_{\mathrm{mdl}}(\vec{x}|D) \propto \int P(\vec{x}, D|\theta)\pi(\theta)d\theta \tag{7.30}$$

It is interesting to contrast this with the WF MML estimator and the volumewise MML estimator when a uniform subjective prior w is used. In that case:

$$\begin{aligned} \theta'_{\mathrm{v}}(D) &= \arg\max_{\theta} \{P(D|\theta)\pi(\theta)\} \quad \text{(if } w \text{ uniform)} & (7.31) \\ \theta'_{\mathrm{wf}}(D) &= \arg\max_{\theta} \{P(D|\theta)\frac{1}{\pi(\theta)}\} \quad \text{(if } w \text{ uniform)} & (7.32) \end{aligned}$$

and we see that, while the MDL predictions are based on the Bayesian evidence-predictions with Jeffreys' prior, the volumewise MML predictions are equivalent the Bayesian MAP-predictions with Jeffreys' prior and the WF MML predictions are based on the MAP-predictions with an inverse Jeffreys' prior.

7.5.2 The Model Class and the Subjective Prior

We wanted to apply our predictive distributions for a model class $\mathcal{M}$ sophisticated enough to yield good predictive performance in principle, while simple enough to allow the predictive distributions to be computed efficiently. We decided to use the Naive Bayes model class which has exactly these properties. We introduced Naive Bayes in the previous chapter (page 146) where (Equation 6.32) we also showed how to compute $\sqrt{|I(\theta)|}$ for this class. We repeat the formula for Jeffreys' prior $\pi(\theta)$ and for $\sqrt{|I(\theta)|}$ where θ is a member of the class of Naive Bayes models for sample space $\mathbf{E} = \mathbf{E}_1 \times \ldots \times \mathbf{E}_m$; for details about the notation we refer to the previous chapter):

$$\pi(\theta) \propto \sqrt{|I(\theta)|} = \prod_{k=1}^{K} (\theta_k^m)^{\frac{1}{2}(\sum_{i=1}^{m-1}(k_i-1)-1)} \prod_{i=1}^{m-1} \prod_{k'=1}^{K} \prod_{l=1}^{k_i} (\theta^i_{l|k'})^{-\frac{1}{2}} \tag{7.33}$$

From Equation (7.29) we see $\mathcal{P}_{\mathrm{mdl}} = \mathcal{P}_{jef}$. This implies that the instantiation of $\mathcal{P}_{jef}$ for the Naive Bayes case has been calculated already in Chapter 6. It is given by Equation (6.22).

The MML predictive distributions depend on a subjective prior. As candidate priors we used the so-called *ESS (Equivalent Sample Size) priors* [70]. ESS priors have a clear interpretation in terms of prior knowledge: ESS priors are indexed by a *virtual data set* D'. The ESS(D')-prior is defined such that, for every data set D and every test vector $\vec{x}$, the evidence prediction $P_{av}(\vec{x}|D)$ with prior ESS(D') is identical to the maximum likelihood prediction $P(\vec{x}|\hat{\theta}(D \cup D'))$. In other words, the ESS(D')-prior can be regarded as a 'summary' of data D': if, in a previous experiment, data D' was observed, then the corresponding prior can be used to combine this with the data D from the present experiment. Here we consider $\mathrm{ESS}(D')$ only for the case where D' contains all possible $\vec{x} \in \mathbf{E}$ equally often, say f times. We will allow fractional f. It is straightforward to show the following proposition; we omit the proof. In the proposition we freely use the notation introduced in Chapter 6, Sections 6.3 and 6.5.1.

