Machine Learning (2020) 109:1039–1099
https://doi.org/10.1007/s10994-020-05869-5

Joint maximization of accuracy and information for learning the structure of a Bayesian network classifier

Dan Halbersberg[1] · **Maydan Wienreb**[1] · **Boaz Lerner**[1]

Received: 24 July 2017 / Revised: 11 May 2019 / Accepted: 30 January 2020 / Published online: 28 February 2020

Abstract
Although recent studies have shown that a Bayesian network classifier (BNC) that maximizes the classification accuracy (i.e., minimizes the 0/1 loss function) is a powerful tool in both knowledge representation and classification, this classifier: (1) focuses on the majority class and, therefore, misclassifies minority classes; (2) is usually uninformative about the distribution of misclassifications; and (3) is insensitive to error severity (making no distinction between misclassification types). In this study, we propose to learn the structure of a BNC using an information measure (IM) that jointly maximizes the classification accuracy and information, motivate this measure theoretically, and evaluate it compared with six common measures using various datasets. Using synthesized confusion matrices, twenty-three artificial datasets, seventeen UCI datasets, and different performance measures, we show that an IM-based BNC is superior to BNCs learned using the other measures—especially for ordinal classification (for which accounting for the error severity is important) and/or imbalanced problems (which are most real-life classification problems)—and that it does not fall behind state-of-the-art classifiers with respect to accuracy and amount of information provided. To further demonstrate its ability, we tested the IM-based BNC in predicting the severity of motorcycle accidents of young drivers and the disease state of ALS patients—two class-imbalance ordinal classification problems—and show that the IM-based BNC is accurate also for the minority classes (fatal accidents and severe patients) and not only for the majority class (mild accidents and mild patients) as are other classifiers, providing more informative and practical classification results. Based on the many experiments we report on here, we expect these advantages to exist for other problems in which both accuracy and information should be maximized, the data is imbalanced, and/or the problem is ordinal, whether the classifier is a BNC or not. Our code, datasets, and results are publicly available http://www.ee.bgu.ac.il/~boaz/software.

Keywords 0/1 loss function · Bayesian network classifiers · Class imbalance · Information measures · Ordinal classification · Structure learning

Editor: James Cussens.

✉ Dan Halbersberg
halbersb@post.bgu.ac.il

Extended author information available on the last page of the article

1 Introduction and related work

Classifiers, e.g., the neural network (NN), random forest (RF), and support vector machine (SVM), excel in prediction but not in knowledge representation, which is needed in problems for which key factor identification is sought, such as in an attempt to understand possible causes of accidents, a disease, or a machine/process fault. The Bayesian network (BN) excels in knowledge representation, which makes it ideal to identify key factors, but it is not considered a supreme classifier. To achieve high accuracy (ACC), learning the structure of a BN classifier (BNC) should maximize a (discriminative) score that is specific to classification and not a generative one based on the likelihood function that may fit a general BN structure, but not necessarily that of a BNC structure. Indeed, when a BNC was learned to minimize the 0/1 loss function, it showed superiority to BNCs learned using marginal and class-conditional likelihood-based scores and even to state-of-the-art classifiers like NN and SVM (Kelner and Lerner 2012).

However, by maximizing accuracy (minimizing the 0/1 loss function) in learning its structure, the BNC—similar to other machine learning classifiers—cannot account for the error distribution and, thus, is not informative enough about the classification result and the contribution of each class to the error (Provost et al. 1998; García et al. 2010), and it may also be sub-optimal (Ranawana and Palade 2006). Other discriminative measures used in learning a classifier, such as the area under curve (AUC), suffer from the same shortcoming, because they all relate to ACC. Moreover, in most cases, these measures only suit binary classification problems. Also, it may explain why other studies (García et al. 2009) suggested measures such as the consensus measure of accuracy.

On the other hand, measures that maximize information and account for error distribution, e.g., mutual information (MI) (Cover and Thomas 2012), the Matthew correlation coefficient (MCC) (Baldi et al. 2000), and the confusion entropy (CEN) (Wei et al. 2010) usually are not accurate enough. Labatut and Cherifi (2011) claimed that most of the non-accuracy measures were initially developed for other purposes than to compare/evaluate classifiers (e.g., to measure the association between two random variables, the alignment between two raters, or the similarity between two sets). Therefore, they may lead to confusing terminology or even to wrong interpretation, or they may be noisy and ad hoc for a particular problem.

A second challenge for a BNC, as well as for all other machine-learning classifiers, is that for imbalanced data, they usually predict all (or almost all, depending on the imbalance level) samples of the minority classes as of the majority class. These classifiers show high accuracy, which is in the order of the prior probability of the majority class, since they classify all samples to this class, but at the same time, they may misclassify all samples of the minority classes. Class imbalance can traditionally be tackled using different approaches, e.g., random sampling—upsampling the minority class(es) or downsampling the majority class (Chawla 2005; Provost 2000). However, these two sampling methods result in over-fitting and domain deformation or loss of data, respectively. In addition, tackling imbalance by random downsampling or upsampling, or applying different costs to different misclassifications provides an optimistic ACC estimate, and thus is not recommended (Provost 2000). Also other accuracy-driven measures, e.g., precision, sensitivity, and specificity lead to sub-optimal solutions in the presence of class imbalance (Ranawana and Palade 2006). More advanced methods to tackle class imbalance include feature selection (Wasikowski and Chen 2010); sampling subsets of the classes (Liu et al. 2009); combination of down- and upsampling using e.g., the synthetic minority over-sampling technique

(SMOTE) (Chawla et al. 2002); combination of down–upsampling with an ensemble of classifiers (Galar et al. 2012) or with feature selection (Lerner et al. 2007); cost-sensitive learning (Domingos 1999); measuring the balanced accuracy (over all classes) (Brodersen et al. 2010) or its geometric mean (García et al. 2010); and hierarchical decomposition of the classification task, where each hierarchy level is designed to tackle a simpler problem that is represented by classes that are approximately balanced (Lerner et al. 2007). Although probably never tested, classifiers—BNCs and others—learned using information measures such as MI, MCC, and CEN should be less affected by class imbalance data but at the same time also less accurate.

A third challenge is that 0/1 loss-function classifiers do not account differently for different error severities, as they count all misclassifications the same, both for performance evaluation and in learning. However, when the class (target) variable is ordinal, exploiting the ordinal nature of this variable may facilitate learning the classifier and make it more accurate. Considering an ordinal target variable Y, taking one of M values, such that $V_1 < \cdots < V_M$, a learning algorithm can take into account the natural ordering of this variable to induce a classifier, which harnesses this extra information to improve its accuracy. One such classifier is the cumulative probability tree (Frank and Hall 2001), for which Y is transformed into $M-1$ binary variables such that the ith binary variable represents the test $Y > V_i$. The model then comprises $M-1$ tree classifiers, where the ith tree is trained to output $P(Y > V_i)$. Another ordinal classifier is the cumulative link model (CLM) (Agresti 2011) that is an extension of the generalized linear model (GLM) for ordinal classification. A third ordinal classifier is the ordinal decision tree, which generalizes the classification and regression tree (CART) (Breiman et al. 1984) to ordinal target variables by considering splitting functions based on ordinal impurity functions (Piccareta 2008), which are specific implementations of the generalized Gini impurity function for a node. Principally—although we are not aware of any such study—the mean absolute error, MAE, (Hyndman and Koehler 2006), which sometimes is used to evaluate the error between a prediction and the true value, may also be used to augment learning an ordinal classifier. While such a measure can capture the ordinal information in a problem and potentially penalize different errors differently as we desired, it is not informative regarding the error distribution and is still sensitive to class imbalance.

To motivate this study further, let's consider two examples. The first is prediction of the severity of young-driver (YD) motorcycle accidents (MAs). Road injuries are the leading cause of death among YDs (ages 18–24) (Toledo et al. 2012); YDs make up 9–13% of the population, but their percentage in driver fatalities is 18–30% (OECD 2006). Besides the tragic human cost, a fatal accident costs (OECD 2006) around \$1.5M, where in the US alone, the cost of YD road accidents in 2002 was \$40 billion. MAs are particularly deadly, and luckily fatal MAs are only $\sim 1\%$ of all accidents, whereas severe and minor accidents are around 12% and 87% of the accidents, respectively. However, experiments show that MA classifiers tend to focus on the majority class of minor accidents at the expense of the minority classes of severe and fatal accidents (Halbersberg and Lerner 2019). In addition they are uninformative about their error distribution and are insensitive to error severity (making, e.g., no distinction between misclassification of fatal accidents as severe or minor although the former is less harsh than the latter). Road-safety experts wish their MA classifier to not only maximize accuracy, but also to be informative about its errors, to be as indifferent as possible to data imbalance between minor and fatal accidents, and to penalize misclassifications of fatal accidents as severe and as minor differently.

The second example is prediction of the disease state of an ALS patient. Amyotrophic lateral sclerosis (ALS) is a devastating neurodegenerative illness of the human motor

system with an unknown pathogenesis (Kiernan et al. 2011), which is still not visibly affected by the therapies available today, and from which 50% of patients die within three to five years of onset, and about 20% survive between five to ten years (Mitchell and Borasio 2007; Kiernan et al. 2011). The ALS functional rating scale (ALSFRS) is a widely accepted metric in the ALS medical community for the evaluation of ALS-related disability and progression (Brooks et al. 1996), with values between 0 for no functionality and 4 for full functionality for ten ALSFRS items describing physical functionalities in, e.g., breathing, speaking, and walking. By considering the ALSFRS as the target (class label), we may define ALS disease state prediction as an ordinal problem. With respect to the relative frequencies of ALSFRS values, which typically may vary from around 1% for ALSFRS of 0 to 42% and 35% for values of 3 and 4, respectively, disease state prediction also becomes a class imbalance problem. ALS patients, along with their doctors and carers, wish for disease state prediction to be very accurate (Gordon and Lerner 2019) but at the same time informative, to not be fooled by the imbalance among disease states, and to consider mild misclassification less harshly than severe misclassification.

