Chapter 11

Classification

by G. Corani, J. Abellán, A. Masegosa, S. Moral, M. Zaffalon

11.1 Introduction

Classification is the problem of predicting the *class* of a given instance, on the basis of some attributes (*features*) of it. In the Bayesian framework [1], a classifier is learned from data by updating a *prior* density, which represents the beliefs before analyzing the data and which is usually assumed uniform, with the *likelihood*, which models the evidence coming from the data; this yields a *posterior* joint probability over classes and features. Once the classifier has been learned from data, it can classify novel instances; under 0-1 loss, it returns the most probable class after conditioning on the value of the features of the instance to be classified.

Yet, Bayesian classifiers can happen to issue *prior-dependent* classifications, namely the most probable class changes under different priors. This might be acceptable if the prior has been carefully elicited and represents domain knowledge; however, in general the uniform prior is taken as default, without further investigation.

This consideration have lead to the development of *credal classifiers*, which extend Bayesian classifiers to imprecise probabilities. The term "credal classifier" was firstly used in [685], when introducing the naive credal classifier (NCC). NCC generalizes naive Bayes, representing a condition of prior ignorance[2] through the Imprecise Dirichlet Model (IDM) [643], namely adopting a set of priors instead of a single prior[3]. The set of priors is then updated by element-wise application of Bayes' rule, yielding a set of posteriors.

From this starting idea, imprecise-probability classification has been developed following two main directions: (a) extensions to imprecise probabilities of models based on Bayesian networks classifiers and (b) extension to imprecise probabilities of models based on classification trees. The common trait of such approaches is that they return a set of classes rather than a single class when there is not enough evidence for safely

[1] Cp. Section 8.1.4

[2] Actually, prior *near-ignorance*: full ignorance is not compatible with learning [642, Section 7.3.7].

[3] Cp. Section 8.4.3

classifying an instance by a single class. In this way, credal classifiers produce reliable classifications also on small data sets. Traditional classifiers generally undergo a sharp decrease of accuracy on the instances indeterminately classified by credal classifiers.

In Sections 11.2 we present the traditional naive Bayes; in Section 11.3 we show how it has been extended to imprecise probability, yielding the naive credal classifier (NCC); in Section 11.4 we review how NCC has been evolved yielding more sophisticated classifiers and in Section 11.5 we present the credal classifiers based on classification trees. In Section 11.6, we discuss the metrics suitable for scoring credal classifiers and present some experiments.

11.2 Naive Bayes

The naive Bayes classifier (NBC) [228] "naively" assumes the features to be independent given the class; this introduces a severe bias in the estimate of probabilities, as the real data generation mechanism does *not* generally satisfy such condition. As a consequence of such an assumption, NBC tends to assign an unrealistic high probability to the most probable class, being too confident in its predictions [331]; this phenomenon is emphasized when there are highly redundant features or when there is a large number of features.

Yet, NBC performs well under the 0-1 loss [228, 331]. A first reason is that the bias of the probability estimates may not matter under 0-1 loss: given two classes c_1 and c_2 (c_1 being the correct one), even severe bias in the estimate of $p(c_1)$ will not matter, provided that $p(c_1) > p(c_2)$ [289]. The success of NBC can be further explained by looking at the bias-variance decomposition of the misclassification error: NBC has in fact high bias but also low variance, which is the key factor on small data sets. In this way, NBC can be competitive in small and medium data sets, while it is generally outperformed by more complex classifiers on larger data sets.

Moreover, under careful feature selection NBC can become competitive [253, 282] even against state-of-the-art algorithms. A further strength of NBC is computational speed.

There has been a huge research aimed at improving the performance of naive Bayes; comprehensive references can be found in [331], [228] and [347]. The quantity of such attempts demonstrates the interest raised by NBC, which is recognized as one of the top 10 data mining algorithms [680]. The extension of NBC to imprecise probability is called *naive credal classifier* (NCC) [685].

11.2.1 Derivation of naive Bayes

Let us denote by C the classification variable (taking values in $\mathcal{C}$) and as $A_1, \ldots, A_k$ the k *discrete* feature variables (taking values from the sets $\mathcal{A}_1, \ldots, \mathcal{A}_k$). Values of class and features are denoted by lower-case letters, such as c and $\mathbf{a} = (a_1, \ldots, a_k)$. For a variable X (be it the class variable or a feature variable), $P(X)$ denotes a probability mass function over all the states $x \in \mathcal{X}$, while $P(x)$ denotes the probability of $X = x$. Both $P(X)$ and $P(x)$ are *epistemic* probabilities; they represent subjective probabilities,

obtained by updating prior beliefs with the evidence of the observations.

Instead, the *physical* probability of the actual data generation mechanism is called *chance*; it is unknown and we aim at making inference about it. We denote by $\theta_{c,\mathbf{a}}$ the chance that $(C, A_1, \ldots, A_k) = (c, \mathbf{a})$, by $\theta_{a_i|c}$ the chance that $A_i = a_i$ conditional on $C = c$, by $\theta_{\mathbf{a}|c}$ the chance that $A_1, \ldots, A_k = (a_1, \ldots, a_k)$ conditional on $C = c$.

The naive assumption of independence of the features given the class can be expressed as:

$$\theta_{\mathbf{a}|c} = \prod_{i=1}^{k} \theta_{a_i|c}. \tag{11.1}$$

We assume the data to be complete; the treatment of missing data will be discussed later in Section 11.3.1 and thus we can proceed by a standard conjugate Bayesian analysis, see also Section 8.4.1. We denote by n the total number of instances; by $n(c)$ the counts of class c and by $n(a_i, c)$ the counts of instances where $A_i = a_i$ and $C = c$; by $\mathbf{n}$ the vector of all counts of type $n(c)$ and $n(a_i, c)$. The *multinomial* likelihood can be expressed as a product of powers of the theta-parameters:

$$L(\theta|\mathbf{n}) \propto \prod_{c \in \mathcal{C}} \left[\theta_c^{n(c)} \prod_{i=1}^{k} \prod_{a_i \in \mathcal{A}_i} \theta_{a_i|c}^{n(a_i,c)} \right]. \tag{11.2}$$

The prior conjugate to the multinomial likelihood is constituted by a product of Dirichlet distributions; in fact, it is analogous to the likelihood, with the counts $n(\cdot)$ replaced everywhere by $st(\cdot) - 1$. The parameter $s > 0$ is the *equivalent sample size* and can be thought of as a number of *hidden instances*, thus interpreting conjugate Bayesian priors as additional samples. The parameters $t(\cdot)$ can be interpreted as the proportion of hidden instances of a given type; for instance, $t(c_1)$ is the expected proportion of hidden instances for which $C = c_1$.

A so-called *non-informative*

$$t(c) = \frac{1}{|\mathcal{C}|};\ t(a_i, c) = \frac{1}{|\mathcal{C}||\mathcal{A}_i|}, \tag{11.3}$$

which is also known as the BDe prior in the literature of Bayesian networks [387, chapter 18]. We denote by $\mathbf{t}$ the vector collecting all the parameters of type $t(c)$ and $t(a_i, c)$ which characterize the joint prior.

Alternatively, the Laplace prior is commonly adopted; it initializes to 1 all counts $n(c)$ and $n(a_i, c)$ before analyzing the data; such prior is also referred to as K2 [387, chapter 18] in the literature of Bayesian networks. The Laplace prior is obtained by setting $s = |\mathcal{C}|$ and $t(c) = 1/|\mathcal{C}|$ for the marginal probability of the class and $s = |\mathcal{A}_i|$ and $t(c) = 1/|\mathcal{A}_i|$ for the conditional probability of feature $\mathcal{A}_i$ given the class. As can be seen, the Laplace prior does not generally correspond to a joint prior over features and class; moreover, the equivalent sample size cannot be interpreted as the number of hidden instances, since s can be different for each variable.

By multiplying the joint Dirichlet prior and the multinomial likelihood, we obtain a posterior density for $\theta_{c,\mathbf{a}}$, which is again a product of Dirichlet densities; compared to

the likelihood (11.2), the parameters $n(\cdot)$ are replaced by $n(\cdot) + st(\cdot)$:

$$P\left(\theta_{c,\mathbf{a}}|\mathbf{n},\mathbf{t},s\right) \propto \prod_{c\in\mathcal{C}}\left[\theta_c^{n(c)+st(c)-1}\prod_{i=1}^{k}\prod_{a_i\in\mathcal{A}_i}\theta_{a_i|c}^{n(a_i,c)+st(a_i,c)-1}\right]. \tag{11.4}$$

Once $P\left(\theta_{c,\mathbf{a}}|\mathbf{n},\mathbf{t}\right)$ has been estimated, the learning step has been accomplished.

At this point, NBC can classify instances, once the assignment $\mathbf{a} = (a_1,\ldots,a_k)$ of the features is known. Considering that $P(c|\mathbf{a},\mathbf{n},\mathbf{t}) \propto P(c,\mathbf{a}|\mathbf{n},\mathbf{t})$, we focus on the latter expression, obtaining:

$$P(c,\mathbf{a}|\mathbf{n},\mathbf{t}) = P(c|\mathbf{n},\mathbf{t})\prod_{i=1}^{k}P(a_i|c,\mathbf{n},\mathbf{t}), \tag{11.5}$$

and by taking expecations we get:

$$P(c|\mathbf{n},\mathbf{t}) = \frac{n(c)+st(c)}{n+s} \tag{11.6}$$

$$P(a_i|c,\mathbf{n},\mathbf{t}) = \frac{n(a_i,c)+st(a_i,c)}{n(c)+st(c)}. \tag{11.7}$$

A problem of NBC, and of Bayesian classifiers in general, is that sometimes the classification is *prior-dependent*, namely the most probable class varies under different $\mathbf{t}$. This can be acceptable if the prior is carefully elicited and thus models domain knowledge; yet, this situation is not common and thus in general prior-dependent classifications are unreliable. It can be moreover argued ([642], Sec. 5.5.1) that the uniform prior is a model of prior *indifference*, which does not properly represents a condition of prior ignorance. To address such concerns NCC extends NBC to imprecise probability, substituting the non-informative prior by the Imprecise Dirichlet Model [643].

11.3 Naive Credal Classifier (NCC)

NCC adopts a *joint* credal set, which is modeled by the IDM to represent prior *near-ignorance* [642, Section 4.6.9]. As a consequence, $\mathbf{t}$ ranges within polytope $\mathcal{T}$, rather than being fixed. The polytope contains all the densities for which $\mathbf{t}$ satisfy the constraints, $\forall(i,c)$: $\sum_c t(c) = 1$, $\sum_{a_i} t(a_i\,|c) = t(c)$, $0 < t(a_i\,|c) < t(c)$, $0 < t(c) < 1$.