Proposition 7.1 *Let $\mathcal{M}_{\mathcal{G}}$ be a Naive Bayes model class for random variables $X_1, \ldots, X_m$. Let*

$$a = \prod_{i=1}^{m-1} k_i \quad \textit{and} \quad b_i = \frac{\prod_{j=1}^{m-1} k_j}{k_i}.$$

The ESS prior $w(\theta)$ corresponding to a sample containing each $x \in \mathbf{E}$ exactly f times is given by

$$w(\theta) \propto \prod_{k=1}^{K} (\theta_k^m)^{f \cdot a - 1} \prod_{i=1}^{m-1} \prod_{k'=1}^{K} \prod_{l=1}^{k_i} (\theta_{l|k'}^i)^{f \cdot b_i - 1} \tag{7.34}$$

Consequently, the ESS-prior densities we consider are proportional to a product of parameters, which means they are Dirichlet densities (Definition 6.5). From (7.33) we see that $\sqrt{|I(\theta)|}$ is also proportional to a product of the parameters. The factors by which the likelihood is multiplied in determining the three MML estimators are $w(\theta)(\sqrt{|I(\theta)|})^{-1}$ for θ'_{wf}, $w(\theta)$ for θ'_{p} and $w(\theta)\sqrt{|I(\theta)|})$ for θ'_{v} respectively. Equations (7.33) and (7.34) together imply that these factors can be written as a simple product of parameters. Therefore, they may themselves be treated as Dirichlet densities. This means that the predictive distributions $\mathcal{P}_{\text{mml-wf}}$, $\mathcal{P}_{\text{mml-p}}$ and $\mathcal{P}_{\text{mml-v}}$ may all be re-interpreted as MAP predictive distributions with Dirichlet priors. We can therefore directly apply the formula (6.21) (page 143) to compute all three MML predictive distributions.

Why we cannot use the uniform prior Let

$$\mathrm{WF}(\theta) = P(x^n|\theta)w(\theta)(\sqrt{|I(\theta)|})^{-1}.$$

θ'_{wf} is equal to the θ maximizing $\mathrm{WF}(\theta)$. Observe that $\mathrm{WF}(\theta)$ can be written as a simple product of parameters. We see from (7.33) that the exponent of θ_k^m in $(\sqrt{|I(\theta)|})^{-1}$ may become negative up to an amount of $-\frac{1}{2}(\sum_{i=1}^{m-1}(k_i - 1) - 1)$. Therefore, even with reasonably large training set sizes, $\mathrm{WF}(\theta)$ may still contain two factors $\theta_{k_1}^m$ and $\theta_{k_2}^m$ with a negative exponent. In such a case, $\mathrm{WF}(\theta)$ does not have a maximum. To see this, fix all components of θ except $\theta_{k_1}^m$ and $\theta_{k_2}^m$. We can now view $\mathrm{WF}(\theta)$ as a function of $\theta_{k_1}^m$ as follows: as $\theta_{k_1}^m$ is varied, all parameter values in θ remain constant except $\theta_{k_1}^m$ and $\theta_{k_2}^m$. The latter must be varied along to keep $\sum_i \theta_i^m = 1$. We see immediately that

$$\lim_{\theta_{k_1}^m \downarrow 0} \mathrm{WF}(\theta) = \infty \;\text{ and }\; \lim_{\theta_{k_1}^m \uparrow G} \mathrm{WF}(\theta) = \infty \text{ where } G = 1 - \sum_{i \neq k_1, k_2} \theta_i^m$$

while for $0 < \theta_{k_1}^m < G$, $\mathrm{WF}(\theta)$ is a continuous function of $\theta_{k_1}^m$. It follows that the MML WF estimator θ'_{wf} is undefined.

In particular, this odd behaviour occurs if we pick as our subjective prior $w(\theta)$ the uniform prior. Therefore we decided to use in our experiments only ESS priors, where the fraction f was chosen to be the smallest possible number with which the above mentioned technical difficulty does not occur. Experiments with different ESS subjective priors seemed to produce similar results.

7.6 Empirical Results

For comparing empirically the four predictive distributions discussed in the previous section, we turned once more to the six public domain classification data sets that were already used in Chapter 6. These datasets are described in detail in Table 6.1 on page 147; see also page 125. We performed two sets of experiments.