In this study, we propose to learn a BNC, which leverages knowledge representation, using measures replacing the 0/1 loss function and trading accuracy and information. We are interested in learning the BNC using a measure that maximizes both accuracy and information, considers the error distribution, admits class imbalance, and accounts for error severity (which is significant only for ordinal problems). First, we consider existing measures, such as MI, MCC, and CEN, that all use the entire confusion matrix and not just its diagonal (as ACC) and, therefore, have the potential to meet at least some of our concerns. In addition, we evaluate the MAE, which naturally accounts for error severity. Second, since none of these measures accounts for *all* concerns, we propose next a novel information measure (IM), trading accuracy and information, that accounts for all of them. Third, we extend this measure further, adding to it a term that trades off accuracy and IM, giving the measure an additional degree of flexibility. Then we motivate the proposed measures and thoroughly evaluate them, comparing them with the existing measures theoretically and using several performance measures (which are the same learning measures), synthesized confusion matrices, artificial datasets, UCI ordinal datasets, and three real ordinal problems. We show the advantages of the IM-based BNC compared with BNCs that are learned using alternative measures and other state-of-the-art classifiers with respect to maximization of accuracy and information in ordinal class-imbalance problems. These advantages are manifested here for many databases and several real-world problems, but we believe they hold true for other problems (e.g., ranking problems) having the same requirements, and for classifiers other than the BNC.

In summary, our contribution is that: (1) We propose to utilize the BNC using a measure replacing the 0/1 loss function to jointly maximize accuracy and information, consider the error distribution, admit class imbalance, and account for error severity in tackling class-imbalance ordinal classification problems; (2) Since our theoretical and empirical evaluation of existing measures showed that none of the existing measures accounts for all these concerns, we suggest a novel information measure (IM) that has all the above desired properties; (3) We motivate the proposed measure and thoroughly evaluate it theoretically in comparison with the existing measures and empirically using several performance measures, synthesized confusion matrices, artificial datasets, UCI ordinal datasets, and three real ordinal problems; and (4) We demonstrate the advantages of the IM-based BNC compared with BNCs that are learned using existing measures and with other state-of-the-art classifiers (e.g., NN, SVM, BNC, and RF) with respect to maximization of accuracy and information in ordinal class-imbalance problems. We manifested these advantages using

many databases and several real-world problems, and we believe these hold true for other problems (e.g., ranking problems) having the same requirements, and for classifiers other than the BNC.

The rest of this paper is organized as follows. In Sects. 2 and 3, we review the BNC and candidate measures for learning its structure, respectively. In Sect. 4, we propose new measures for learning a BNC and demonstrate how to control their values to trade learning among the conflicting requirements of accuracy, information, and error severity. In Sect. 5, we experimentally evaluate our information measures comparing them with existing measures using synthesized confusion matrices that pose different classification scenarios and challenges. In Sect. 6, we expand our evaluation and compare empirically BNCs learned based on our (as well as other) measures with state-of-the-art classifiers using databases representing artificial and real-world problems. Finally in Sect. 7, we summarize the study and draw important conclusions.

2 Bayesian network classifiers

The BN compactly represents the joint probability distribution P over a set of variables $X = \{X_1, \ldots, X_n\}$, each, in the discrete case, having a finite set of mutually exclusive states. It consists of a network structure G and a set of parameters θ, where $G = (V, E)$ is a directed acyclic graph in which the nodes V in G are in one-to-one correspondence with the variables in X, and the edges E in G encode a set of conditional independence assertions about variables in X. θ consists of local probability distributions, each for each variable X_i given its parents $PA(X_i)$ in G. Given the network, the joint probability distribution over X comprises the local distributions as (Heckerman 1998):

$$P(X_1, X_2, ..., X_n) = \prod_{i=1}^{n} P(X_i|PA(X_i)). \tag{1}$$

Learning the structure of the BN from a dataset D is NP hard (Cooper and Herskovits 1992), and thus is usually performed heuristically and sub-optimally using, e.g., the search and score (S&S) approach by which the structure that maximizes a score function, which measures the fitness of the structure to the data, is selected. One such score (measure) is the a posteriori probability of the network given the data, $P(G|D)$ (or the marginal likelihood, $P(D|G)$, for equally probable structures) (Cooper and Herskovits 1992), and another measure is based on the minimum description length principle (Lam and Bacchus 1994), penalizing model complexity, where both scores are asymptotically equivalent and correct. However, these scores, similar to other log likelihood (LL) or information-based scores, either likelihood-equivalent or not (Heckerman et al. 1995), cannot optimize a classifier (Friedman et al. 1997) because they are not directed in maximizing the classification accuracy. Instead, it was suggested to learn a BNC by maximizing the conditional log likelihood (CLL) of G given D (Grossman and Domingos 2004):

$$CLL(G|D) = \log \prod_{i=1}^{N} P(c_i|v'_i) = \sum_{i=1}^{N} \log P(c_i|v'_i) = LL(G|D) - \sum_{i=1}^{N} \log P(v'_i), \tag{2}$$

where v'_i and c_i are the feature vector and class label, respectively, for the ith of N instances. However, the computation of CLL is exponential in the number of instances N, and also,

although CLL is asymptotically correct, for a finite sample, the class maximizing CLL can only indicate the correct classification, but it can not guarantee it (Kelner and Lerner 2012).

A score that measures the degree of compatibility between a possible state of the class variable C and the correct class is the 0/1 loss function:

$$L(c_i, \widehat{c}_i) = \begin{cases} 0, & c_i = \widehat{c}_i \\ 1, & c_i \neq \widehat{c}_i \end{cases}, \tag{3}$$

where $\widehat{c}_i$ is the estimated class label for the ith instance. Instead of selecting a structure based on summation of supervised marginal likelihoods over the dataset (2), the risk minimization by cross validation (RMCV) score selects a structure based on summation of false decisions about the class state over the dataset (Kelner and Lerner 2012),

$$RMCV(D, G) = \frac{1}{K} \sum_{k=1}^{K} \frac{K}{N} \sum_{i=1}^{N/K} L(c_{ki}, \arg\max_c P(C = c | v'_{ki}, D \backslash D_k^K, G)), c \in \{c_1, \ldots, c_M\}, \tag{4}$$

where the training set D is divided into K non overlapping validation sets D_k^K (each having N/K instances of the form $v_{ki} = (c_{ki}, v'_{ki})$), and for each such validation set, an effective training set has $|D \setminus D_k^K|$ (i.e., $N(K-1)/K$) instances. As part of the cross validation (CV), the classification error rate, i.e., the RMCV score, is measured on all vectors of D_k^K and averaged over the K validation sets. No use of the test set is made during learning. Note that the RMCV score is normalized by the dataset size N, whereas (2) is not. Although normalization has the same effect on all learned structures, it can clarify the meaning of the score (i.e., an error rate) and help in comparing scores over datasets. Moreover, sharing the same range of values ([0, 1]), RMCV establishes its correspondence to classification accuracy. Also, note that the same RMCV measure can be used for learning the BNC and for evaluating its accuracy, which makes learning oriented towards classification.

To compute the RMCV, the candidate structure has to be turned into a classifier by learning its parameters. Local probabilities are modeled using the unrestricted multinominal distribution (Heckerman 1998), where the distribution parameters are obtained using maximum likelihood (ML) (Cooper and Herskovits 1992), similar to (Kontkanen et al. 1999). Moreover, it has been empirically shown (Grossman and Domingos 2004) that ML parameter estimation does not deteriorate the results compared to maximum conditional likelihood estimation, which can only be obtained by computationally expensive numerical approximation. Learning a BN rather than a structure has an additional cost of parameter learning, though this cost is negligible while using ML estimation and fully observed data.

Starting with the empty or naïve Bayesian graph and using a simple hill-climbing search with the RMCV score establish the RMCV structure learning algorithm for BNCs (Kelner and Lerner 2012). The hill-climbing implementation includes a search over all neighbor graphs at each iteration. A neighbor graph is defined as a single modification of the current graph using one of the following operators: edge addition, deletion, or reversal provided that the derived graph remains a directed acyclic graph. The RMCV BNC showed superiority to other BNCs and state-of-the-art classifiers using synthetic and UCI datasets and, thus, is used in this study to represent a BNC. However, as it is based on the 0/1 loss function, RMCV, similar to other classifiers, is prone to all weaknesses of classifiers as described in Sect. 1.

Table 1 A confusion matrix for a three-class classification problem

Predicted class (X)	True class (Y)		
	Class 1	Class 2	Class 3
Class 1	C_{11}	C_{12}	C_{13}
Class 2	C_{21}	C_{22}	C_{23}
Class 3	C_{31}	C_{32}	C_{33}

3 Evaluating classifier performance

How can we know whether the classification model we have constructed is the most suitable one? Performance measures that evaluate multi-class classifiers are usually based on the confusion matrix between predicted and true classes (Baldi et al. 2000). Although this matrix summarizes all correct and wrong predictions (Table 1), and thereby may represent the classifier error distribution, the common way to evaluate classifier performance is based on the classification accuracy (Ferri et al. 2009; Jurman et al. 2012), i.e., the 0/1 loss function (RMCV score), which is the (normalized) matrix trace.

However, researchers have made claims against the use of accuracy (Ranawana and Palade 2006; García et al. 2010). Provost et al. (1998) and Chawla (2005) argued that accuracy ignores misclassification costs and, therefore, may lead to misleading conclusions. Brodersen et al. (2010) concluded that even while CV is used, measuring performance by accuracy has two critical shortcomings: first, it is a non-parametric approach that does not make it possible to compute a meaningful confidence interval of a true underlying quantity. Second, it does not properly handle imbalanced datasets. As we noted above, upsampling the minority class or downsampling the majority class result in over-fitting and domain deformation or loss of data, respectively. Also, tackling data imbalance by these methods provides an optimistic accuracy estimate and, thus, is not recommended (Provost et al. 1998). Others have stated that accuracy is inappropriate when there are a great number of classes (Caballero et al. 2010).