Checking credal-dominance

To classify an instance under 0-1 loss, traditional classifiers return the most probable class from $P(C|\mathbf{a},\mathbf{n},\mathbf{t})$. Instead, NCC identifies the *non-dominated* classes using the criterion of *maximality*[4]: c_1 dominates c_2 if $P(c_1,\mathbf{a}|\mathbf{n},\mathbf{t}) > P(c_2,\mathbf{a}|\mathbf{n},\mathbf{t})\ \forall\ \mathbf{t}\in\mathcal{T}$. Maximality does not always determine a total order on the possible classes; as a consequence, the decision rule can sometimes select a set of classes rather than a single class.

[4]See also definition 9.7

The test of dominance for NCC under maximality has been designed in [685]. In particular, c_1 dominates c_2 iff:

$$\inf_{\mathbf{t}\in\mathcal{T}} \frac{P(c_1, \mathbf{a}|\mathbf{n}, \mathbf{t})}{P(c_2, \mathbf{a}|\mathbf{n}, \mathbf{t})} > 1 \tag{11.8}$$

subject to:

$$\begin{aligned} 0 \leq t(c) \leq 1 \quad & \forall c \\ \textstyle\sum_c t_c = 1 \quad & \\ 0 < t(a_i, c) < t(c) \quad & \forall (a_i, c) \\ \textstyle\sum_{a_i} t(a_i, c) = t(c) \quad & \forall (c). \end{aligned} \tag{11.9}$$

Considering Eq. (11.6–11.7), the function (11.8) to be minimized becomes:

$$\inf_{\mathbf{t}\in\mathcal{T}} \left\{ \left[\frac{n(c_2) + st(c_2)}{n(c_1) + st(c_1)}\right]^{k-1} \prod_i \frac{n(a_i, c_1) + st(a_i, c_1)}{n(a_i, c_2) + st(a_i, c_2)} \right\} \tag{11.10}$$

subject to the same constraints.

The infimum of problem (11.10) is obtained by setting, $\forall a_i$, $t(a_i|c_1) \to 0$ and $t(a_i|c_2) \to t(c_2)$; moreover, the infimum is achieved when $t(c_1) + t(c_2) = 1$, which allows to express $t(c_2)$ as $1 - t(c_1)$. The obtained function depends only on $t(c_1)$ and, being convex in $t(c_1)$, can be exactly minimized. More details are given in [685].

NCC identifies the *non-dominated* classes by running many pairwise comparisons between classes, as shown in Fig. 11.1. The classification is *determinate* or *indeterminate* if there are respectively one or more non-dominated classes. The set of non-dominated classes contains the most probable class identified by NBC,[5], induced with any joint Dirichlet prior[6]; thus NCC, when determinate, returns the same class of NBC. If there are more non-dominated classes, the most probable class is *prior-dependent*, namely changes under different $\mathbf{t}$. The non-dominated classes are *incomparable*, namely there is no information to rank them. In fact, NCC drops the dominated classes as sub-optimal and expresses indecision about the optimal class by yielding the remaining set of non-dominated classes.

The indeterminacy (% of instances indeterminately classified) of NCC decreases with the size of the learning set because the prior becomes less influential as more data are available. For similarly sized data sets, the determinacy of NCC generally decreases with increasing number of classes or values of the feature, which fragment more the data set.

NBC is generally little accurate on the instances indeterminately classified by NCC; in [145], it is reported an average drop of some 30 points of accuracy for NBC, between the instances determinately and indeterminately classified by NCC. The overall performance of

[5]Assuming the s used to learn NBC to be no larger than the s used to learn NCC.

[6]This is not guaranteed for instance under the Laplace prior, although even in this case the statement is most often satisfied.

IDENTIFICATION OF NON-DOMINATED CLASSES
Input: the feature values $\mathbf{a}$
Output: the non-dominated classes.

1. set NonDominatedClasses := $\mathcal{C}$;
2. **for** class $c_1 \in \mathcal{C}$
   - **for** class $c_2 \in \mathcal{C}$, $c_2 \neq c_1$
     - **if** $\min_{\mathbf{t} \in \mathcal{T}} \frac{p(c_1|\mathbf{a},\mathbf{n},\mathbf{t})}{p(c_2|\mathbf{a},\mathbf{n},\mathbf{t})} > 1$, drop c_2 from NonDominatedClasses;
3. **return** NonDominatedClasses.

Figure 11.1: Identification of non-dominated classes via pairwise comparisons.

NBC is therefore constituted by good accuracy on the *non*-prior-dependent instances and a much lower accuracy on the prior-dependent ones. Instead, indeterminate classification preserves the reliability of NCC on prior-dependent instances; they can moreover be informative, when for instance only few classes are returned, out of many possible ones.

Applications of NCC to real-world case studies include diagnosis of dementia [693] and prediction of presence of parasites in crops [687].

Particular behaviors of NCC

Consider again problem (11.10), namely the minimization of $\frac{P(c_1,\mathbf{a}|\mathbf{n},\mathbf{t})}{P(c_2,\mathbf{a}|\mathbf{n},\mathbf{t})}$. The infimum is obtained by setting $t(a_i, c_1) \to 0 \; \forall i$; in the particular case of $n(a_i, c_1) = 0$, we have $n(a_i, c_1) + st(a_i, c_1) = 0$. In this case, $P(a_i|c_1, \mathbf{n}, \mathbf{t})$ goes to sharp 0 when solving the minimization problem (11.10), causing $P(c_1, \mathbf{a}|\mathbf{n}, \mathbf{t})$ to be 0 as well and therefore preventing c_1 from dominating any class. Thus, the presence of a single count $n(a_i, c_1) = 0$ prevents c_1 from dominating any class, regardless the information coming from the remaining features. This phenomenon is called the *feature problem* in [141]; experiments show [141] that NBC can be quite accurate on the instances which are indeterminately classified because of the feature problem. This can be explained by considering that NCC is in fact ignoring the information from all the remaining features.

A further counter-intuitive behavior of NCC is called the *class problem* in [141] and was already observed also in [685, Sec. 6]; the point is that it is very difficult to dominate a class which is never observed (or very rarely observed) in the data. Let us consider what happens when such class is at the denominator of Eq.(11.8), namely it corresponds to c_2. For any value a_i of the feature, there are no data for estimating $P(a_i|c_2, \mathbf{n}, \mathbf{t})$, which therefore only depends on the prior; under the IDM, $P(a_i|c_2, \mathbf{n}, \mathbf{t})$ varies between 0 and 1 and it is set to 1 to minimize the ratio of probabilities. As this behavior repeats for each feature, $P(\mathbf{a}|c_2, \mathbf{n}, \mathbf{t}) \gg P(\mathbf{a}|c_1, \mathbf{n}, \mathbf{t})$. Recalling that $P(c|\mathbf{a}, \mathbf{n}, \mathbf{t}) \propto P(c|\mathbf{n}, \mathbf{t})P(\mathbf{a}|c, \mathbf{n}, \mathbf{t})$, one can see that the class problem often prevents c_2 from being dominated; thus, c_2 is

often returned as non-dominated. When indeterminacy is due to the class problem, the accuracy of NBC generally does not drop on the instances indeterminately classified.

Both the feature and the class problem can be addressed by restricting the set of priors of the IDM [141], for instance by considering an ϵ-contamination of the credal set of the IDM with the uniform prior. Yet, restricting the credal set expresses preferences among possible values of the parameters; after the restriction, the prior credal set becomes at some extent informative and does not satisfy some properties of the original IDM, such as the representation invariance principle.

11.3.1 NCC2: Conservative treatment of missing data

Very often, real data sets are subject to missingness: this is the case when some values of the variables are not present in the data set.[7] We call these data sets *incomplete*[8].

Dealing with incomplete data sets rests on the assumptions done about the process responsible for the missingness. This process can be regarded as one that takes in input a set of complete data, which is generally not accessible for learning, and that outputs an incomplete data set, obtained by turning some values into missing. Learning about the missingness process' behavior is usually not possible by only using the incomplete data. This fundamental limitation explains why the assumptions about the missingness process play such an important role in affecting classifiers' predictions. Moreover, a missingness process may also be such that the empirical analysis of classifiers is doomed to provide misleading evidence about their actual predictive performance, and hence, indirectly, about the quality of the assumptions done about the missingness process. This point in particular has been discussed in [145, Section 4.6] and [691, Section 5.3.2]. For these reasons, assumptions about the missingness process should be stated with some care.

In the vast majority of cases, common classifiers deal with missing values (sometimes implicitly) assuming that the values are *missing at random* [517] (MAR). This assumption makes it possible to deal with incomplete data through the tools used for complete data. In the case of NBC, MAR justifies, for instance, dropping the missing values from the learning set, as well as those in the instance to classify. Unfortunately, MAR entails the idea that the missingness is generated in a non-selective way (for example in an unintentional way), and this is largely recognized to substantially lead to a narrow scope of MAR in applications. In classification, this may mean that assuming MAR decreases the predictive performance, sometimes in a way that is not easy (or possible) to assess empirically.

As a result, the imprecision introduced by missing data leads to an increase in the indeterminacy of the NCC, which is related to the amount of missingness. In other words, the NCC copes with the weak knowledge about the missingness process by weakening the answers it gives, thus maintaining reliability.

Later work [145] has extended the way NCC deals with missing values in two directions: it has enabled NCC to deal conservatively with missing values also in the instance to

[7] This may happen also to the class variable, but we do not consider such a case in the present chapter.
[8] See also Section 8.8 on data imprecision.

classify; and it has given the option to declare variables of two types: those subject to a MAR process and those to a process whose behavior is basically unknown. The first group is treated according to MAR; the second in a conservative way. The resulting classifier is called NCC2.

Distinguishing variables that are subject to the two types of processes is important because treating MAR variables in a conservative way leads to an excess of indeterminacy in the output that is not justified. In fact, the experimental results of NCC2 show that the indeterminacy originated from missing data is compatible with informative conclusions provided that the variables treated in a conservative way are kept to a reasonable number (feature selection can help in this respect, too). Moreover, they show that the classifiers that assume MAR for all the variables are often substantially unreliable when NCC2 is indeterminate.