Experiments with 0/1-Score

In the first set of experiments, we computed the crossvalidated 0/1-scores for each of the four methods by using 5-fold crossvalidation (following the testing scheme used in [50]). The definition of 0/1-score and n-fold crossvalidation can be found in Section 6.5.2 of the previous chapter. As the results appeared to be strongly dependent on the way the data was partitioned in the 5 folds to be used, we repeated the whole crossvalidation cycle 10000 times with different, randomly chosen partitionings of data. The results are given in Table 7.1. In the MML-WF case the predictive distribution $\mathcal{P}_{\text{mml-wf}}$ was used; MML-P stands for $\mathcal{P}_{\text{mml-p}}$, MML-V stands for $\mathcal{P}_{\text{mml-v}}$ and MDL stands for $\mathcal{P}_{\text{mdl}}$.

We can observe that, first of all, with respect to the 0/1-score, there seems to be

	MIN				MEAN				MAX			
	MML				MML				MML			
	WF	P	V	MDL	WF	P	V	MDL	WF	P	V	MDL
AU	83.5	**83.6**	83.5	**83.6**	**84.9**	**84.9**	84.8	**84.9**	**86.2**	86.1	86.1	**86.2**
DB	73.4	**73.6**	73.3	73.2	**75.5**	75.4	75.4	75.3	77.1	77.2	77.2	**77.3**
GL	56.5	56.5	56.1	**58.4**	62.6	62.6	62.7	**64.9**	67.3	68.2	67.3	**70.1**
HD	**81.9**	**81.9**	**81.9**	81.1	**84.5**	84.4	84.4	84.1	**87.0**	**87.0**	**87.0**	**87.0**
IR	**93.3**	92.7	92.7	92.0	**94.4**	94.3	94.3	**94.4**	96.0	96.0	**96.7**	**96.7**
LY	**79.1**	78.4	78.4	**79.1**	**84.2**	84.0	83.6	83.9	**88.5**	87.8	**88.5**	**88.5**

Table 7.1: Classification 0/1-scores with 10000 independent 5-fold crossvalidation runs.

no clear winner between the different predictive distributions used, and the differences between the results are usually small. Secondly, for all data sets used, the average results obtained here (i.e. the figures in the middle column) are very close to the optimal results for the Naive Bayes class that can be found in the literature (which are often arrived at using other inference methods). In two cases ('diabetes' and 'heart disease') our results seem to be optimal. For extensive references on results obtained by other methods we refer to [154] and [50]. Apparently, the minimum encoding approaches work very well independently of exactly which of the four versions is used.

Experiments with log-score

In the second set of experiments, instead of predicting only the value of the class variable X_m, we used the predictive distributions for computing the joint probability for the unseen testing vectors as a whole. In this case there is no useful analogue to the 0/1-score, so the accuracy of the methods was now measured by using the log-score, which is simply the logarithmic loss; see Chapter 2, page 44 for several interpretations of this loss function.

To prevent the large fluctuation in the results, we used in this experiment the leave-one-out form of crossvalidation, where the task is at each stage to predict one data vector, given all the others. The results of this experiment can be found in Table 7.2. From these results we can now see that the MDL approach produced consistently the best score, and of the MML estimators considered here, the MML-V estimator was more accurate than the MML-P estimator, which performed better than MML-WF (with the exception of the Lymphography database).

	10% training data				100% training data			
	MMLWF	MMLP	MMLV	MDL	MMLWF	MMLP	MMLV	MDL
AU	16.68	16.61	16.54	**14.98**	14.64	14.63	14.62	**14.44**
DB	13.80	13.79	13.77	**13.65**	13.25	13.25	13.24	**13.23**
GL	14.22	14.19	14.13	**12.14**	11.38	11.34	11.30	**10.25**
HD	12.99	12.95	12.90	**12.41**	11.75	11.74	11.73	**11.67**
IR	4.34	4.22	4.08	**3.60**	3.20	3.17	3.14	**3.07**
LY	19.27	19.27	19.25	**16.78**	15.85	15.89	15.90	**14.73**

Table 7.2: Leave-one-out crossvalidated log-scores in the joint probability estimation task. The lower the log-score, the better the predictive performance.