Indeed, many studies have been conducted trying to suggest other measures for evaluating the classifier performance. For example, Wallace and Boulton (1968) suggested measuring the goodness of classification based on the minimum message length borrowed from information theory. Ferri et al. (2009) compared and analyzed relationships of 18 classifier performance measures. They concluded that measures providing a qualitative understanding of error, such as accuracy, perform badly when distortion occurs during the learning phase because the dataset is too small or a bad algorithm is used. Moreover, they confirmed that some measures suffer from the imbalanced data limitation. They offered to use the area under curve (AUC) measure. Baldi et al. (2000) compared nine binary classifier performance measures, among them information measures and quadratic error measures. However, none of them adequately combines information, error severity, and ways to handle class imbalance.

Before we start reviewing relevant classifier performance measures, let's recall that besides the question of which measure to use to evaluate a classifier, there is also the question of which measure to use for learning (training the classifier). Not always are the two measures the same, which raises the question why. For example, the NN and RF classifiers are evaluated using classification accuracy, but usually are trained according to some (non classification) error and information gain, respectively. This was also the case with BNCs,

until very recently (Kelner and Lerner 2012), when the classifiers were trained according to an LL-driven measure, but evaluated using accuracy.

Following, we review several common measures as a replacement for the classification accuracy for learning and evaluating a BNC.

3.1 Mutual information

In information theory and statistics, entropy is used to measure the uncertainty about a certain variable (Cover and Thomas 2012). If X is a discrete random variable with K values, then the information content in each value k of this variable is $h(k) = -\log P(X = k)$. Therefore, a less likely value of X contains more information than a highly probable one. The entropy is the average information content of X that is distributed according to P:

$$H(X) = -\sum_{k=1}^{K} P(X = k) \log P(X = k), \tag{5}$$

where in this paper, we use the natural base logarithm. Similarly, the joint entropy between two variables X and Y, which measures how much uncertainty there is in the two variables together, is defined as: $H(X, Y) = -\sum_{x,y} P(x, y) \log P(x, y)$ (Cover and Thomas 2012).

The mutual information (MI) between X and Y can be defined as the reduction in entropy (uncertainty) of Y by the conditional entropy of Y on X, i.e., $I(X;Y) = H(Y) - H(Y|X)$. For classification, if X and Y are holding predictions and true values, respectively, MI measures the reduction in uncertainty for the true class $Y = y$ due to the prediction $X = x$ (Baldi et al. 2000),

$$MI = I(X;Y) = \sum_{x} \sum_{y} P(x, y) \log \left(\frac{P(x, y)}{P(x)P(y)} \right). \tag{6}$$

Since MI measures how prediction decreases the uncertainty regarding the true class, we should prefer a classifier with a high MI value.

3.2 Confusion entropy

The confusion entropy (CEN) (Wei et al. 2010) exploits the distribution of misclassifications of a class as any other of $M - 1$ classes and of the $M - 1$ classes as that class:

$$CEN = \sum_{m=1}^{M} P_m CEN_m, \tag{7}$$

where P_m refers to the confusion probability of class m,

$$P_m = \frac{\sum_{k=1}^{M} (C_{m,k} + C_{k,m})}{2 \sum_k \sum_l C_{k,l}}, \tag{8}$$

where $C_{m,k}$ is the (m, k) element of the confusion matrix between X and Y.

The denominator for all classes is equal to the sum of all confusion matrix elements multiplied by two, and the numerator for P_m equals the sum of row m and column m (i.e.,

the sum of all samples that belong to class m and those that were classified to class m). CEN_m refers to the confusion entropy of class m,

$$CEN_m = \sum_{k \neq m} (P^m_{m,k} \log_{2M-2}(P^m_{m,k}) + P^m_{k,m} \log_{2M-2}(P^m_{k,m})), \tag{9}$$

where $P^m_{k,m}$ is the probability of misclassifying samples of class k to class m subject to class m,

$$P^m_{k,m} = \frac{C_{k,m}}{\sum_{j=1}^{M}(C_{m,j} + C_{j,m})}, \quad \forall k \neq m, \tag{10}$$

i.e., the misclassification is normalized by the sum of all samples that belong to class m and those that were classified as class m.

For an M class problem, the misclassification information involves both information on how the samples with true class label c_i have been misclassified to one of the other $M-1$ classes and information on how the samples of the other $M-1$ classes have been misclassified to class c_i (Wei et al. 2010).

3.3 Matthew correlation coefficient

The Matthew correlation coefficient (MCC), known also as the Pearson correlation, has been used in the binary classification case (Baldi et al. 2000). Its generalization to the multiclass problem was introduced by (Gorodkin 2004), where MCC is the correlation between the true ($\mathbf{U}$) and predicted ($\mathbf{V}$) class matrices (Jurman et al. 2012),

$$MCC = \frac{COV(\mathbf{U}, \mathbf{V})}{\sqrt{COV(\mathbf{U}, \mathbf{U})COV(\mathbf{V}, \mathbf{V})}}. \tag{11}$$

$\mathbf{U}$ and $\mathbf{V}$ are $N \times M$, and N and M are the numbers of samples and classes, respectively, and $COV(\mathbf{U}, \mathbf{V})$ is:

$$COV(\mathbf{U}, \mathbf{V}) = \frac{1}{M} \sum_{m=1}^{M} \sum_{i=1}^{N} (u_{im} - \overline{u}_m)(v_{im} - \overline{v}_m), \tag{12}$$

where the average prediction and true value of class m are $\overline{v}_m = \frac{1}{N} \sum_{i=1}^{N} v_{im}$ and $\overline{u}_m = \frac{1}{N} \sum_{i=1}^{N} u_{im}$, respectively.

Consider the case in which the class variable is perfectly balanced and all off-diagonal entries in the confusion matrix are F, for false, and all main diagonal entries are T, for true. That is, F is the number of misclassifications of class i to class j, $\forall j \neq i$ (and thus there are $(M-1)F$ misclassifications for each class), and T is the number of correct classifications of class i, $\forall i$. A strong (monotone) connection between CEN and MCC for this case is (Jurman et al. 2012):

$$CEN = (1 - MCC)\left(1 + \log_{2M-2}\left(\frac{T + (M-1)F}{(M-1)F}\right)\right)\left(1 - \frac{1}{M}\right). \tag{13}$$

According to (13), the relation between CEN and MCC depends on the *log* of the ratio of the number of samples belonging to class i (in this case, this number is shared by all classes as the class variable is perfectly balanced) to the number of misclassifications of this class.

Similarly, we can write the relationship between MI and MCC as:

$$MI = \log(MCC) + \frac{T\log(T) + (M-1)F\log(F)}{T + (M-1)F} + \log\left(\frac{M[T + (M-1)F]}{T^2 + (M-2)TF - (M-1)F^2}\right), \tag{14}$$

and for the case for which $F = 1$ and $T \gg M$, we can derive an approximation:

$$MI \approx \log(MCC) + \log(M).$$

3.4 Mean absolute error

The mean absolute error (MAE) measures the prediction error as the average deviation of the predicted class vector (X) from the true class vector (Y) (Hyndman and Koehler 2006),

$$MAE = \sum_x \sum_y P(x,y)|x-y|, \tag{15}$$

which is the sum of all possible errors, each is the (x, y) element of the confusion matrix, weighted by their relative prevalence according to the confusion matrix, $P(x, y)$.

4 Trading between information and accuracy

As our experimental evaluation shows (Sect. 5), when applied in learning a BNC, none of the presented measures can accomplish all we ask—maximization of both accuracy and information, tackling class imbalance, and accounting for error severity. By using the joint probability distribution $P(x, y)$ between predictions X and true classes Y (as in Sects. 3.1 and 3.4, where (x, y) is an element in the confusion matrix), we suggest the information measure (IM) that balances between the mutual information between X and Y (Sect. 3.1) and a score, we call total error severity (ES), that evaluates the classifier error simultaneously over all classes, penalizing errors by their severity (Halbersberg and Lerner 2016),

$$\begin{aligned} IM &= -MI(X,Y) + ES(X,Y) \\ &= \sum_x \sum_y P(x,y)\left(-\log\left(\frac{P(x,y)}{P(x)P(y)}\right) + \log(1+|x-y|)\right), \end{aligned} \tag{16}$$

where $|x-y|$ is the "severity" of a specific error, that of predicting x where the true value is y. $ES(X,Y) = \sum_{x=1}^{M}\sum_{y=1}^{M} P(x,y)\log(1+|x-y|)$ measures weighted [by the joint probability $P(x, y)$] errors between predictions the classifier has made and labels for the M true classes. Since ES refers to the "distance" measured on an ordinal scale between two classes, it will contribute to IM only for ordinal classification problems, where such a distance has a meaning, and will not contribute in non-ordinal problems (where only MI between predictions and true values will contribute to IM).

By taking the logarithm of the sum of the error severity $|x-y|$ and 1 (16), we put ES and MI on common ground, letting them span the same range and be additive. Let's consider those conditions/scenarios that establish the range of values IM gets. As Table 2

Table 2 Extreme conditions/scenarios for IM in an M-class classification problem

	$X = Y$ (i.e., $x = y \quad \forall y$) and a uniform class distribution ($1/M$ diagonal matrix entries)	$\lvert x - y\rvert = M - 1$ for $y = M$, $x = 1$ i.e., $(1, M)$ is the only non-empty matrix entry	Uniform confusion matrix distribution
ES	0	$\log(M)$	$\frac{M-1}{M^2}\log(2M!)$
$-MI$	$-\log(M)$	0	0
IM	$-\log(M)$	$\log(M)$	$\frac{M-1}{M^2}\log(2M!)$

demonstrates, when there is no difference between the true and predicated classes, i.e., perfect classification, $y = x$, ES takes its minimal value of $P(x, x)\log(1 + 0) = 0$, as desired. In this scenario, X and Y are identical and, thus, dependent, and MI will take its maximal value when the class variable is uniformly distributed, $MI(Y, Y) = \sum_{y=1}^{M}\sum_{y=1}^{M} P(y, y)\log\left(\frac{P(y,y)}{P(y)P(y)}\right) = \log(M)$, which is also the entropy of Y, $MI(Y, Y) = \min\{H(Y), H(Y)\} = H(Y)$ (Cover and Thomas 2012). This scenario sets the minimal (best) value of IM, which is $-\log(M)$ (Table 2). When, on the other hand, the severity is maximal, i.e., all samples are of true class $y = 1$ and classified as class $x = M$ (or vice versa), which means that $P(x = M, y = 1) = 1$ and $\lvert x - y\rvert = M - 1$, ES is $P(M, 1)\log(1 + M - 1) = \log(M)$. In this scenario, the only entry in the double sum of MI is $P(M, 1)\log\left(\frac{P(M,1)}{P(x=M)P(y=1)}\right) = \log(1) = 0$. Thus, $-MI(X, Y) + ES(X, Y) = \log(M)$ is the highest value IM takes.