Formal justifications of the rule NCC2 uses to deal with missing values can be found in [145]. This work discusses also, more generally, the problem of incompleteness for uncertain reasoning. An open source implementation of NCC2 is available [144].

11.4 Extensions and developments of NCC

11.4.1 Lazy NCC

Two aspects of NCC which can be improved are (a) the high bias due to the naive assumption and (b) a sometimes high indeterminacy, not only because of the feature or the class problem. The lazy NCC (LNCC) [146] addresses both problems by combining NCC and *lazy learning*.

The lazy learning algorithms defer the training, until it has to classify an instance (*query*). In order to classify an instance, a lazy algorithm:

1. ranks the instances of the training set according to the distance from the query;
2. trains a local classifier on the k instances nearest to the query and returns the classification using the local classifier;
3. discards the locally trained classifier and keeps the training set in memory in order to answer new queries.

Lazy classifiers are *local*, as they get trained on the subset of instances which are nearest to the query. The parameter k (*bandwidth*) controls the bias-variance trade-off for lazy learning. In particular, a smaller bandwidth implies a smaller bias (even a simple model can fit a complex function on a small subset of data) at a cost of a larger variance (as there are less data for estimating the parameters). Therefore, learning locally NBC can be a winning strategy as it allows reducing the bias; moreover, working locally reduces the chance of encountering strong dependencies between features [287]. The good performance of local NBC is studied for instance in [287].

An important problem dealing with lazy learning is how to select the bandwidth k. The simplest approach is to empirically choose k (for instance, by cross-validation on the

training set) and to then use the same k to answer all queries. However, the performance of lazy learning can significantly improve if the bandwidth is adapted query-by-query, as shown in [77] in the case of regression.

LNCC tunes the bandwidth query-by-query using a criterion based on imprecise probability. After having ranked the instances according to their distance from the query, a local NCC is induced on the k_{min} closest instances (for instance, $k_{min} = 25$) and classifies the instance. The classification is accepted if determinate; otherwise, the local NCC is updated by adding a set of further k_{upd} instances (we set $k_{upd} = 20$) to its training set. The procedure continues until either the classification is determinate or all instances have been added to the training of the local NCC. Therefore, the bandwidth is increased until the locally collected data smooth the effect of the choice of the prior. The naive architecture makes it especially easy updating LNCC with the k_{upd} instances; it only requires to update the vector $\mathbf{n}$ of counts, which is internally stored by LNCC.

By design LNCC is generally more determinate than NCC; in the worst case, LNCC is as determinate as NCC. Thus, LNCC addresses both the bias and the indeterminacy problem mentioned at the beginning of this section.

Extensive experiments [146] show that LNCC generally is more accurate or as accurate as NCC, but generates a less indeterminate output; it can be therefore regarded as delivering higher performance than NCC. This is especially apparent in data sets containing thousands of instances, where LNCC generally uses a bandwidth of a few hundreds; this allows a considerable bias reduction and results in generally better classifications.

11.4.2 Credal model averaging

Model uncertainty is the problem of having multiple models which provide a good explanation of the data, but lead to different answers when used to make inference. In this case, selecting a single model while discarding the competing ones underestimates uncertainty. *Bayesian model averaging* (BMA) [348] addresses model uncertainty by averaging over the set of candidate models, assigning to each model a weight corresponding to its posterior probability.

In case of NBC, given k features, there are 2^k possible NBCs, each characterized by a different subset of features; we denote by $\mathcal{G}$ the set of such models and by g a generic model of the set (g can be seen as standing for *graph.*). Using BMA, the posterior probability $P(c, \mathbf{a}|\mathbf{n}, \mathbf{t})$ is computed by averaging over *all* the 2^k different NBCs, namely by marginalizing g out:

$$P(c, \mathbf{a}|\mathbf{n}, \mathbf{t}) \propto \sum_{g \in \mathcal{G}} P(c, \mathbf{a}|\mathbf{n}, \mathbf{t}, g) P(\mathbf{n}|g) P(g), \tag{11.11}$$

where $P(g)$ is the prior probability of model g and $P(\mathbf{n}|g) = \int P(\mathbf{n}|g, \boldsymbol{\theta}) P(\boldsymbol{\theta}|g) d\boldsymbol{\theta}$ is its marginal likelihood. The posterior probability of model g is $P(g|\mathbf{n}) \propto P(\mathbf{n}|g)P(g)$. For a Bayesian network learned with the BDe prior, the marginal likelihood corresponds to the BDeu score [337].

BMA implies two main challenges [109]: the computation of the exhaustive sum of Eq. (11.11) and the choice of the prior distribution over the models. The computation of

BMA is difficult, because the sum of Eq. (11.11) is often intractable; in fact, BMA is often computed via algorithms which are both approximate and time-consuming. However, Dash and Cooper [167] provide an exact and efficient algorithm to compute BMA over all the 2^k NBCs.

As for the choice of the prior, a common choice is to assign equal probability to all models; in the following it is understood that BMA is induced with the uniform prior over the models. However, the uniform prior has the drawbacks already discussed in Sec. 11.2.1; its adoption is questioned also within the literature of BMA (see the rejoinder of [348]). The idea of *credal model averaging* (CMA) is thus to substitute such uniform prior by a credal set.

Within BMA, the prior probability of model g is generally expressed as:

$$P(g) = \prod_{A_i \in g} p_i \prod_{A_i \notin g} (1 - p_i), \tag{11.12}$$

where p_i is the prior probability of feature A_i to be present in the true model; moreover, $A_i \in g$ and $A_i \notin g$ index the features which, within the naive model g, are respectively linked to the class or isolated. The uniform prior is obtained by setting $p_i := 0.5$ for all i.

CMA models a condition close to ignorance about the prior probability of the 2^k NBC_s, by letting vary each p_i within the interval $\epsilon < p_i < 1 - \epsilon$, with $\epsilon > 0$ and $\epsilon \ll 1$. This defines a credal set $K(G)$ of mass functions $P(G)$. The introduction of ϵ is necessary to allow learning from the data; otherwise, the posterior probability of the models will keep ranging between 0 and 1, even after having observed the data.

The test of maximality for CMA is:

$$\inf_{P(G) \in K(G)} \frac{\sum_{g \in \mathcal{G}} P(c_1, \mathbf{a} | \mathbf{n}, \mathbf{t}, g) P(\mathbf{n} | g) P(g)}{\sum_{g \in \mathcal{G}} P(c_2, \mathbf{a} | \mathbf{n}, \mathbf{t}, g) P(\mathbf{n} | g) P(g)} > 1. \tag{11.13}$$

The dominance test [143] is computed by extending to imprecise probability the BMA algorithm by [167]. The set of non-dominated classes identified by CMA includes the most probable class identified by BMA; thus CMA, when determinate, returns the same class of BMA.

The experiments of [143] show that the accuracy of BMA sharply drops on the instances where CMA gets indeterminate. The finding that a Bayesian classifier is little accurate on the instances indeterminately classified by its counterpart based on imprecise probabilities is indeed consistent across the various credal classifiers developed so far. Future research could involve CMA for NCC, namely imprecise averaging of credal classifiers.

11.4.3 Profile-likelihood classifiers

A way for extending graphical models to imprecise probability, alternative to coping with multiple priors via the IDM, is adopting a profile likelihood approach.[9] The profile

[9] See also the sketch of further inference approaches at page 209.

likelihood [481] approach extends the paradigm of maximum likelihood by retaining a set of parameter estimates: all the estimates yielding a likelihood above a certain threshold.

Consider a *credal set* $\mathbf{P}$, i.e., a collection of probability distributions P_θ over variable θ, which takes values in a set Θ. Given the available data $\mathcal{D}$, the *normalized likelihood* [481] is:

$$lik(\theta) := \frac{P_\theta(\mathcal{D})}{\sup_{\theta' \in \Theta} P_{\theta'}(\mathcal{D})}. \tag{11.14}$$

We then remove from $\mathbf{P}$ the distributions whose profile likelihood is below a threshold $\alpha \in [0, 1]$:

$$\mathbf{P}_\alpha := \{P_\theta\}_{\theta \in \Theta | lik(\theta) \geq \alpha}. \tag{11.15}$$

In [99, 19], it is discussed how to perform inference and classification using the profile likelihood with naive structures: the resulting model is called *naive hierarchical credal classifier* (HNCC) and corresponds to a collection of NBCs, each characterized by a different parameter estimate with profile likelihood $lik \geq \alpha$. The classifier applies instance by instance the criterion of *maximality* in order to determine the class (or the set of classes) to be returned. Further investigation could be necessary to setup a principled criterion for defining the value of α, which controls the degree of imprecision of HNCC; however, in [19], preliminary experiments are performed using α=0.75 or 0.95. For this choice of the parameter, the behavior of HNCC shows some similarity with that of NCC.

11.4.4 Tree-Augmented Networks (TAN)

A way to reduce the bias of NBC is to relax the independence assumption using TAN (tree-augmented naive Bayes) [290]. In particular, TAN is a Bayesian network where each feature node has one parent (the class) and possibly also a second one (a feature); yet, a feature node cannot have more than two parents. In fact, TAN has been shown to outperform both general Bayesian networks and NBC [290, 427].

The test of dominance for a credal TAN takes in principle a form similar to Eq. (11.8) for NCC; yet, the TAN structure generates a set of additional constraints w.r.t. to the NCC case, so that the minimization problem becomes much harder. To have a tractable optimization problem for TAN it is necessary to slightly modify the definition of the credal set w.r.t. to NCC; this is the subject of the next section.

Global IDM, Local IDM and Extreme IDM

Three kinds of IDM can be used with credal networks: the *global*, the *local* and the *extreme* (EDM) [92]. The differences between these approaches are explained in the following by considering the simple network

$$C \to A$$

.

Let us focus on the class node. The constraints which define the set of Dirichlet distributions for the IDM (both local and global) are:

$$\begin{cases} \sum_c t(c) = 1 \\ 0 < t(c) < 1 \quad \forall c \in \mathcal{C}. \end{cases} \tag{11.16}$$

The credal set $K(C)$ is a set of mass functions $P(C)$, within which $P(c)$ is estimated as:

$$P(c) \in \left[\frac{n(c)}{s+n}, \frac{s+n(c)}{s+n} \right]. \tag{11.17}$$

The EDM restricts the set of priors defined by Eq. (11.16) to the most extreme mass functions, i.e., each $t(c)$ can be only zero or one. Consequently, the probability of class c can only assume two values, namely the upper or the lower extreme of interval (11.17).