To study the small sample behavior of the methods in more detail, we rerun the leave-one-out crossvalidation experiments, but used at each stage only s (randomly chosen) vectors of the available $n-1$ vectors for producing the predictive distribution, where s varied between 1 and $n-1$. In this case, the results obtained are quite similar with all six datasets: the results with $s = 0.1n$ can be found in Table 7.2.

In figures 7.2-7.4 we plotted the behavior of the methods as a function of the amount s of training data. In these figures, the log-scores are scaled with respect to the score produced by the MMLWF method so that the MMLWF method gets always a score 0, and a positive score means that the actual log-score was better than the MMLWF log-score by the corresponding amount.

From figures 7.2-7.4 we now see two interesting things: firstly, the different predictive distributions seem to converge with increasing amount of training data, as was expected from the discussion in Section 7.5. Secondly, the relative differences between the methods seem to grow when the amount of available data is decreased. We see in particular that for *extremely* small sample sizes, the three MML approaches are comparable while the MDL approach may even be a bit worse. Then, for still very (but not extremely) small sample sizes all datasets show the same pattern: MDL outperforms MML-V, which in turn outperforms MML-P which in turn outperforms MML-WF. As the sample size grows, the ordering between the four approaches remains unchanged (except for the lymphography database) but the differences become smaller.

The fact that the results are consistent over datasets pertaining to six completely different domains suggests that the differences between the various approaches presented here may be practically significant in cases with small amount of data.

We note that the difference in performance between MDL and the three MML approaches is probably largely due to the fact that the MDL predictive distribution is based on 'using all models in the class at once' (which is achieved by averaging over the models) while the other three are based on prediction using a *single* model; we saw in the previous chapter that, with the uniform prior, there is, for very small sample sizes, much to be gained by predicting using the evidence and predicting using a single model (Section 6.5.3). Different experiments we reported elsewhere [87] suggest the same phenomenon. This means that the results reported here cannot be simply interpreted as telling us that 'MDL works better than MML'. They do suggest the following though: the influence of using Fisher information in (what we consider) the wrong way, as done in MML-WF, versus not using it at all (as in MML-P = MAP) vs. using it in the 'right' way (as done in MML-V) is really very small, even for small sample sizes. Small as it is though, it has fairly consistent effects (MML-V performing better than MML-P performing better than MML-WF) and is thus in agreement with our asymptotic theoretical analysis which depended on taking a worst-case point of view and assuming a uniform subjective prior.

Looking at figures 7.2-7.4 again, we hypothesize that for extremely small sample sizes, our asymptotic results simply give no evidence; as more data arrives, we enter a region where they become significant and the performance of the three MML methods is as predicted by the asymptotic theory. Then, as the sample size grows truly large, the law of large numbers 'takes over' and the differences between the three methods become negligible.

7.7 Conclusion and Future Work

We have shown that the claimed similarity between Wallace and Freeman's MML approach and Rissanen's 1996 MDL approach is superficial, and that when applying the approaches for predictive modeling, we arrive at quite different methods in practice. We pointed out a technical oversight in the derivation of the WF MML estimator and introduced two revised versions of the MML estimator, of which the volumewise optimal estimator was shown to be related to Rissanen's MDL estimator.

Theoretically, the revised MML estimators should allow for more compression of the data than the WF MML estimator, at least in a worst-case sense. As this was theoretically shown to be the case only asymptotically and only for uniform subjective priors, this raised the question of the practical small sample behavior of these methods. To be able to study this question empirically, we tested the different prediction methods for the Naive Bayes model class.

In the empirical tests performed, it was observed that while in simple classification tasks the methods showed quite similar performance, in joint probability distribution estimation the MDL approach produced consistently the best results. Moreover, the revised MML estimators introduced here usually gave better results than the WF MML estimator. The differences were largest for small amounts of training data (except for *extremely* small ones). They became smaller with an increasing amount of training data, as was expected from the theory.