A third interesting scenario in Table 2 is when the confusion matrix distribution is uniform, and then ES takes a middle value of $\frac{M-1}{M^2}\log(2M!)$[1].

In summary, not only that MI and ES are in the same range, but they are in opposite trends, which encouraged us to sum them, where MI is added in a negative sign, as we wish to minimize both $-MI$ and ES. As Table 2 shows, IM is in the range $[-\log(M), \log(M)]$, where $-\log(M)$ is for perfect classification (all samples are correctly classified) and the data is balanced across the classes, and $\log(M)$ is for the extreme misclassification case, when all samples belong to class 1, but are classified as class M (or vice versa). If we identify the error severity with an adaptive cost for penalizing different misclassification errors differently (Grossman and Domingos 2004; Elkan 2001), then the IM can be interpreted as a cost matrix (Table 3).

Now, let us prove that IM gets its minimum at the same point -MI and ES get their minimum. We base our proof on Lemma 1 that shows that a function (IM) that is the sum of two other functions (-MI and ES) that get their global minimum at the same point will get its global minimum at that point.

[1] For a uniform confusion matrix distribution, $P(x, y) = 1/M^2 \quad \forall x, y$, $MI(X, Y) = \sum_{x=1}^{M}\sum_{y=1}^{M} 1/M^2 \log\left(\frac{1/M^2}{1/M \cdot 1/M}\right) = 0$. We will separate the computation of ES to three elements: on, above, and below the diagonal of the confusion matrix. The sum on the diagonal is 0 (as there are no error terms on the diagonal) and that above the diagonal equals that below the diagonal (due to the symmetry of $\lvert x - y\rvert$). Thus, $ES(X, Y) = 2 \cdot 1/M^2 \cdot \sum_x \sum_{y>x} \log(1 + \lvert x - y\rvert) = 2/M^2 \cdot S_n$, where S_n is the sum over all matrix entries above the diagonal, which is also the arithmetic series for which the first element is $\log(2) + \log(3) + \cdots + \log(M)$, the last element is $\log(2)$, and the number of series elements is $M - 1$. That is, $ES(X, Y) = 2/M^2 \cdot (M - 1)/2 \cdot (\log(2) + \log(3) + \cdots + \log(M) + \log(2)) = (M - 1)/M^2 \cdot \log(2M!)$.

Table 3 Cost matrix of IM

Predicted class (X)	True class (Y)			
	Class 1	Class 2	...	Class M
Class 1	log(1)	log(2)		log(M)
Class 2	log(2)	log(1)		log($M-1$)
...			...	
Class M	log(M)	log($M-1$)		log(1)

Lemma 1 *A function that is the sum of two functions that get their global minima at the same point will also get its global minimum at that point.*

Proof Let $x, y \in A$, and let $\arg\min_{x\in A} f(x) = x^*$, a global minimum of f, and $\arg\min_{x\in A} g(x) = x^*$, also a global minimum of g. Let us assume by contradiction that $\arg\min_{x\in A} h(x) = g(x) + f(x) = y$, $y \neq x^*$. It follows that $f(y) > f(x^*)$ and also $g(y) > g(x^*)$, which means that $g(y) + f(y) > g(x^*) + f(x^*)$, but we assumed that y is a global minimum of $h(x) = g(x) + f(x)$ for all $x \in A$, which makes the contradiction. □

As we have seen, IM is a proper measure to tackle ordinal classification problems, and it answers the requirements of combining information and error severity to classification accuracy, and of handling class imbalance (see Sect. 5 for empirical evaluation). But, it may poorly evaluate the classifier in cases where the classifier has poor performance (e.g., there are more errors than correct classifications), and in these cases, MI dominants IM. It is easy to propose a corresponding theoretical confusion matrix (Sect. 5.5 and Fig. 6), but it can also happen in practice, for example, when the algorithm starts its greedy search with a classifier that is close to random. Therefore, to trade better IM and accuracy, we modify IM with a term $\alpha \geq 1$ that adjusts the error severity (see "Information measure with alpha" section in Appendix):

$$IM_\alpha = \sum_x \sum_y -P(x,y) \log\left(\frac{\alpha P(x,y)}{P(x)P(y)}\right) + \sum_x \sum_{y, x\neq y} P(x,y) \log\left(\alpha(1+|x-y|)\right). \quad (17)$$

Then IM_α (i.e., IM that is controlled by α) can be written as (see "Information measure with alpha" section in Appendix):

$$IM_\alpha = IM - \log(\alpha) ACC, \quad (18)$$

where α's role in practice is to determine the balance between ACC and IM (and not to add costs to error severities). The measure range is $-\log(\alpha M) < IM_\alpha < \log(M)$. The minimal value $-\log(\alpha M)$ is achieved for perfect classification, when all samples are correctly classified and the data is balanced. In this case, $IM = -\log(M)$, and because ACC is 1, $IM_\alpha = -\log(M) - \log(\alpha) = -\log(\alpha M)$. The maximal value $\log(M)$ is the extreme misclassification case, when all samples belong to class 1, but are classified as M (in this case, $ACC = 0$, so the second element in Eq. (18), $-\log(\alpha)ACC$, cancels out).

Note that when $\alpha = 1$, IM is a special case of IM_α. As α increases, IM_α decreases regardless of IM, which is independent of α and becomes negligible compared to $\log(\alpha)ACC$. Then, as the following Lemma shows, ACC becomes a special case of IM_α.

Table 4 Example for alpha analysis with three classes

Predicted class (X)	True class (Y)		
	C_1	C_2	C_3
C_1	T	F	F
C_2	F	T	F
C_3	F	F	T

Lemma 2 *As α increases, IM_α is monotone with ACC.*

Proof Let A_i and A_j be two classifiers for a number of classes $M > 2$, and let $\alpha \gg M$. For A_i

$$IM_\alpha(A_i) = IM(A_i) - \log(\alpha)ACC_i.$$

Without loss of generality, we assume that $ACC_i > ACC_j > 0$. Since $\alpha \gg M$, and since IM is upper bounded by $\log(M)$, IM is negligible to the second element, so

$$IM_\alpha(A_i) = -\log(\alpha)ACC_i \quad and \quad IM_\alpha(A_j) = -\log(\alpha)ACC_j,$$

which means that:

$$IM_\alpha(A_i) < IM_\alpha(A_j).$$

□

That is, for $\alpha \gg M$, IM_α is monotone with ACC, and thus learning a BNC structure by minimizing IM_α yields a BNC that also maximizes ACC, and the structure minimizing IM_α is the same structure maximizing ACC. That is, ACC is a special case of IM_α for large α (but only for large α). Therefore, IM_α balances between IM and ACC and provides extra sensitivity beyond that provided by IM to different tradeoffs between accuracy and information, error distributions, and error severities.

To demonstrate the impact of α on IM_α, we use a simple example. Let U be a matrix of dimension $M = 3$, where all off-diagonal and main diagonal elements are F (false) and T (true), respectively, and let $F = \frac{1}{3}T$ (as in Sect. 3.3, F is the number of misclassifications of class i to class j, $\forall j \neq i$, and T is the number of correct classifications of class i, $\forall i$) (Table 4). We executed 81 ($M^4 = 3^4$) scenarios and calculated for each scenario ACC, IM, and IM_α, the latter with a range of α values in [1,81]. Figure 1 shows that as α increases, IM_α increases as $\log(\alpha)$ACC, and ACC and IM are, as expected, independent of α. For $\alpha = 1$, $IM = IM_\alpha \approx \frac{2}{3}$ACC, and for $\alpha = 81$, $IM_\alpha \approx 90\%$ of ACC. An interesting intermediate point is $\alpha = M^2 = 9$. Up until $\alpha = 9$, the IM_α gains more than 80% of its maximum value (ACC). But, due to the logarithm function, for $\alpha > 9$, the increase rate is low, and for example for $\alpha = 81$, IM_α gains only a bit more than 90% (even for $\alpha = 100,000$, it only gains a little bit more than 95% of ACC).

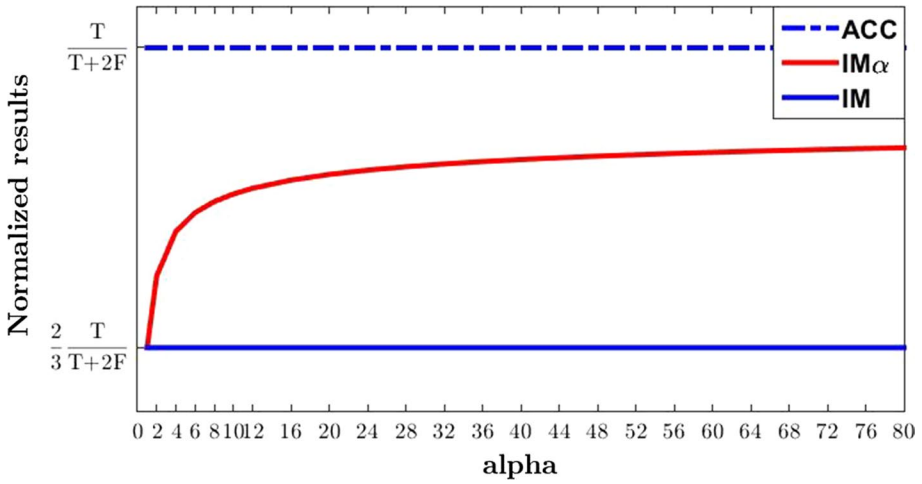

Fig. 1 α analysis for class variable with three classes (Color figure online)

5 Measure evaluation using synthesized confusion matrices

Our first examination of the proposed measures was in six experiments using synthesized confusion matrices that exhibit different scenarios. The advantage in using synthesized confusion matrices is in dispensing with training and testing the classifiers. Since values of different measures are in different ranges, to be able to present all measures on the same graph, we normalize each measure to [0-1] by:

$$Measure_{Norm} = \frac{Measure - min(Measure)}{max(Measure) - min(Measure)}. \tag{19}$$

Note that some of the performance measures (e.g., ACC and MCC) should be maximized and some (i.e., CEN and IM) should be minimized.