Let us now consider the feature node. The local IDM, similarly to Eq. (11.16), adopts the constraints:

$$\begin{cases} \sum_a t(a,c) = 1 \quad \forall c \in \mathcal{C} \\ t(a,c) > 0 \qquad \forall a \in \mathcal{A}, \forall c \in \mathcal{C}. \end{cases} \tag{11.18}$$

Yet, there is no relation between the $t(a,c)$ of Eq. (11.18) and the $t(c)$ of Eq. (11.16). For each c, the credal set $K(A|c)$ is a set of mass functions $P(A|c)$, where $P(a|c)$ is estimated as:

$$P(a|c) \in \left[\frac{n(a,c)}{s+n(c)}, \frac{s+n(a,c)}{s+n(c)} \right]. \tag{11.19}$$

The credal sets $\{K(A|c)\}_{c \in \mathcal{C}}$ and $K(C)$ are thus independent of each other; the credal network is *separately specified*.

Instead, the *global* IDM is based on a set of *joint* Dirichlet distributions, defined by the constraints (already given in Section 11.2):

$$\begin{cases} \sum_{c \in \Omega_C} t(c) = 1 \\ t(c) > 0 & \forall c \in \mathcal{C} \\ \sum_a t(a,c) = t(c) & \forall c \in \mathcal{C} \\ t(a,c) > 0 & \forall a \in \mathcal{A}, \forall c \in \mathcal{C}. \end{cases} \tag{11.20}$$

In particular, the third constraint links $t(a,c)$ and $t(c)$, unlike in the local IDM; therefore, the model is *not* separately specified. For a specific value of $t(c)$, the credal set $K(A|c)$ contains the mass functions $P(A|c)$ such that:

$$P(a|c) \in \left[\frac{n(c,a)}{st(c)+n(c)}, \frac{st(c)+n(c,a)}{st(c)+n(c)} \right]. \tag{11.21}$$

The global IDM estimates narrower intervals than the local, as can be seen by comparing Equation (11.21) and Equation (11.19):[10] this implies less indeterminate classifications.

[10] Recall that $\sum_c t(c) = 1$ and that $t(c) > 0 \ \forall c \in \mathcal{C}$.

Yet, the global IDM is more difficult to compute; so far, exact computation with the global IDM has been possible only with NCC. Instead, the local IDM can be computed for any network topology and is in fact often adopted with general credal networks; yet, it returns wider intervals, which are sometimes too little informative in real applications.

The EDM restricts the global IDM to its extreme distributions; it therefore allows $t(a,c)$ to be either 0 or $t(c)$, keeping the constraint $\forall c \in \Omega_C : \sum_{a\in\Omega_A} t(a,c) = t(c)$ inherited from the global IDM. The extreme points of the EDM corresponds in this case to the bounds of the interval in Equation (11.21); but in general, they are an inner approximation of the extremes of the global IDM [92]. The EDM can be interpreted as treating the s hidden instances as s rows of missing data; the rows are assumed *identical*; the ignorance is due to the fact that it is unknown which values the hidden instances contain. Therefore, the replacement considered by the EDM for the hidden instances are finite, while the global IDM consider infinite possible replacements, by letting the parameters $t(c)$ vary in a continous fashion, as in the constraints of Eq.(11.9).

The reliability of the EDM has been validated [142] by showing that NCC return almost identical classification when induced with the global IDM and the EDM.

Credal TAN

A credal TAN was firstly proposed in [690]; the classifier was reliable and very accurate when returning a single class, but raised a problem of excessive indeterminacy, due to the local IDM.

More recently [142] the credal TAN has been developed using the EDM. Experimental results [142] show that the novel credal TAN provides overall a better performance than the previous one, returning less indeterminate classifications without decreasing reliability. Yet, in some cases it is still much more indeterminate than NCC; this is due to the TAN structures (learned using Bayesian procedures), which sometimes assign the second parent to a feature node even though it induces a contingency table full of 0s; this generates large indeterminacy when used under imprecise probability. Future research could be about learning parsimonious TAN structures, more suited to usage with imprecise probability.

11.5 Tree-based credal classifiers

A decision tree [11] (also called a classification tree) is a simple structure, a tree, that can be used as a classifier. Within a decision tree, each node represents an attribute variable and each branch represents one of the states of this variable. A tree leaf specifies the expected value of the class variable conditioned to the path from the root to this leaf and depending on the information contained in the training data set. Quinlan's ID3 algorithm [499] for building decision trees is based on uncertainty-based information theory taking probability theory as the model to represent uncertainty. Building a tree from data implies the selection of the attributes for the nodes (branching attributes),

[11] The term 'decision tree' is used in different meaning in different contexts, namely as a way to describe sequential decision making (see Section 9.2) or as a synonym to classification tree.

stopping and pruning rules. Usually, these rules are based on measures of information and entropy.

In the last years, studies about properties and behavior of measures of uncertainty on more general theories than probability, have been presented, principally in the Dempster-Shafer theory of evidence (DST) (Dempster [214] and Shafer [548]), and in the theory of general credal sets (Walley [642]). In DST, Yager [681] extends the concept of uncertainty in probability theory, considering two main origins of uncertainty: conflict and non-specificity. In Abellán and Moral [4] and Abellán et al. [2], functions are presented to quantify these types of uncertainty on credal sets (Klir [382]).

It is possible to apply these measures of uncertainty on credal sets to build classification trees taking imprecise probability as the basic model. In Abellán and Moral [5], a method to build decision trees based on credal sets and measures of uncertainty is presented. This method starts with an empty tree and, for branching at each node, it selects the variable with the greatest degree of total uncertainty reduction in relation to the class variable. In the theory of probability, branching always implies a reduction in entropy. It is, therefore, necessary to include an additional criterion in order to not create excessively complex models. For this aim, a total uncertainty criterion it is used, i.e., a measure of total uncertainty which quantifies conflict and non-specificity. In credal sets, although the conflict part of the uncertainty produced by branching is smaller, the non-specificity part is greater. So, in this case a very simple stopping criterion can be proposed: when branching produces an increase in uncertainty (a decrease in conflict is not offset by an increase in non-specificity). Finally, it is used a dominance criterion to the set values of the variable to be classified in the corresponding leaf.

In the following sections, we will analyze the measure of information used and the procedure to build decision trees.

11.5.1 Uncertainty Measures on Credal Sets. The maximum entropy function

The study of uncertainty measures in the Dempster-Shafer theory of evidence [214, 548] has been the starting point for the development of these measures in more general theories (a study of the most important measures proposed in the literature can be seen in [382]). As a reference for the definition of an uncertainty measure on credal sets, Shannon's entropy [557] has been used due to its operation on probabilities. In any theory which is more general than probability theory, it is essential that a measure be able to quantify the uncertainty that a credal set represents: the parts of *conflict* and *non-specificity* [382].

In recent years, Klir and Smith [383] and Abellán and Moral [8] justified the use of the maximum of entropy on credal sets as a good measure of total uncertainty that verifies a set of basic properties [384]. The problem lies in separating this function into others which really do measure the parts of conflict and non-specificity, respectively. More recently, Abellán et al. [2] presented a separation of the maximum of entropy into functions which are capable of coherently measuring the conflict and non-specificity of a credal set $\mathcal{P}$ on a finite variable X, as well as algorithms for facilitating its calculation in

capacities of order 2 [7, 9] and this may be expressed in the following way:

$$S^*(\mathcal{P}) = S_*(\mathcal{P}) + (S^* - S_*)(\mathcal{P}),$$

where S^* represents the maximum of entropy and S_* represents the entropy minimum on the credal set $\mathcal{P}$:

$$S^*(\mathcal{P}) = \max_{P \in \mathcal{P}} \sum_x P(x) \log(P(x)),\ \ S_*(\mathcal{P}) = \min_{P \in \mathcal{P}} \sum_x P(x) \log(P(x)),$$

where $S_*(\mathcal{P})$ coherently quantifies the conflict part of the credal set $\mathcal{P}$ and $(S^* - S_*)(\mathcal{P})$ represents the non-specificity part of $\mathcal{P}$ [2].

In the particular case of belief functions, Harmanec and Klir [333] have already considered that upper entropy is a measure of total uncertainty. They justify it by using an axiomatic approach. However, uniqueness is not proved. But, perhaps the most compelling reason to use this function is given in Walley [642]. We start by explaining the case of a single probability distribution, P, on a finite set $\mathcal{X}$. It is based on the logarithmic scoring rule . To be subject to this rule means that we are forced to select a probability distribution Q on $\mathcal{X}$, and if the true value is x then we must pay $-\ln(Q(x))$. For example, if we say that $Q(x)$ is very small and x is found to be the true value, we must pay a lot. If $Q(x)$ is close to one, then we must pay a small amount. If our information about X is represented by a subjective probability P, then we should choose Q so that $P(-\ln(Q(X)))$ is minimum, where P is the mathematical expectation (prevision) with respect to P. This minimum is obtained when $Q = P$ and the value of $P(-\ln(P(x)))$ is the entropy of P: the expected loss or the minimum amount that we would require to be subject to the logarithmic scoring rule. This rule is widely used in statistics. The entropy is the negative of the expected logarithm of the likelihood under distribution P. The reason for taking logarithms is that if we do the prediction in two independent experiments at the same time, then the payment should be the addition of the payments in the two experiments.

For the case of a credal set, $\mathcal{P}$, we can also apply the logarithmic scoring rule, but now we choose Q in such a way that the upper expected loss $\overline{P}(-\ln(Q(x)))$ (the supremum of the expectations with respect to the probabilities in $\mathcal{P}$) is minimum. Under fixed Q, $\overline{P}(-\ln(Q(x)))$ is the maximum loss we can have (the minimum we should be given to accept this gamble). As we have freedom to choose Q, we should select it, so that this amount $\overline{P}(-\ln(Q(x)))$ is minimized.

Walley shows that this minimum is obtained for the distribution $\widehat{P} \in \mathcal{P}$ with maximum entropy.[12] Furthermore, $\overline{P}(-\ln(\widehat{P}(x)))$ is equal to $S^*(\mathcal{P})$, the upper entropy in $\mathcal{P}$. This is the minimum payment that we should require before being subject to the logarithmic scoring rule. This argument is completely analogous with the probabilistic one, except that we change expectation to upper expectation. This is really a measure of uncertainty, as the better we know the true value of X, then the less we should need to be paid to accept the logarithmic scoring rule (lower value of $S^*(\mathcal{P})$).

[12]The proof is based on the Minimax theorem which can be found in Appendix E of Walley's book [642].