Future Work In future work we hope to be able to show that our formulas for the two revised MML estimators are also valid for subjective non-uniform priors. Also, it would be interesting to consider alternative, more sophisticated ways of testing the small sample behaviour of the predictive distributions based on these estimators.

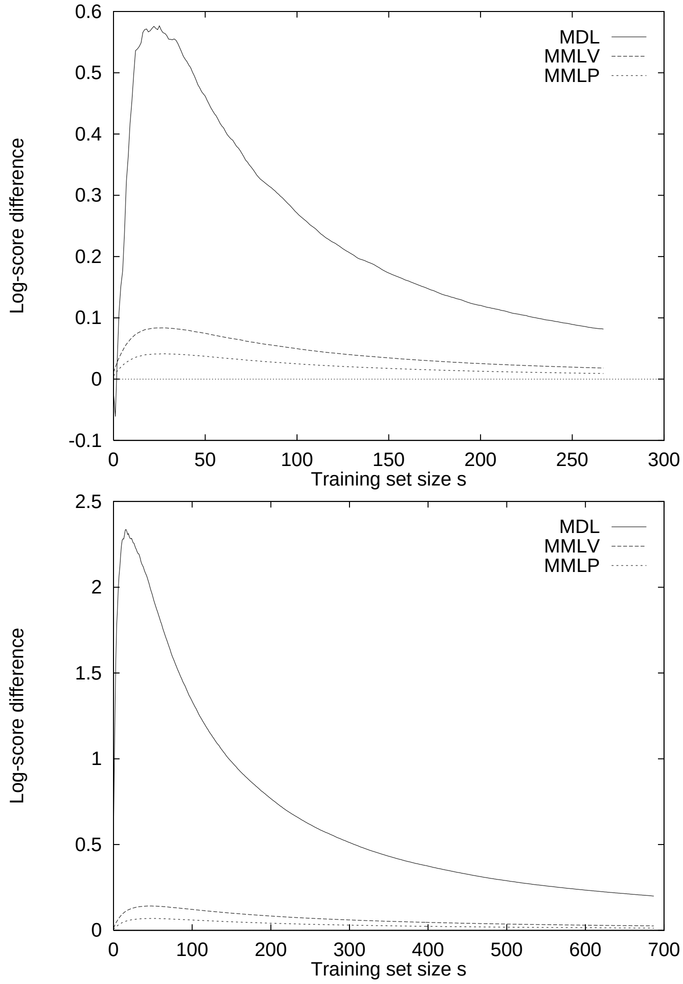


Figure 7.2: The upper picture shows the HD dataset leave-one-out crossvalidated results as a function of the training data available, scaled with respect to the MMLWF score. The lower picture shows the same function for the AU dataset. The higher the score, the better the prediction accuracy.