5.1 Sensitivity to class imbalance

In this experiment, 101 confusion matrices for two classes were created: each for 100 samples and perfect classification (Table 5). The only difference among the matrices is in the number of samples coming from each class, which is measured by m (which control the balance). For $m = 0$, the confusion matrix is highly imbalanced (i.e., all samples belong to class 1). As m increases, the confusion matrices become more balanced, and for $m = 50$, the classes are perfectly balanced. As m increases from 50 towards 100, the confusion matrices become imbalanced again (i.e., for $m = 100$, all samples belong to class 2). Figure 2 presents the experiment results for nine measures and settings: IM, IM_α ($\alpha = 10$), IM_α ($\alpha = 100$), IM_α ($\alpha = 1000$), MI, CEN, MCC, MAE, and ACC. In Fig. 2 (and also in Figs. 3, 4, 5, 6), measures that behave the same share the same symbol and graph color.

Figure 2 shows that while IM, IM_α, and MI are sensitive to the level of balance and peak to a balanced distribution ($m = 50$), CEN, MCC, MAE, and ACC are indifferent to the level of balance. The latter four measures receive a perfect score for all scenarios since

Table 5 Sensitivity to class imbalance

Predicted class (X)	True class (Y)	
	C_1	C_2
C_1	$100 - m$	0
C_2	0	m
$m = [0, 100]$		

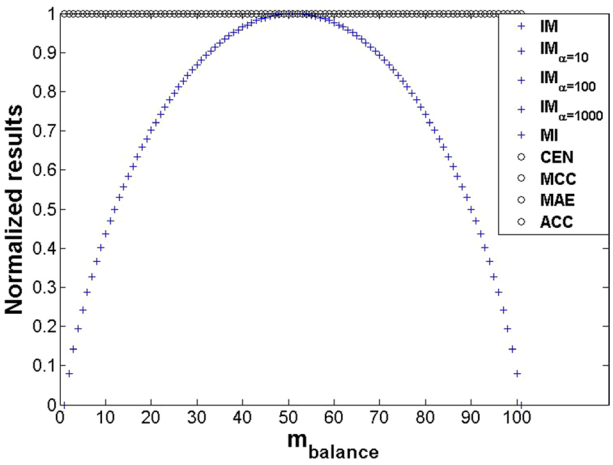

Fig. 2 Sensitivity to class imbalance (Color figure online)

ACC remains 1 throughout the experiment, the correlation remains perfect (MCC), and there are no errors distributed (CEN and MAE). Because the experiment was conducted without misclassification errors, $\text{IM} = \text{IM}_\alpha = \text{MI}$. In real problems, a classifier errs and the classes are almost always imbalanced. Thus, when a classifier is trained by CEN, MCC, MAE, or ACC, it will be fooled by the majority class, misclassifying all/most samples of the minority classes, whereas a classifier that is trained by IM, IM_α, or MI is expected to err evenly for all classes.

5.2 Sensitivity to the number of classes

In this experiment, 99 confusion matrices were created, each with a different number of classes ranging from 2 to 100 (i.e., when $M = 2$, the confusion matrix is a matrix of dimension 2, and when $M = 100$, the matrix is of dimension 100). As in the previous experiment, all matrices demonstrate a perfect classifier, with diagonal entries equal to 10 (Table 6). Figure 3 shows that while IM, IM_α, and MI are sensitive to the number of classes, CEN, MCC, MAE, and ACC are not. Although the four latter measures show perfect performance, it is only because there are no errors in this scenario. In real problems, a classifier tends to err more as the number of classes in the classification problem increases. While CEN, MCC, MAE, and ACC show no sensitivity to this number, IM, IM_α, and MI do show such sensitivity.

Table 6 Sensitivity to the number of classes

Predicted class (X)	True class (Y)			
	C_1	C_2	...	C_M
C_1	10	0		0
C_2	0	10		0
...			...	
C_M	0	0		10
$M = [2, 100]$				

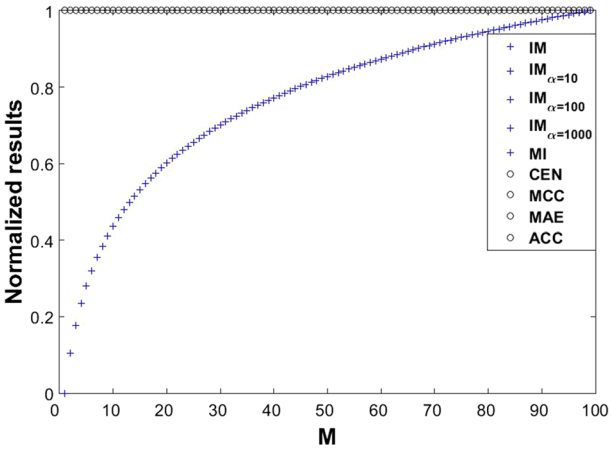

Fig. 3 Sensitivity to the number of classes (Color figure online)

5.3 Sensitivity to the error severity

In this experiment, 99 confusion matrices with 100 classes were created, each representing the worst classification scenario (all samples are of Class 1 and misclassified), but with a different error severity. That is, the number of misclassifications in each matrix is fixed, but the error severity (i.e., $|x - y|$) changes from the mildest (all Class 1's samples are misclassified to Class 2) to the harshest (all Class 1's samples are misclassified to Class 100). This severity is represented in the matrices by the parameter S, which changes in [1, 99] according to the position (severity) of the error in the confusion matrix (Table 7) (note that in each matrix, only one cell is non-zero holding the entire error "E"). Figure 4 reveals that only IM, IM$_\alpha$, MI, and MAE are sensitive to the error severity, losing accuracy with the increase of the severity, as is expected from a performance measure. CEN obtains a perfect score for all error severities, and MCC and ACC are the worst in performance (always 0), but all three measures are insensitive to the error severity regardless of their result, which manifests an additional shortcoming of them as performance measures.

5.4 Sensitivity to the error distribution

In this experiment, 34 confusion matrices represent scenarios of wrongly classifying 99 samples of Class 4 (of four classes) with different error distributions. This distribution is controlled by m (Table 8). As m increases, the distribution becomes more uniform

Table 7 Sensitivity to the error severity

Predicted class (X)	True class (Y)			
	C_1	C_2	...	C_{100}
C_1	0	0		0
C_2	$\begin{cases} E, & S=1 \\ 0, & else \end{cases}$	0		0
...			...	0
C_{100}	$\begin{cases} E, & S=99 \\ 0, & else \end{cases}$	0	0	0
$S=[1,99]$				

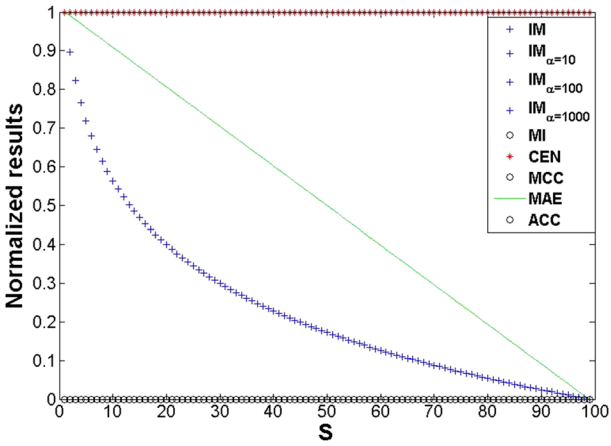

Fig. 4 Sensitivity to the error severity (Color figure online)

and vice versa. Note that the total error severity is equal in all scenarios/matrices (i.e., $\sum |x-y| = 198 \quad \forall Matrix$). Figure 5 shows that MI, MCC, MAE, and ACC are not sensitive to the error distribution, whereas the other measures are. However, CEN decreases as m increases because the measure "prefers" the error distribution not to be uniform, whereas IM and IM_α increase linearly with m because they excel for uniform error distribution.

5.5 ACC–information tradeoff

This experiment demonstrates with a simple example the tradeoff between ACC and information (as we expect will be measured by IM). Let $U1$ and $U2$ be two confusion matrices for two classifiers for $M = 3$. In Case 1 (Table 9), $U1$ has an ACC of 80% compared to a slightly lower accuracy of 79% for $U2$, but it can easily been seen that $U2$ reveals more information about the classification than $U1$, which has information only concerning Class 1's predictions. Quantitatively, $U1$'s MI is 0 compared to $U2$'s MI, which is 0.31. In Case 2 (Table 10), the ACCs of $U1$ and $U2$ are equal, but $U2$'s MI is higher than $U1$'s (0.32 compared to 0). RMCV (which is learned using ACC), for instance, would not show any difference between the two classifications. However, in both cases, although $U1$ and $U2$ are similar (Case 1) or identical (Case 2) with respect to accuracy, they provide different

Table 8 Sensitivity to the error distribution

Predicted class (X)	True class (Y)			
	C_1	C_2	C_3	C_4
C_1	0	0	0	m
C_2	0	0	0	$99-2m$
C_3	0	0	0	m
C_4	0	0	0	0
$m = [0, 33]$				

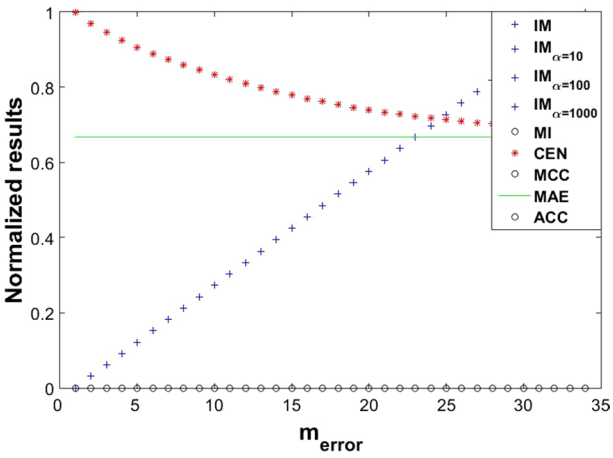

Fig. 5 Sensitivity to the error distribution (Color figure online)

degrees of information about the problem. This is reflected in different IM values, where that of $U2$ is higher than that of $U1$ in both cases.