The approach used here is different of what it is called *principle of maximum entropy* [365]. This principle always considers a unique probability distribution: the one with maximum entropy compatible with available restrictions. Here we are not saying that $\mathcal{P}$ can be replaced by the probability distribution of maximum entropy. We continue using the credal set to represent uncertainty. We only say that the uncertainty of the credal set can be measured by its upper entropy.

In order to obtain the maximum of entropy on a set of probability intervals in the application of the IDM from a sample, we can use the algorithm presented in Abellán and Moral [6] for probability intervals. When using values between 1 and 2 for the parameter s, we can use a simpler procedure of Abellán [1], which is a simplification of the one presented in Abellán and Moral [6].

Assume that we have a variable X taking values on a set $\mathcal{X}$. If we have a sequence of independent and identically distributed sequence of observations of this variable. Then, applying the IDM, the estimated probability intervals for any value $x_i \in \mathcal{X}$ (see equation 11.17) are:

$$\left[\frac{n(x_i)}{N+s}, \frac{n(x_i)+s}{N+s}\right], \qquad \forall x_i \in \mathcal{X},$$

where $n(x_i)$ is the frequency of observations in which $X = x_i$ in the sequence and N the total sample size.

To compute the maximum entropy of the credal set associated to these intervals, we must first determine the set $W = \{x_j | \, n(x_j) = \min_i\{n(x_i)\}\}$. Let $|W|$ be the cardinality of the set W. If we use $\widehat{P}$ to denote the distribution where the maximum of entropy will be reached, the procedure of Abellán [1] can be expressed in the following way:

Case 1 $|W| > 1$ or $s = 1$

$$\widehat{P}(x_i) = \begin{cases} \dfrac{n(x_i)}{N+s} & x_i \notin W \\ \dfrac{n(x_i)+s/|W|}{N+s} & x_i \in W. \end{cases}$$

Case 2 $|W| = 1$ and $s > 1$.

- Assign:

  $n(x_j) \longleftarrow n(x_j) + 1$ (where $W = \{x_j\}$),
  $s \longleftarrow s - 1$.

- Obtain new W.
- Obtain $\widehat{P}$ as in Case 1.

11.5.2 Obtaining Conditional Probability Intervals with the IDM

We adopt the same notation of Section 11.2; we recall in particular that the class variable is denoted as C (taking values in $\mathcal{C}$) while the *discrete* features are denoted by $\mathbf{A} = A_1, \ldots, A_k$ (the k feature variables taking values from the sets $\mathcal{A}_1, \ldots, \mathcal{A}_k$). Again, we assume the data to be complete. If $\mathbf{Y}$ is a subset of all the variables $\mathbf{A}$, then $\mathbf{y}$ will denote a generic value of it (a value for each one of the variables in the subset).

We work with the local IDM, described in Section 11.4.4; thus, the values of the attribute variables select a subset of data, which is used to estimate a credal set only for variable C. When classifying a new case, the values of attribute variables will be used to select the appropriate credal set for C.

Definition 11.1. A configuration, σ, about $\mathbf{A}$ is an assignment of values for a subset of variables: $\mathbf{Y} = \mathbf{y}$, where $\mathbf{Y} \subseteq \mathbf{A}$.

If $\mathcal{D}$ is a data set and σ is a configuration, then $\mathcal{D}[\sigma]$ will denote the subset of $\mathcal{D}$ given by the cases which are compatible with configuration σ (cases in which the variables in σ have the same values as the ones assigned in the configuration).

Definition 11.2. Given a data set and a configuration σ, we consider the credal set $\mathcal{P}^\sigma$ for variable C with respect to σ defined by the set of probability distributions, P, such that

$$P(c) \in \left[\frac{n(c)^\sigma}{N^\sigma + s}, \frac{n(n)^\sigma + s}{N^\sigma + s}\right], \qquad \forall c \in \mathcal{C},$$

where $n(c)^\sigma$ is the number of occurrences of $(C = c)$ in $\mathcal{D}[\sigma]$, N^σ is the number of cases in $\mathcal{D}[\sigma]$, and $s > 0$ is a parameter.

We denote this interval as

$$\left[\underline{P}^\sigma(c), \overline{P}^\sigma(c)\right].$$

This credal set is the one obtained on the basis of the local IDM [643] applied to the subsample $\mathcal{D}[\sigma]$. This set is also associated with a reachable set of probability intervals and with a belief function (Abellán [1])

Example 11.3. Assume that we have a class variable with 3 possible values: $\mathcal{C} = \{c_1, c_2, c_3\}$.

Suppose that we have a database and a configuration σ such that:

$$n(c_1)^\sigma = 4, \qquad n(c_2)^\sigma = 0, \qquad n(c_3)^\sigma = 0.$$

With $s = 1$, we have the following vector of probability intervals, using the IDM:

$$([\frac{4}{5}, 1]; [0, \frac{1}{5}]; [0, \frac{1}{5}]).$$

The credal set $\mathcal{P}^\sigma$ has three vertices:

$$\{(1, 0, 0); (\frac{4}{5}, \frac{1}{5}, 0); (\frac{4}{5}, 0, \frac{1}{5})\}.$$

This credal set is represented in Figure 11.2. Each point of the triangle represents a probability distribution in which the probability of c_i is the distance to the edge opposite to vertex c_i. The credal set is represented by the shadowed triangle. A general algorithm to find all the extreme points of a credal set that is defined by a system of intervals is given by Walley [642] and Campos, Huete, and Moral [187].

It is simple to compute the upper entropy of this credal set applying the above algorithm. We obtain: $S^*(\mathcal{P}^\sigma) = H(\frac{4}{5}, \frac{1}{10}, \frac{1}{10}) = 0.639$, where H is the classical Shannon entropy.

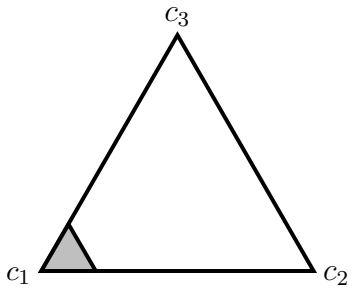

Figure 11.2: Simplex representation of the credal set in Example 11.3.

If we have a different database with the same relative frequencies for the values of C, but different sample size, then the credal set changes. If the sample size is smaller then the intervals are wider and if the sample size is larger then the intervals are more precise.

11.5.3 Classification Procedure

In a classification tree each interior node is labeled with a variable of the data set $A_i \in \mathbf{A}$. Each leaf will have a decision rule to assign a value of the class variable C. In traditional classification trees, the decision rule assigns a single value of C (see Figure 11.3).

In a classification tree there is a correspondence between nodes and configurations. Each node defines a configuration: the set of variables that can be found in the path to that node from the root, with the values associated with the children that lie in this path. A complete configuration (a value for each one of the variables in $\mathbf{A}$) defines a leaf: we start at the root, and at each inner node with label A_i, we select the child corresponding to the value of A_i in the configuration.

Given a data set $\mathcal{D}$, each node of the tree defines a credal set for C in the following way: we first consider the configuration σ associated to it, and then the credal set, $\mathcal{P}^\sigma$, as in Definition 11.2. For example, we have seen in Figure 11.3 that the node with label c_3 determines a configuration $\sigma = (A_1 = 1, A_3 = 0)$. This configuration has an associated data set, $\mathcal{D}[\sigma]$, which is the subset of the original $\mathcal{D}$ given by those cases for which $A_1 = 1$ and $A_3 = 0$. $\mathcal{P}^\sigma$ is the credal set built from this restricted data set, using the local IDM.

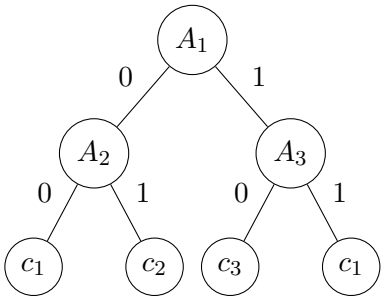

Figure 11.3: Example of a Classification Tree.

The method for building classification trees is based on measuring the total uncertainty of the credal set associated with each leaf. In the following we shall describe how to build the structure of the tree. The decision rules will be considered later.

The method starts with a tree with a single node. We shall describe it as a recursive algorithm, which is started with the empty node (the root node) with no label associated to it. Each node will have a list $\mathcal{L}^*$ of possible labels of variables which can be associated to it. The procedure will initially be started with the complete list of variables.

We will consider the following function implemented: $\texttt{Inf}(\sigma,A_i)$:

$$\texttt{Inf}(\sigma, A_i) = \left(\sum_{a_i \in \mathcal{A}_i} r(a_i)^\sigma TU(\mathcal{P}^{\sigma \cup (A_i = a_i)}) \right),$$

where $r(a_i)^\sigma$ is the relative frequency with which A_i ($r(a_i)^\sigma = n(a_i)^\sigma / N^\sigma$) takes value a_i in $\mathcal{D}[\sigma]$, $\sigma \cup (A_i = a_i)$ is the result of adding the value $A_i = a_i$ to configuration σ and TU is a total uncertainty measure. In the procedure, we will use the maximum entropy function as total uncertainty measure.[13]

If No is a node and σ a configuration associated with it, `Inf` tries to measure the weighted average total uncertainty of the credal sets associated with the children of this node if variable A_i is added to it (and there is a child for each one of the possible values of this node). The average is weighted by the relative frequency of each one of the children in the data set.

In the following we describe the method. The basic idea is very simple and it is applied recursively to each one of the nodes we obtain. For each one of these nodes, we consider whether the total uncertainty of the credal set at this node can be decreased by adding one of the nodes. If this is the case, then we add a node with a maximum decrease of uncertainty. If the uncertainty cannot be decreased, then this node is not expanded and it is transformed into a leaf of the resulting tree. This procedure is explained in the algorithm of Figure 11.4.

In this algorithm, A_l is the branching variable of node No. The intuitive idea is that when we assign this variable to No, we divide the database associated with this node

[13] As we see, the procedure is opened to any total uncertainty measure.