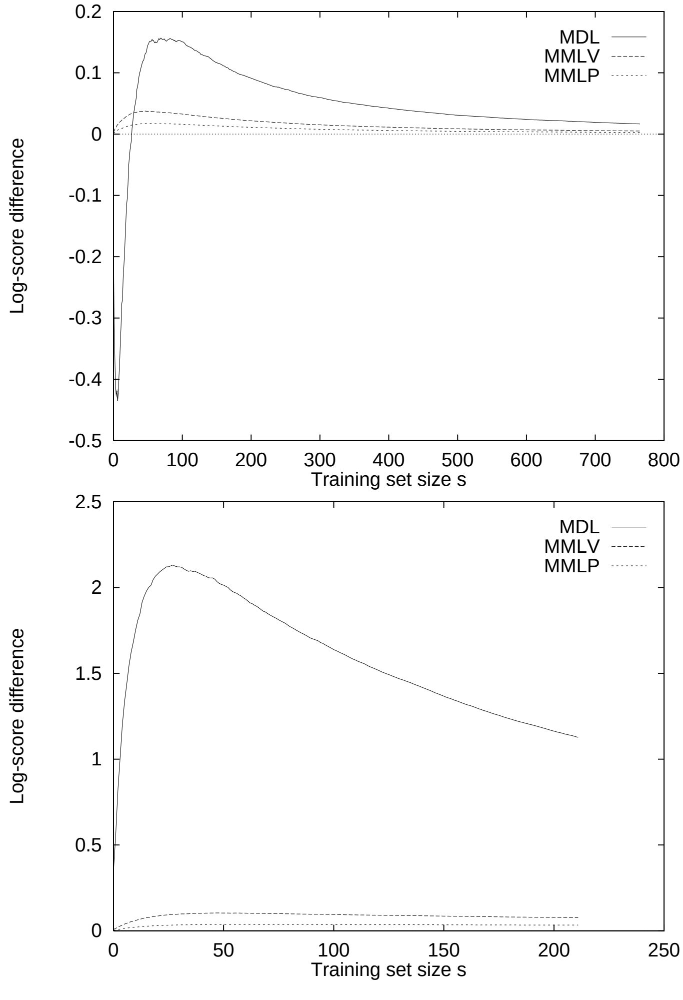
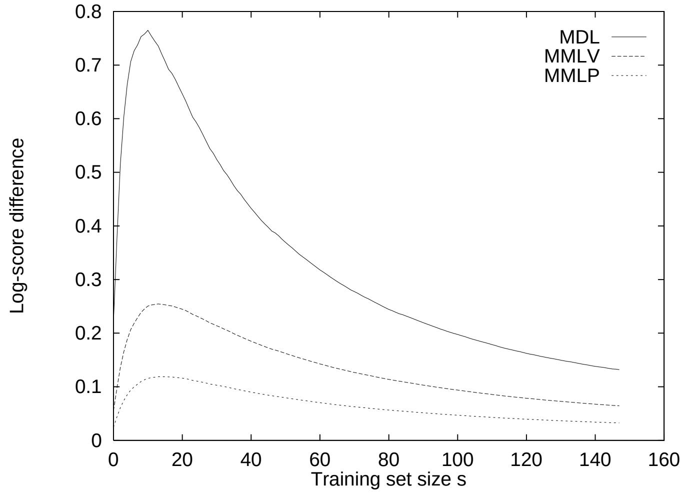


Figure 7.3: The upper picture shows the DB dataset leave-one-out crossvalidated results as a function of the training data available, scaled with respect to the MMLWF score. The lower picture shows the same function for the GL dataset. The higher the score, the better the prediction accuracy.



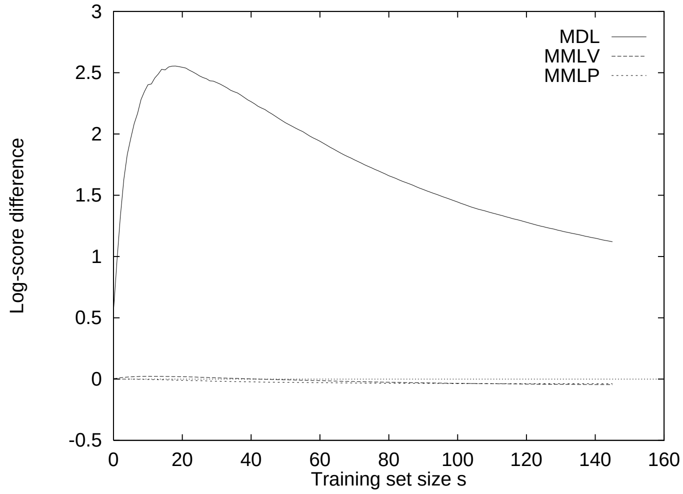


Figure 7.4: The upper picture shows the IR dataset leave-one-out crossvalidated results as a function of the training data available, scaled with respect to the MMLWF score. The lower picture shows the same function for the LY dataset. The higher the score, the better the prediction accuracy.