To demonstrate this example in the general case, we created (Table 11) 51 confusion matrices for 100 samples equally distributed between two classes but with different types of errors. The type of error is determined by the value of m, which is the number of Class 1's samples that are wrongly classified as Class 2 (and the number of Class 2's samples that are wrongly classified as Class 1), whereas $50 - m$ is the number of Class 1's (2's) samples that are correctly classified. Figure 6 presents the experimental results for the same measures, but in this case, we used IM_α ($\alpha = 3$), IM_α ($\alpha = 10$), and IM_α ($\alpha = 100$) to see the differences among the measures more clearly. For $m = 0$, ACC, MCC, and MAE are 1, and they linearly decrease with m until 0 for $m = 50$. Note, however, that as m increases (and the accuracy deteriorates), the information shared by the classifier increases (Table 11).

As Fig. 6 shows, MI decreases with m as the confusion matrix becomes more uniformly distributed until a uniform distribution at $m = 25$ (for which MI = 0). For m greater than 25, MI increases at the same rate of the decrease until $m = 25$ because MI does not distinguish between correct and wrong classifications. Table 12 demonstrates two mirror cases—the first shows perfect classification and the second shows perfect misclassification—but both have the same MI value. This is the main disadvantage of MI that it does not distinguish symmetrical cases, and a high MI value can equally imply a very good or a very bad classifier.

In addition, Fig. 6 shows that IM decreases with m up to a certain point ($m = 35$) and from that point starts to increase due to an enhanced contribution of MI to IM. This

Table 9 Case 1 for demonstrating ACC and information tradeoff

Predicted class (X)	True class (Y)		
	C_1	C_2	C_3
(a) U1			
C_1	80	20	0
C_2	0	0	0
C_3	0	0	0
(b) U2			
C_1	59	0	0
C_2	21	20	0
C_3	0	0	0

Table 10 Case 2 for demonstrating ACC and information tradeoff

Predicted class (X)	True class (Y)		
	C_1	C_2	C_3
(a) U1			
C_1	80	20	0
C_2	0	0	0
C_3	0	0	0
(b) U2			
C_1	60	0	0
C_2	20	20	0
C_3	0	0	0

contribution led MI to start its increase at 25, sooner than IM. This seems to be the greatest disadvantage of IM, that following a severe decline in the classification performance, MI becomes more dominant and worsens IM. Models that classify with $m > 35$ are not superior to the model with $m = 35$ regarding classification although their IM improves. CEN seems to be more appropriate in such a scenario since it starts its incline only at 40, and this incline is very moderate compared with IM. Until 35, IM_α (for all values of alphas) behaves, as expected, between ACC and IM. It decreases from $m = 0$ to $m = 35$ at a higher rate than ACC, which is similar to that of IM. However, it does not increase as IM beyond $m = 35$ due to the increased impact of α on the classification errors. By that, IM_α overcomes the above disadvantage of IM. The value of alpha determines the type of behavior. When α is small, IM_α behaves similarly to IM, and when α is large, IM_α behaves similarly to ACC.

In concluding Sect. 5, Table 13 summarizes how the seven evaluated measures meet requirements we may have from a classification-oriented measure used for learning a BNC. We use a green check-mark to indicate that a specific measure meets a certain property (requirement), a red X-mark to indicate that it does not meet the property, and a combined black mark to indicate that the measure meets the property, but only under certain conditions/constraints. The table shows that IM and IM_α are the only measures that meet all requirements. Full details and proofs are in "Sensitivity analysis" section of Appendix.

Table 11 ACC—information tradeoff

Predicted class (X)	True class (Y)	
	C_1	C_2
C_1	$50 - m$	m
C_2	m	$50 - m$
$m = [0, 50]$		

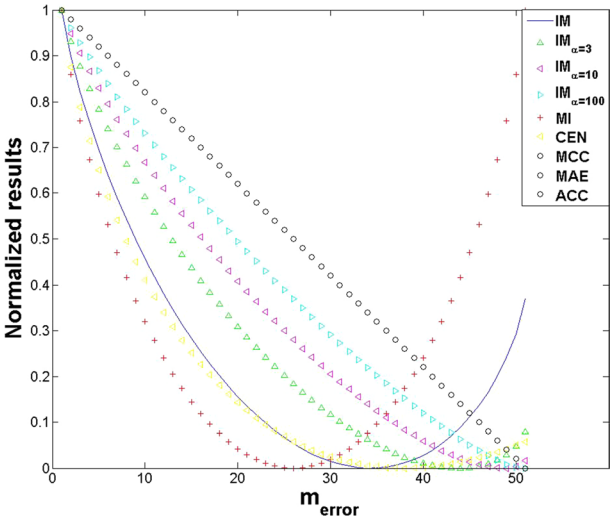

Fig. 6 ACC—information tradeoff (Color figure online)

6 Experiments and results

In this section, we empirically evaluate BNCs learned using the seven measures that were described in Sects. 3 and 4: IM, IM_α, MI, CEN, MCC, MAE, and the zero-one loss function (i.e., ACC). First, we create seven structure learning algorithms based on the RMCV algorithm (although we could base on other classifiers). For each measure, in each learning iteration of this search and score (S&S) algorithm, all neighboring BNCs (derived from the current BNC by an edge addition, deletion, or reversal) are compared to the current BNC (after learning the graph parameters) based on the measure and the BNC confusion matrix, and learning proceeds as long as more accurate graphs are found in consecutive iterations.

That is, we suggest seven variants of the RMCV algorithm for which learning is performed according to a different measure:

- Learning BNC according to IM
- Learning BNC according to IM_α
- Learning BNC according to MI
- Learning BNC according to CEN
- Learning BNC according to MCC
- Learning BNC according to MAE
- Learning BNC according to RMCV (ACC)

Table 12 (a) Perfect classification and (b) completely wrong classification that share the same MI value

Predicted class (X)	True class (Y)	
	C_1	C_2
(a)		
C_1	10	0
C_2	0	10
(b)		
C_1	0	10
C_2	10	0

Table 13 Summary of properties (columns) we expect from different measures (rows) used in learning a BNC (see also the above experiments with artificial confusion matrices and "Sensitivity analysis" section of Appendix) (Color table online)

	Sensitivity to class imbalance	Sensitivity to the number of classes	Sensitivity to error distribution	Tackle error severity	Balance accuracy and information
ACC	×	✓	×	×	×
MAE	✓	✓	✓	✓	×
MCC	✓	✓	✓	✓	✓
CEN	×	×	✓	×	✓
MI	✓	✓	✓	✓	✓
IM	✓	✓	✓	✓	✓
IM_α	✓	✓	✓	✓	✓

Each variant leads to its own classifier with its own confusion matrix. A confusion matrix of each of the seven variants is evaluated according to seven measures: IM, IM_α, MI, CEN, MCC, MAE, and ACC. That is, learning a BNC by each variant is made according to its own measure, but evaluation in the test is made according to all measures. In other words, each of the measures evaluates the confusion matrix derived by each of the trained BNC variants using the test set. Note that since IM_α decreases with α, we compare performances of classifiers trained with different α values and select the best IM_α-based variant (classifier) using the IM measure, which is independent of α.

The BNC based on IM_α is designed as a wrapper algorithm (Algorithm 1), which repeats the learning phase with different αs selected from the range $[2, M^3]$, as recommended in Sect. 4. In order to avoid an exhaustive search and due to the *log* behavior of alphas, we only search for alphas between 2 and M ($\alpha = 1$ is exactly IM), $\frac{M+M^2}{2}$, M^2, $\frac{M^2+M^3}{2}$, and M^3. The wrapper chooses the alpha that maximizes the IM measure (Sect. 4). Note that there is no use in the testing set in this phase. The wrapper algorithm's input is similar to that of RMCV and consists of: a training set (D_{tr}), test set (D_{tst}), number of classes (M), number of folds for the RMCV's cross-validation (K), and an initial graph (G_0). First, the α value that maximizes IM is found together with the corresponding BNC's structure. Then, after learning the parameters for this structure to turn it into a classifier, this classifier is tested using the test set to provide a confusion matrix that is evaluated as those yielded by the other measures.

Input: $D_{tr}, D_{tst}, M, K, G_0$
Output: $G^*, \alpha^*, ConfusionMatrix$
$CurrScore = 0$;
for $i \in [2 : M, \frac{M+M^2}{2}, M^2, \frac{M^2+M^3}{2}, M^3]$ **do**
  $(G, IM) = \text{Run\_RMCV\_IM}_\alpha(D_{tr}, G_0, K, i)$;
  if $IM > CurrScore$ **then**
    $CurrScore = IM$;
    $\alpha^* = i$;
    $G^* = G$;
  end
end
Compute $\theta = \text{Learn\_Parameters}(D_{tr}, G^*)$;
Compute $ConfusionMatrix = \text{Test\_Classifier}\ (D_{tst}, G^*, \theta)$;

Algorithm 1: The IM_α wrapper algorithm.