Procedure `BuilTree(`No`,`$\mathcal{L}^*$`)`

1. If $\mathcal{L}^* = \emptyset$, then `Exit`.
2. Let σ be the configuration associated with node No
3. Compute the credal set associated with σ
   and compute its total uncertainty $TU(\mathcal{P}^\sigma)$.
4. Compute the value
   $\alpha = \min_{A_i \in \mathcal{L}^*} \texttt{Inf}(\sigma, A_i)$
5. If α is greater than or equal to $TU(\mathcal{P}^\sigma)$ then
   6. `Exit`
7. If α is smaller than $TU(\mathcal{P}^\sigma)$, then
   8. Let A_l be the variable for which the minimum α is attained
9. Remove A_l from $\mathcal{L}^*$
10. Assign A_l to node No
11. For each possible value a_l of A_l
    12. Add a node No_l
    13. Make No_l a child of No
    14. Call `BuilTree(`No_l`,`$\mathcal{L}^*$`)`

Figure 11.4: Procedure to build credal decision trees.

among its different children. In each one of the children, we can have more precise average knowledge about C but based on a smaller sample. We consider that the total uncertainty of the associated credal sets can be a good measure of the appropriate trade-off between the precision gained by dividing the database according to the different values of A_l and the precision lost by estimating the probability distribution of C from a smaller database.

Traditional probabilistic classification trees, are built in a similar way, but with the difference that we have precise estimations of probability values and the uncertainty measure is Shannon entropy. The quantity that is used to decide what variable to use to add a branch to a node is called *information gain* and it is similar to $TU(\mathcal{P}^\sigma) - \texttt{Inf}(\sigma, A_i)$, which is what it is computed to decide the branching variable. The only difference is that information gain is applied to precise probabilities. If P^σ is a precise probability estimation of probabilities about C in $\mathcal{D}[\sigma]$ (maximum likelihood is the usual estimation method), then the information gain is given by

$$\texttt{InfGain}(\sigma, A_i) = H(P^\sigma) - \left(\sum_{a_i \in \mathcal{A}_i} r(a_i)^\sigma H(P^{\sigma \cup (A_i = a_i)}). \right)$$

The information gain is also called the *mutual information* between A_i and C in sample $\mathcal{D}[\sigma]$ and it is always a non-negative number. It is important to remark that the quantity we use, $TU(\mathcal{P}^\sigma) - \texttt{Inf}(\sigma, A_i)$, is not the supremum, nor the infimum of the mutual information, as in fact we are computing a difference of two upper values. If we were computing the supremum or the infimum of mutual information, we would always

obtain a non-negative value. On the other hand, $TU(\mathcal{P}^\sigma) - \texttt{Inf}(\sigma, X_i)$ can be negative and this is important as it is the criterion to stop branching. So, the procedure is not based on a sensitivity analysis of mutual information under imprecision. It can better understood as a method to choose between models: by comparing the information they give about the class variable.

Complexity

We are going to estimate the complexity as a function of the sample size, N, and the number of variables. As we never add a branch to a node which is compatible with no cases of the data, then the number of leaves is of order $O(N)$. The total number of nodes of the tree is of the same order. So we call procedure `BuilTree` a maximum of $O(N)$ times. Each time we call the procedure, we have to evaluate the weighted average of total uncertainty. To compute frequencies, we revise all the cases of the data set (N cases). In the method, for each call, we have to compute `Inf` for each variable. So, taking into account that m is the number of variables, we have a complexity for a call (without the recursive part) of $O(N \cdot m)$. Finally, the total complexity of building the complete tree in the method is $O(N^2 \cdot m)$.

We have considered that the number of possible values of a variable is constant. In fact, this is not an important factor, as the upper entropy for our interval probabilities can be found in linear time as a function of the number of possible classes by using algorithm in [6].

Decision at the Leaves

In order to classify a new example, with observations of all the variables except the class variable C, we obtain the leaf corresponding to the observed configuration, i.e. we start at the root of the tree and follow the path corresponding to the observed values of the variables in the interior nodes of the tree, i.e., if we are at a node with variable A_i which takes the value a_i, then we choose the child corresponding to this value. We then use the associated credal set for C, $\mathcal{P}^\sigma$, to classify the new example.

To do that, we first follow[14] *dominance* under *maximality*, as discussed in Section 11.3. We recall that class c_1 is dominated under maximality if and only if for every P in $\mathcal{P}^\sigma$, there is a class value c_2 such that $P(c_2) > P(c_1)$. In this particular application of the IDM, dominance under the strict preference ordering is equivalent to *interval dominance*: c_1 is dominated if and only if there is a class value c_2 such that $\overline{P}^\sigma(c_1) < \underline{P}^\sigma(c_2)$.

The decision rule is to assign to every leaf with credal set $\mathcal{P}^\sigma$, the set of non-dominated class values corresponding to $\mathcal{P}^\sigma$. In this way, we obtain a credal classifier, in which we obtain a set of predicted values for class variable, non-dominated cases, instead of a unique prediction.

[14] See also Section 9.1

Growing forests

The method of [5] splits on the attribute with the greatest total uncertainty reduction. In [161] it has been proposed a different approach in order to decide on which variable to split: this implies considering a credal set for each candidate attribute, computing the uncertainty reduction for each distribution in the credal set and thus obtaining, for each attribute, an interval of uncertainty reduction. The variable to be used for splitting is decided by comparing such intervals; but the comparison can well identify more non-dominated attributes for splitting. In this case, it is proposed to grow a forest, each tree of the forest being obtained by splitting on one of the non-dominated attributes. Experiments with this method are however not yet available.

11.6 Metrics, experiments and software

The overall performance of a credal classifier is fully characterized by four indicators [145]:

- *determinacy*, i.e., the percentage of instances determinately classified;
- *single-accuracy*, i.e., the accuracy on the *determinately* classified instances;
- *set-accuracy*, i.e., the accuracy on the *indeterminately* classified instances;
- *indeterminate output size*: the average number of classes returned on the indeterminately classified instances.

Note that set-accuracy and indeterminate output size are meaningful only if the data set has more than two classes.

These metrics completely characterize the performance of a credal classifier. As for comparing credal and traditional classifiers, this has been often done by assessing the drop of accuracy of the traditional classifier on the instances indeterminately classified by the credal; see for instance the empirical results in [145].

As for the problem of choosing among alternative credal classifiers, two metrics have been proposed in [146]. The first metric, borrowed from multi-label classification,[15] is the *discounted-accuracy*:

$$\text{d-acc} = \frac{1}{N}\sum_{i=1}^{N}\frac{(accurate)_i}{|Z_i|},$$

where $(accurate)_i$ is a 0-1 variable, showing whether the classifier is accurate or not on the i-th instance; $|Z_i|$ is the number of classes returned on the i-th instance and N is the number of instances of the test set. Although discounting *linearly* the accuracy on the output size might look arbitrary, a principled justification for it has been given in [688].

The non-parametric *rank test* overcomes this problem. On each instance, it ranks two classifiers CL_1 and CL_2 as follows:

[15] The metric is referred to as *precision* in [604].

- if CL_1 is accurate and CL_2 inaccurate: CL_1 wins;
- if both classifiers are accurate but CL_1 returns less classes: CL_1 wins;
- if both classifiers are wrong: tie.
- if both classifiers are accurate with the same output size: tie.

The wins, ties and losses are mapped into ranks and then analyzed via the Friedman test [216]. The rank test is more robust than d-acc, as it does not encode an arbitrary function for the discounting; yet, it uses less pieces of information and can therefore be less sensitive. Overall, a cross-check of both indicators is recommended.

However, both the discounted-accuracy and the rank test are subject to the drawback shown in the following example. Let us consider: a classification problem with two classes; a *random* classifier, which returns the class at random from a uniform distribution; a *vacuous* classifier, which returns both classes, being thus non-informative. Both discounted-accuracy and rank test would judge as equivalent the performance of the random and the vacuous classifier; in particular, discounted-accuracy would assign to both classifiers a score of 1/2.

In fact, both classifiers do not know how to predict the class, but only the vacuous classifier declares it. From this viewpoint, one might argue that the vacuous should be rewarded with more than 1/2: for instance, common sense suggests that a doctor who admits to be unable to diagnose is preferable to another one who issues random diagnoses. However, the reliability of a classifier is tightly related to the variability of its accuracy [688]; the aversion to this variability is what makes some people prefer credal classifier to precise ones. In the previous example, the vacuous classifier obtains 1/2 on each instance, while the score of the random continuously changes between 0 and 1. Therefore both classifiers have the same *expected* discounted-accuracy, but the discounted-accuracy of the vacuous is subject to much less variance. Let us consider the discounted-accuracy score as a reward. A risk-averse decision maker would prefer the vacuous classifier over the random one, since they have the same expected reward of the random, but the former is subject to less variance. In [688, 689], this point is explored representing the subjective preferences of the decision maker via a utility function applied to the discounted-accuracy scores. It is formally shown that, under any concave utility function, the vacuous classifier has higher expected utility than the random one: this provides a way out of the indecision between random and vacuous classifier. Empirical experiments show that if a risk-averse utility is adopted, NCC is generally preferable to NBC. Utility-based accuracy measures constitute a promising approach to soundly compare traditional classifiers with classifiers based on imprecise probability.

It remains however open the problem of how to deal with cost-sensitive settings, in which different types of misclassification incur in different costs: so far, the analysis with the utility has been developed for settings in which all errors are regarded as equal.

Scoring the Conditional Probability of the Class

If the final aim is to approximate the conditional information of the class C given the attributes instead of optimizing the accuracy, then a different measure should be used. If conditioned to the attributes, what we obtain is a precise probability for the probability of the class P, then the usual score for each case is the logarithm of the likelihood score: $\log(P(c))$, where c is the true value of the class. This scored is added for the different cases in the test set. If instead of a single probability, P, what we obtain is a credal set $\mathcal{P}$, then there is not an obvious way of generalizing this score. Here we propose to use the following one: Let $\widehat{P}$ the probability having maximum entropy in $\mathcal{P}$, then we use the logarithm of the likelihood score computed with this maximum entropy probability. This should not be misinterpreted as a substitution of $\mathcal{P}$ by $\widehat{P}$. **In fact, it is only a method of scoring an imprecise probability through that distribution with maximum entropy.** . It is a situation similar to d-CCU metric. This metric can be seen as the result of distributing in an uniform way the set Z_i assigned by a credal classifier (if we obtain a precise classifier by this procedure, then in average it will obtain the same value of the metric than the imprecise one). This does not mean that the credal classifier is going to be transformed in a precise one. Analogously, using the maximum entropy distribution, $\widehat{P}$, is only a procedure to scoring the imprecise classifier given by $\mathcal{P}$.

11.6.1 Software

JNCC2 [144] is the Java implementation of NCC; it is available from `www.idsia.ch/~giorgio/jncc2.html` and runs through a command-line interface.

On occasion of this book, we also realized a plug-in for WEKA [677] which includes a number of credal classifiers, namely NCC, LNCC, CMA, and the credal classification trees. The plug-in implements the metrics of Section 11.6 and allows to run and compare all such classifier through the graphical user interface of WEKA. The integration with WEKA makes also available many tools for data analysis (e.g., feature selection), which can be thus used with credal classifiers. The WEKA plug-in is available under the GNU GPL license from the webpage `http://decsai.ugr.es/~andrew/weka-ip.html`.

11.6.2 Experiments

This section gives an idea of how the different algorithms compare and show that the integration with WEKA makes accessible several useful tools (e.g., feature selection) which can be used to improve the performance of credal classifiers. We show results obtained running the credal classifiers available within the WEKA plug-in (NCC, LNCC, CMA, IPtree) on 39 public data sets from the UCI repository. Firstly, we have analyzed the performance of the classifiers without feature selection; later, we have exploited the integration with WEKA to analyze the effect of feature selection on credal classifiers. In a couple of cases, NCC turns out to be more determinate than LNCC; this is due to the fact that NCC implements the restriction of the prior credal set of [141] (ϵ=0.01), while this feature is currently not yet implemented for LNCC. We do not compare credal classifiers

with their Bayesian counterparts (e.g., NCC vs NBC), as this kind of comparison is already extensively covered in the provided references.

The experiments have been performed using 10 runs of 10 folds cross-validation; numerical features have been discretized using the supervised discretization by [260]. For the sake of simplicity, we focus our analysis only on determinacy and discounted-accuracy; however, it would have been necessary to comprehensively consider the metrics of Section 11.6, if our goal was a deep analysis of the behaviors of the classifiers. The determinacy is important as it characterize the main practical difference between credal and traditional classifiers; the d-acc gives an overall evaluation of the performance of a credal classifier, which however tends to favor the more determinate classifiers. We can therefore expect some correlation between determinacy and d-acc. We have tested the significance of the differences through the t-test corrected for resampling ($\alpha = 0.01$), available within WEKA.

Without using feature selection, we find some known results: for instance, out of 39 data sets, LNCC is significantly more determinate than NCC in 18 cases and CMA in 23 cases (although in 5 cases NCC is significantly more determinate than CMA). In remaining cases, the differences are not significant. The IPtree is however the most determinate classifier: for instance, compared to CMA, it is 10 times significantly more determinate and only 3 times significantly less determinate (in the remaining 26 cases, the differences are not significant). The average determinacy of the classifiers across all data sets are respectively 0.81%, 0.85%, 0.93% and 0.95% for NCC, LNCC, CMA and IPtree.

As for the discounted-accuracy, LNCC has generally higher discounted-accuracy than NCC (12 wins, 25 ties, 2 losses) and so does CMA (12 wins, 25 ties, 2 losses). However, on the selected data sets, IPtree achieves better discounted-accuracy than both CMA (10 wins, 24 ties, 5 losses) and LNCC (8 wins, 27 ties, 4 losses). The average discounted-accuracy of the classifiers are respectively 0.72, 0.72., 0.75 and 0.76 for NCC, LNCC, CMA and IPtree.

We now analyze the effect of feature selection; we choose the Correlation-Based Feature Selection [320], which has been shown to be quite effective both with NBC and the C4.5 tree; it therefore appears as a reasonable choice for the set of credal classifiers which we are considering. We have designed the experiments to perform feature selection from scratch on each single experiment of cross-validation, using the data from the current training set; in this way, we avoid the optimistic bias due to feature selection, when feature selection is performed offline on the whole amount of available data [508].

A first effect of feature selection is that often it significantly increases determinacy: this happens in 10, 11, 7 and 3 times for respectively NCC, LNCC, CMA and IPtree; in no case feature selection implies a significant decrease of determinacy. Under feature selection, the average determinacy is respectively 86%, 89%, 95% and 95% for NCC, LNCC, CMA and IPtree.

Under feature selection, the discounted-accuracy of NCC significantly increases on 10 data sets, which are roughly the same over which it also significant increases its determinacy. A possible explanation is that feature selection mitigates, as side effect,

both the class and the feature problem.

Feature selection is beneficial also for LNCC, leading in 6 cases to a significant improvement of discounted-accuracy (although in one case there is a significant degradation); it is instead less effective for CMA (4 significant improvements, but also 1 significant deterioration); it should be however considered that the model averaging algorithms of CMA automatically remove irrelevant features. For the IPtree the effect of feature selection is more controversial: there is one significant improvement, but also three significant deteriorations. This can be due to the fact that trees automatically perform a kind of feature selection, when deciding on which feature to split, thus taking less advantage from an external step of feature selection. Under feature selection, the average discounted-accuracy is respectively 0.73, 0.74, 0.75, 0.75 for NCC, LNCC, CMA and IPtree. Results data by data set are available in Tables 11.1 and 11.2 of the Appendix.

Experiments Comparing Conditional Probabilities of the Class

A final experiment has been carried out to compare the estimation of probabilities of the IDM versus a Bayesian procedure. For this aim for each database, we have built a classification with the procedure used for the imprecise classification tree (IPtree). In the leaves, we have estimated the probabilities of the class variable, instead of assigning a set of values. We have considered different procedures the IDM and Bayesian methods based on a prior Dirichlet distribution with different values of s parameter.

Finally, we have scored the procedures with $\log \widehat{P}(c_i|\mathbf{n})$ where c_i is the true value of the class, $\mathbf{n}$ the vector of observations of the attributes, and $\widehat{P}$ is the maximum entropy distribution in the case of IDM model, or the estimated probability in the case of a precise Bayesian classifier.

The results can be seen in Table 11.3. The results for the IDM are in IDM-s columns (s=0.5,1,2) and the results for the Bayesian procedures are in B-s columns. For a value s the Bayesian procedures estimates the probabilities with a Dirichlet prior distribution with parameters uniform vector $\mathbf{t}$ and global sample size s. In these columns k is the number of different classes.

In the results we can see that $s = 1$ in the IDM provides the best results (in accordance with recommendations of this value of the parameter). Furthermore, this IDM provides better results than any of the Bayesian procedures, being the differences meaningful with 5 of these procedures using the Wilkonson signed-ranks test (see Table 11.4).

These results support the fact that there is no justification in using too precise procedures as the Bayesian ones. An imprecise method with a procedure to select one of the possible probabilities which is not based on information obtained from the data, can outperform the precise Bayesian procedures with a standard score as the log likelihood. The method to select the probability is maximum entropy, which can be justified as the distribution $\widehat{P}$ in the credal set with greatest value $\underline{P}(\ln(\widehat{P}(c)))$ (see the justification of maximum entropy as a global uncertainty measure in Subsection 11.5.1). That is, a procedure to select the distribution among a set of possible ones taking the log likelihood score as basis can be better than any of the precise Bayesian procedures. So, the imprecise credal set can be seen as a more faithful representation of the information content of the

data, without adding any additional information necessary to make the final probabilities precise.

Acknowledgments

Work partially supported by the Swiss NSF grant n. 200021-118071/1, 200020_137680 / 1, 200020_134759 / 1 and by the Hasler Foundation grant n. 10030.

Dataset	NCC	NCC-fsel		LNCC	LNCC-Fsel		CMA	CMA-fsel		IPtree	IPtree-fsel	
audiology	0.07	0.38	○	0.06	0.43	○	0.96 ○	1.00	○	0.95 ○	0.94	○
cmc	0.97	0.98		1.00 ○	1.00	○	0.93 ●	1.00	○	0.93 ●	0.98	
contact-lenses	0.65	0.65		0.64	0.63		0.96 ○	0.94	○	1.00 ○	0.96	○
credit	0.98	0.98		0.99 ○	1.00	○	0.98	0.99		0.96	0.96	
german-credit	0.96	0.98	○	1.00 ○	1.00	○	0.87 ●	0.97		0.89 ●	0.97	
pima-diabetes	0.99	0.99		1.00	1.00		1.00	1.00		0.97 ●	0.99	
ecoli	0.90	0.90		0.94 ○	0.94	○	1.00 ○	1.00	○	0.96 ○	0.96	○
eucalyptus	0.80	0.99	○	0.95 ○	0.98	○	0.99 ○	0.99	○	0.95 ○	0.99	○
glass	0.68	0.71		0.77 ○	0.81	○	1.00 ○	1.00	○	0.97 ○	0.96	○
grub-damage	0.61	0.57		0.74 ○	0.68		0.56	0.69		0.75 ○	0.76	○
haberman	0.95	0.95		1.00 ○	1.00	○	1.00 ○	1.00	○	0.96	0.96	
hayes-roth	0.52	0.52		0.52	0.52		0.59	0.59		0.59	0.59	
hepatitis	0.95	0.96		0.98	0.99	○	0.94	0.98		0.96	0.95	
hypothyroid	0.87	1.00	○	0.96 ○	1.00	○	1.00 ○	1.00	○	1.00 ○	1.00	○
ionosphere	0.97	0.96		0.97	0.99		1.00 ○	1.00	○	0.97	0.97	
iris	0.98	0.98		0.98	0.97		1.00	0.99		0.99	0.98	
kr-vs-kp	0.99	1.00	○	1.00 ○	1.00	○	0.96 ●	1.00	○	1.00 ○	1.00	○
labor	0.86	0.87		0.89	0.93		0.93	0.97	○	0.96	0.96	
liver-disorders	1.00	1.00		1.00	1.00		1.00	1.00		1.00	1.00	
lymphography	0.58	0.81	○	0.54	0.82	○	0.90 ○	0.96	○	0.97 ○	0.95	○
nursery	1.00	1.00		1.00 ○	1.00	○	1.00	1.00		1.00	1.00	
optdigits	0.98	0.98		0.99 ○	0.99	○	1.00 ○	1.00	○	0.90 ●	0.90	●
page-blocks	0.97	0.99	○	0.99 ○	1.00	○	1.00 ○	1.00	○	1.00 ○	1.00	○
pasture-prod.	0.63	0.76		0.61	0.76		0.99 ○	0.99	○	0.96 ○	0.96	○
primary-tumor	0.10	0.23	○	0.17 ○	0.38	○	0.88 ○	0.88	○	0.80 ○	0.80	○
segment	0.96	0.97	○	0.96	0.98	○	1.00 ○	1.00	○	0.98 ○	0.98	○
solar-flare-C	0.85	0.96	○	0.98 ○	1.00	○	0.98 ○	0.98	○	1.00 ○	1.00	○
sonar	0.96	0.96		0.99 ○	0.99	○	0.99 ○	0.99	○	0.95	0.95	
soybean	0.93	0.94		0.92 ●	0.93		1.00 ○	1.00	○	0.98 ○	0.99	○
spambase	1.00	1.00		1.00	1.00	○	1.00 ○	1.00	○	0.98 ●	0.99	●
spect-reordered	0.95	0.95		1.00 ○	1.00	○	0.85 ●	0.96		0.95	0.97	
splice	0.99	0.99	○	1.00 ○	1.00	○	1.00 ○	1.00	○	0.98 ●	0.98	
squash	0.42	0.43		0.36	0.48		0.84 ○	0.97	○	0.95 ○	0.93	○
tae	0.97	0.97		1.00	1.00		0.43 ●	0.43	●	1.00	1.00	
vehicle	0.93	0.92		0.99 ○	0.99	○	1.00 ○	1.00	○	0.97 ○	0.97	○
vowel	0.75	0.87	○	0.94 ○	0.98	○	1.00 ○	1.00	○	0.94 ○	0.93	○
white-clover	0.10	0.49	○	0.25 ○	0.66	○	0.95 ○	0.95	○	0.99 ○	0.99	○
wine	0.97	0.97		0.91 ●	0.91	●	1.00 ○	1.00	○	0.99	0.99	
yeast	0.97	0.97		0.99 ○	0.99	○	1.00 ○	1.00	○	0.97	0.97	
Average	0.81	0.86		0.85	0.89		0.93	0.95		0.95	0.95	