Since the RMCV algorithm must be initialized by a graph, when in the following experiments we evaluate each of the seven algorithms, we do that with both the empty graph and the naïve Bayesian classifier (NBC) as initializations. In total, for each database, we train 14 classifiers (for seven measures X two initializations).

This section is divided into three experiments. In Sect. 6.1, we compare the seven algorithms using 23 (artificial) synthetic datasets. In Sect. 6.2, we compare the seven algorithms using 17 real world and UCI datasets. While in these two sections we evaluate the BNC learned using each of the seven measures, in Sect. 6.3, we compare the BNC learned using IM_α with state-of-the-art machine learning classification algorithms, such as neural network (NN), decision tree (DT), random forest (RF), and support vector machine (SVM).

In each experiment, we evaluate the results using the Friedman non parametric test that was designed for comparing multiple algorithms/classifiers over multiple databases. A Friedman test can be applied to classification accuracies, error ratios, or any other measure (Demšar 2006). Since the Friedman test only tells us if one algorithm is superior to the others, but not which algorithm is the most accurate, Demšar (2006) suggested that the test be followed by a post hoc test, the Nemenyi test or the Wilcoxon signed ranks test. The Nemenyi test compares all algorithms to each other regarding the ranks computed in the Friedman test in order to find which algorithm is superior to the others. The Wilcoxon signed ranks test, in contrast to the Nemenyi test, does not use the Friedman ranks, but rather computes the difference between two algorithms for each dataset and assigns ranks according to the absolute difference (i.e., the Wilcoxon test ranks differences between algorithms and not algorithms directly).

6.1 Artificial datasets

This experiment included 23 artificial (synthetic) databases (Table 14) that were derived from the synthetic BN structure in Fig. 7. The baseline BN consists of 20 variables (nodes) where the target variable is Node 20. We made sure that, on the one hand, this BN would not be too complicated (dense), but on the other hand, it would possess all types of variable connections: diverging, serial, and converging (Ide and Cozman 2002). This BN also has the following properties:

Table 14 Characteristics of 23 artificial databases

Database number	# Classes	# Samples	Class balance
1	2	2000	Yes
2	3	2000	Yes
3	4	2000	Yes
4	5	2000	Yes
5	6	2000	Yes
6	7	2000	Yes
7	8	2000	Yes
8	9	2000	Yes
9	4	500	Yes
10	4	1000	Yes
11	4	1500	Yes
12	4	2000	Yes
13	4	2500	Yes
14	4	3000	Yes
15–23	4	2000	Different degrees of imbalance

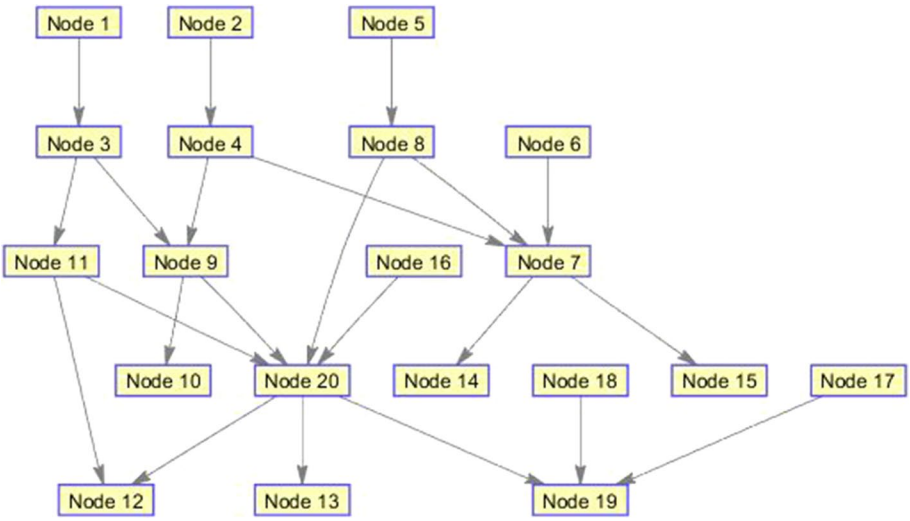

Fig. 7 Synthetic BN to create the artificial databases of Table 14

– Each variable has a cardinality of three.
– The target variable is fully balanced (each class has the same prior probability), unless otherwise mentioned.

We then derived from the baseline BN, 22 other BNs to perform a sensitivity analysis for: target variable cardinality (number of classes), sample size, and class balance (Table 14):

- Target variable cardinality: eight databases (Databases 1–8) containing 2, 3, 4,...,9 classes of the target variable (Node 20).
- Sample size: six databases (Databases 9–14) containing: 500, 1000, 1500, 2000, 2500, and 3000 samples.
- Class balance: nine databases (Databases 15–23) containing different balances for the target variable. Database 15 is perfectly balanced[2] and further databases gradually become less balanced. The percentage of samples each class holds was set heuristically.

Further details about the sampling technique for these databases are in "Artificial BN sampling" section of Appendix. Note that since in most evaluations (see below), we tested and report results for three separate category databases for which a single parameter is tested: the number of classes, number of samples, and degree of class imbalance, we included a benchmark database with four classes, 2000 samples, and no imbalance in the three categories (i.e., Databases 3, 12, and 15).

The BN of Fig. 7 was sampled ten times in each setting of the 23 of Table 14 to create ten data permutations for each of the 23 databases. Each permutation is divided into five equally sized datasets (folds) as part of a CV5 experiment, where each fold in its turn is used for the test and the other four folds are used for training. That is, each of the 23 databases in Table 14 is used and tested 50 times using different training and tests sets, and thus 1150 experiments using 1150 datasets are performed in total. Each of the seven algorithms trains and tests two classifiers (one for each initial graph) on each of the 50 datasets of the 23 databases (i.e., 16,100 classifiers). For each database, algorithm, and initial graph, we calculate all scores (i.e., IM, IM_α, MI, CEN, MCC, MAE, and ACC) as averages over the 50 confusion matrices of the 50 corresponding test sets.

Tables 15 and 16 show the average accuracies (ACC) and IM_α scores, respectively, achieved by the seven learning algorithms initialized by the empty graph. In each row of the two tables, the best classifier is marked in bold font, whereas the worst is marked in italic font. The last row in each table presents the average and standard deviation of the algorithms over all databases. A similar table for IM scores is Table 42 in "IM scores for artificial databases" section of Appendix.

Table 15 reveals that the IM_α-based BNC (where α has been optimized according to Algorithm 1) achieves the highest average accuracies although the BNCs were learned with the goal of maximizing IM_α and not ACC. This is because IM contains ACC components in both MI and ES terms which makes it maximize ACC while maximizing the IM_α score. Another explanation is that IM_α trades between IM and ACC; hence, for databases where maximizing accuracy leads to better performance, a large α is automatically chosen by the algorithm, and for databases where maximizing IM leads to better performance, a small α is selected. The tuning of α is done on training and validation sets, while the results shown are for an independent testing set. IM- and ACC- (RMCV) based BNCs (first and last columns of Table 15) do not seem to show any superiority over one another. Again, one would expect an ACC-based BNC to achieve better accuracy results, but the IM-based BNC does not fall behind.

[2] Note that three of the 23 databases, Databases 3, 12, and 15 have the same parameters: numbers of classes (4) and samples (2000), and no imbalance. Note, however, that each of these databases is randomly generated from the BN, and although they bring some bias when considered among the 23 databases, each is used in the sensitivity analysis of the algorithm to check a different characteristic: variable cardinality, learning curve, and degree of imbalance.

Table 15 Mean (std) ACC values of BNCs learned using seven measures and the RMCV algorithm that is initialized by the empty graph for 23 artificial databases

DB	IM	IM_α	MI	CEN	MCC	MAE	ACC
1	**93.30 (2)**	**93.30 (2)**	**93.30 (2)**	*93.28 (2)*	**93.30 (2)**	**93.30 (2)**	**93.30 (2)**
2	**90.65 (1)**	90.61 (1)	90.47 (2)	*63.84 (24)*	90.56 (1)	90.58 (1)	90.51 (1)
3	78.64 (3)	**79.57 (3)**	76.95 (5)	*44.36 (16)*	78.57 (3)	79.34 (3)	79.04 (3)
4	71.51 (3)	71.75 (3)	71.16 (3)	*36.32 (16)*	71.80 (2)	69.88 (7)	**71.95 (2)**
5	63.60 (8)	**66.25 (3)**	62.37 (8)	*26.63 (10)*	65.60 (4)	63.80 (7)	65.88 (3)
6	60.06 (3)	**60.56 (3)**	59.50 (3)	*23.98 (8)*	60.40 (3)	58.28 (5)	60.12 (3)
7	56.39 (3)	**57.34 (3)**	54.46 (6)	*19.92 (5)*	56.34 (4)	52.54 (10)	56.45 (4)
8	52.90 (3)	**53.84 (2)**	48.56 (9)	*16.91 (6)*	52.66 (3)	43.05 (14)	52.87 (2)
9	70.70 (6)	**71.70 (5)**	67.70 (9)	*41.92 (12)*	69.39 (8)	70.37 (6)	70.53 (7)
10	75.02 (7)	77.38 (2)	73.29 (8)	*43.98 (10)*	75.56 (7)	75.86 (5)	**77.62 (3)**
11	76.03 (5)	**77.82 (3)**	76.09 (5)	*40.16 (14)*	77.22 (4)	77.00 (3)	77.20 (3)
12	78.64 (3)	**79.57 (3)**	76.95 (5)	*44.36 (16)*	78.57 (3)	79.34 (3)	79.04 (3)
13	79.51 (2)	**80.17 (2)**	79.94 (2)	*42.33 (12)*	79.83 (2)	79.88 (2)	79.84 (2)
14	81.29 (2)	**81.34 (2)**	80.93 (2)	*41.89 (14)*	81.01 (2)	81.07 (2)	80.94 (2)
15	79.20 (2)	**79.31 (2)**	75.48 (9)	*38.72 (10)*	79.08 (2)	77.94 (6)	79.02 (2)
16	78.44 (2)	**78.74 (2)**	77.59 (4)	*40.64 (10)*	78.22 (2)	78.59 (2)	78.40 (3)
17	78.17 (3)	**78.42 (2)**	75.11 (7)	*38.77 (4)*	76.55 (7)	77.94 (3)	77.92 (3)
18	77.28 (6)	**78.47 (4)**	73.44 (11)	*45.46 (5)*	68.57 (12)	74.77 (8)	76.02 (8)
19	78.63 (4)	**79.49 (2)**	79.07 (2)	*48.46 (2)*	77.70 (6)	79.22 (2)	78.75 (4)
20	76.86 (6)	**78.34 (3)**	70.46 (9)	*56.88 (2)*	76.62 (5)	75.79 (6)	76.50 (5)
21	76.36 (5)	76.04 (5)	**77.31 (4)**	*69.17 (3)*	75.92 (4)	73.19 (5)	72.41 (5)
22	75.64 (4)	75.51 (4)	**77.14 (4)**	*73.20 (2)*	76.71 (4)	74.20 (3)	73.91 (3)
23	85.45 (2)	*85.39 (2)*	**85.72 (2)**	*85.39 (2)*	85.65 (2)	*85.39 (2)*	*85.39 (2)*
Avg (std)	75.40 (10)	**76.15 (9)**	74.04 (10)	*46.81 (19)*	75.04 (10)	74.41 (11)	75.37 (10)