○, ● statistically significant increase or decrease

Table 11.1: Determinacy of the classifiers with and without feature selection. NCC represents the baseline, to which refer the increases and decreases.

Dataset	NCC	NCC-fsel	LNCC	LNCC-Fsel	CMA	CMA-fsel	IPtree	IPtree-fsel
audiology	0.21	0.47 ○	0.20 ●	0.48 ○	0.73 ○	0.74 ○	0.79 ○	0.76 ○
cmc	0.50	0.51	0.50	0.51	0.50	0.51	0.49	0.52
contact-lenses	0.76	0.73	0.76	0.72	0.85	0.81	0.84	0.77
credit	0.86	0.86	0.84	0.85	0.86	0.86	0.84 ●	0.84
german-credit	0.75	0.73	0.74	0.73	0.73	0.74	0.68 ●	0.72
pima-diabetes	0.75	0.76	0.75	0.76	0.75	0.76	0.74	0.75
ecoli	0.80	0.80	0.81	0.81	0.81	0.81	0.80	0.80
eucalyptus	0.53	0.58 ○	0.59 ○	0.58 ○	0.56 ○	0.58 ○	0.62 ○	0.58 ○
Glass	0.63	0.63	0.66 ○	0.67 ○	0.72 ○	0.70 ○	0.69 ○	0.67
grub-damage	0.46	0.44	0.46	0.45	0.34 ●	0.31 ●	0.36 ●	0.38 ●
haberman	0.72	0.72	0.73	0.73	0.72	0.72	0.73	0.73
hayes-roth	0.58	0.58	0.58	0.58	0.58	0.58	0.58	0.58
hepatitis	0.84	0.83	0.84	0.83	0.84	0.82	0.80	0.81
hypothyroid	0.93	0.98 ○	0.97 ○	0.97 ○	0.99 ○	0.98 ○	0.99 ○	0.98 ○
ionosphere	0.89	0.90	0.88	0.90	0.90	0.91	0.90	0.91
iris	0.93	0.94	0.93	0.94	0.93	0.94	0.93	0.94
kr-vs-kp	0.88	0.92 ○	0.95 ○	0.93 ○	0.88	0.92 ○	0.99 ○	0.94 ○
labor	0.89	0.85	0.89	0.85	0.88	0.86	0.84	0.84
liver-disorders	0.57	0.57	0.57	0.57	0.57	0.57	0.57	0.57
lymphography	0.68	0.75 ○	0.66 ●	0.75 ○	0.81 ○	0.78 ○	0.73	0.75
nursery	0.90	0.90	0.96 ○	0.95 ○	0.90 ●	0.90 ●	0.96 ○	0.95 ○
optdigits	0.92	0.92	0.94 ○	0.94 ○	0.92	0.92	0.77 ●	0.77 ●
page-blocks	0.93	0.95 ○	0.96 ○	0.97 ○	0.94 ○	0.96 ○	0.96 ○	0.96 ○
pasture-prod.	0.75	0.73	0.72	0.73	0.80	0.76	0.73	0.73
primary-tumor	0.19	0.26 ○	0.22 ○	0.29 ○	0.36 ○	0.36 ○	0.38 ○	0.37 ○
segment	0.92	0.93 ○	0.93 ○	0.94 ○	0.93 ○	0.94 ○	0.94 ○	0.94 ○
solar-flare-C	0.81	0.86 ○	0.87 ○	0.88 ○	0.89 ○	0.89 ○	0.90 ○	0.89 ○
sonar	0.76	0.77	0.76	0.75	0.77	0.77	0.73	0.72
soybean	0.92	0.91	0.91 ●	0.90 ●	0.92	0.90	0.92	0.92
spambase	0.90	0.92 ○	0.94 ○	0.93 ○	0.90	0.92 ○	0.92 ○	0.92 ○
spect	0.79	0.79	0.83	0.84	0.77 ●	0.80	0.79	0.81
splice	0.95	0.96	0.65 ●	0.80 ●	0.96 ○	0.96	0.93 ●	0.93 ●
squash-stored	0.57	0.49 ●	0.50 ●	0.46 ●	0.59	0.61	0.65	0.61
tae	0.46	0.46	0.47	0.47	0.38 ●	0.38 ●	0.47	0.47
vehicle	0.61	0.60	0.69 ○	0.67 ○	0.61	0.60	0.69 ○	0.69 ○
vowel	0.58	0.59	0.66 ○	0.65 ○	0.63 ○	0.60	0.76 ○	0.68 ○
white-clover	0.39	0.50 ○	0.46 ○	0.55 ○	0.54 ○	0.54 ○	0.62 ○	0.61 ○
wine	0.98	0.98	0.93 ●	0.94 ●	0.98	0.98	0.92 ●	0.92 ●
yeast	0.57	0.57	0.58	0.58	0.58	0.58	0.57	0.57
Average	0.72	0.73	0.72	0.74	0.75	0.75	0.76	0.75

○, ● statistically significant improvement or degradation

Table 11.2: Discounted accuracy of the classifiers with and without feature selection. NCC represents the baseline, to which refer the increases and decreases.

Dataset	IDM-0.5	IDM-1.0	IDM-2.0	B-0.5*k	B-1.0*k	B-2.0*k	B-0.5	B-1.0	B-2.0
anneal	-0.051	-0.070	-0.107	-0.117	-0.185	-0.295	-0.047	-0.062	-0.091
audiology	-1.388	-1.354	-1.374	-1.748	-2.063	-2.459	-1.395	-1.355	-1.365
autos	-1.132	-1.138	-1.228	-1.261	-1.436	-1.671	-1.146	-1.128	-1.172
breast-c.	-0.901	-0.859	-0.859	-0.917	-0.875	-0.847	-0.973	-0.917	-0.875
cmc	-1.505	-1.449	-1.434	-1.491	-1.446	-1.416	-1.591	-1.524	-1.471
horse-c.	-0.668	-0.626	-0.626	-0.685	-0.643	-0.618	-0.742	-0.685	-0.643
german-c.	-0.940	-0.855	-0.855	-0.971	-0.890	-0.835	-1.076	-0.971	-0.890
pima-dia.	-0.775	-0.761	-0.761	-0.779	-0.766	-0.755	-0.794	-0.779	-0.766
glass2	-0.735	-0.720	-0.720	-0.739	-0.722	-0.718	-0.766	-0.739	-0.722
hepatitis	-0.755	-0.676	-0.676	-0.771	-0.694	-0.640	-0.864	-0.771	-0.694
hypoth.	-0.045	-0.047	-0.054	-0.048	-0.055	-0.067	-0.045	-0.045	-0.048
ionosphere	-0.445	-0.423	-0.423	-0.451	-0.425	-0.418	-0.489	-0.451	-0.425
kr-vs-kp	-0.033	-0.039	-0.039	-0.033	-0.038	-0.048	-0.031	-0.033	-0.038
labor	-0.663	-0.616	-0.616	-0.672	-0.613	-0.586	-0.755	-0.672	-0.613
lymph.	-1.216	-1.119	-1.075	-1.079	-1.047	-1.071	-1.277	-1.160	-1.079
mushroom	-0.002	-0.004	-0.004	-0.002	-0.004	-0.008	-0.001	-0.002	-0.004
segment	-0.284	-0.297	-0.339	-0.366	-0.460	-0.603	-0.285	-0.291	-0.320
sick	-0.114	-0.113	-0.113	-0.115	-0.113	-0.112	-0.116	-0.115	-0.113
solar-flare	-0.172	-0.168	-0.168	-0.174	-0.169	-0.167	-0.180	-0.174	-0.169
sonar	-0.940	-0.833	-0.833	-0.965	-0.862	-0.789	-1.091	-0.965	-0.862
soybean	-0.451	-0.482	-0.560	-0.962	-1.306	-1.746	-0.449	-0.476	-0.546
sponge	-0.419	-0.415	-0.426	-0.423	-0.422	-0.435	-0.441	-0.428	-0.421
vote	-0.229	-0.219	-0.219	-0.231	-0.220	-0.215	-0.246	-0.231	-0.220
vowel	-1.306	-1.305	-1.378	-1.549	-1.792	-2.095	-1.314	-1.299	-1.343
zoo	-0.346	-0.398	-0.515	-0.582	-0.798	-1.088	-0.339	-0.376	-0.463
Average	-0.621	-0.600	-0.616	-0.685	-0.722	-0.788	-0.658	-0.626	-0.614

Table 11.3: Maximum entropy log-likehood of IDM versus Bayesian scores

Wilcoxon Signed-Ranks Test					
B-0.5	B-1.0*k	B-2.0*k	B-0.5	B-1.0	B-2.0
$z = -3.86$	$z = -3.21$	$z = -0.79$	$z = -2.89$	$z = -2.38$	$z = -3.26$
$p < 0.01$	$p < 0.01$	$p > 0.1$	$p < 0.01$	$p < 0.05$	$p < 0.01$

Table 11.4: Test results: z is the value of the normal two tailed distribution and p the significance level.