Table 15 also reveals that CEN is the worst method to evaluate classifier performance, due to a major limitation of the measure. When all entries in a confusion matrix belong to one predicted class (which is the case when the initial graph is empty), the measure will take a very low value (CEN is a measure we wish to minimize). That is because, in Eq. (9), all CEN_m will result in zero except for one. This may also be seen in the following example. In Tables 17(a) and 17(b), we see the confusion matrices for Database 3 (Table 14) for an empty initial graph and for its best neighbor, respectively. The CEN scores are 0.3365 and 0.4138, respectively. Therefore, the algorithm terminates, and the empty graph is chosen by the CEN-based BNC even though it is obvious that the best neighbor which yields Table 17(b) is better in terms of accuracy and information.

Table 16 shows the average IM_α scores achieved by the seven algorithms. The IM_α has the highest average IM_α score. We recall that the IM_α is normalized; hence, its range is [0, 1]. For the purpose of visualization and to better distinguish between results, we multiply each score by 100. Thus, the following results of normalized IM_α are in the range [0, 100].

Figure 8 compares the classification performance (IM_α) between IM-, IM_α-, and ACC-based BNCs for an empty initial graph and the 23 databases. The first row refers to the comparison of the IM_α- and IM-based BNCs, the second row to IM_α- and ACC-based BNCs, and the third row to that between the IM- and ACC-based BNCs. The comparison

Table 16 Mean (std) IM_α (multiplied by 100) values of BNCs learned using seven measures and the RMCV algorithm that is initialized by the empty graph for 23 artificial databases

DB	IM	IM_α	MI	CEN	MCC	MAE	ACC
1	84.74 (3)	84.74 (3)	**84.76 (3)**	*84.73 (3)*	84.74 (3)	84.75 (3)	84.75 (3)
2	**84.15 (2)**	**84.15 (2)**	83.97 (2)	*57.79 (24)*	84.00 (2)	84.05 (2)	83.91 (2)
3	74.23 (3)	**75.10 (3)**	72.62 (5)	*46.17 (15)*	73.91 (3)	74.77 (3)	74.31 (3)
4	69.12 (3)	**69.41 (2)**	68.69 (2)	*41.40 (15)*	69.09 (2)	68.03 (5)	69.22 (2)
5	63.90 (3)	**65.84 (3)**	62.94 (3)	*32.14 (9)*	64.95 (3)	61.1 (5)	65.01 (3)
6	62.03 (2)	**62.43 (2)**	61.57 (3)	*33.24 (11)*	62.02 (2)	60.40 (4)	61.72 (3)
7	59.81 (2)	**60.49 (2)**	58.52 (4)	*32.68 (8)*	59.36 (3)	57.21 (6)	59.33 (3)
8	57.46 (3)	**58.27 (2)**	54.66 (6)	*28.57 (9)*	57.13 (2)	51.10 (10)	57.16 (2)
9	66.32 (5)	**67.14 (5)**	64.41 (7)	*44.35 (13)*	65.22 (6)	66.20 (6)	66.06 (6)
10	70.57 (5)	72.21 (2)	68.95 (6)	*46.69 (11)*	70.99 (5)	70.83 (4)	**72.29 (3)**
11	71.25 (4)	**72.70 (4)**	71.55 (4)	*42.68 (14)*	71.96 (4)	72.15 (3)	71.86 (4)
12	74.23 (3)	**75.10 (3)**	72.62 (5)	*46.17 (15)*	73.91 (3)	74.77 (3)	74.31 (3)
13	74.98 (2)	**75.59 (2)**	75.53 (2)	*44.45 (13)*	75.12 (3)	75.33 (2)	75.04 (3)
14	76.70 (2)	**76.74 (2)**	76.32 (2)	*43.78 (14)*	76.24 (2)	76.37 (2)	76.16 (2)
15	74.58 (3)	**74.68 (2)**	71.47 (7)	*38.46 (13)*	74.25 (3)	73.36 (5)	74.16 (3)
16	73.71 (2)	**73.86 (2)**	72.96 (4)	*40.84 (12)*	73.19 (3)	73.63 (2)	73.32 (3)
17	70.64 (3)	**70.88 (2)**	68.43 (5)	*34.70 (6)*	69.32 (6)	70.28 (3)	70.25 (3)
18	68.69 (5)	**69.59 (4)**	65.74 (9)	*36.83 (5)*	61.67 (9)	66.63 (7)	67.58 (7)
19	68.34 (3)	**68.92 (2)**	68.74 (2)	*38.09 (2)*	67.42 (5)	68.73 (3)	68.22 (3)
20	65.10 (5)	**66.44 (3)**	59.97 (8)	*41.62 (3)*	64.75 (4)	64.23 (5)	64.72 (5)
21	58.02 (7)	57.15 (8)	**59.56 (6)**	*47.77 (3)*	57.84 (6)	54.15 (7)	52.77 (8)
22	52.92 (6)	52.46 (6)	**55.24 (6)**	*48.82 (3)*	54.40 (6)	50.31 (5)	49.95 (4)
23	53.58 (2)	53.28 (1)	**54.49 (3)**	*53.18 (1)*	54.43 (3)	*53.18 (1)*	*53.18 (1)*
Avg (std)	68.48 (9)	**69.01** (9)	67.55 (8)	*43.70 (11)*	68.08 (8)	67.46 (10)	68.06 (9)

Table 17 Example for CEN limitation–confusion matrices for Database 3

Predicted class (X)	True class (Y)			
	C_1	C_2	C_3	C_4
(a) Initial empty graph				
C_1	0	0	0	0
C_2	0	0	0	0
C_3	365	379	406	350
C_4	0	0	0	0
(b) Best neighbor structure to the empty graph				
C_1	69	91	2	2
C_2	241	235	6	3
C_3	54	49	385	333
C_4	1	4	13	12

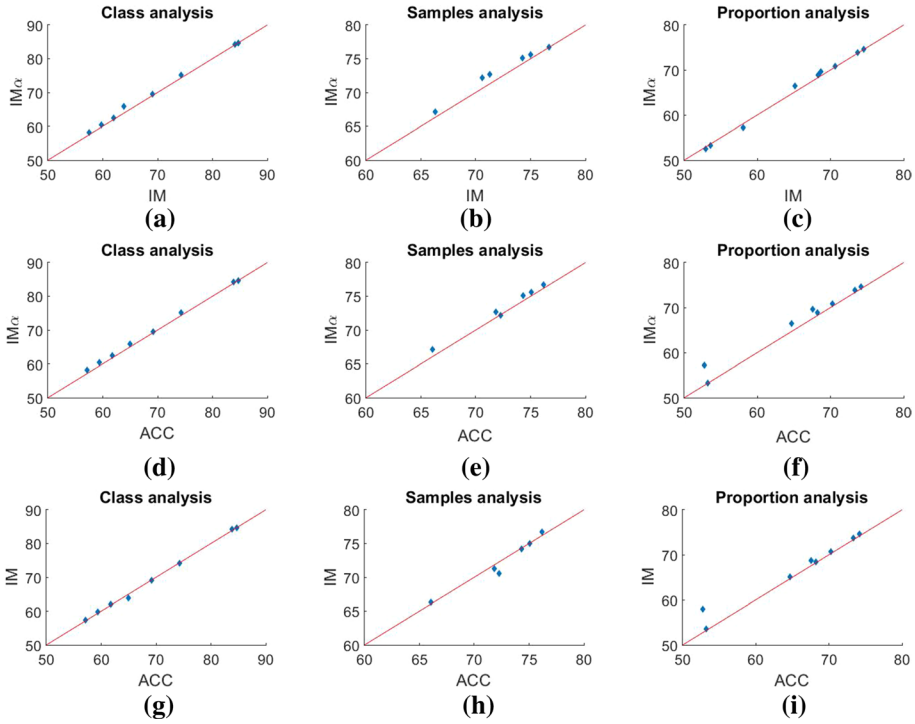

Fig. 8 IM_α scores of IM_α versus IM, IM_α versus ACC, and IM versus ACC for BNCs initialized by an empty graph for 23 artificial databases

is given in Figures 8a, d, g for Databases 1–8 (see Table 14) that allow analysis of the influence of the number of classes, Figs. 8b, e, h for Databases 9–14 that allow analysis of the influence of the number of samples, and Figs. 8c, f, i for Databases 15–23 that allow analysis of the influence of class imbalance. We call the three categories according to this division of the databases: class analysis (left column in Fig. 8), samples analysis (middle column), and proportion analysis (right column). The points in Fig. 8 which are above the $x = y$ (red) line represent databases for which the algorithm written on the y-axes is favored over the one written on the x-axes and vice versa.

Figure 8 (first two rows) shows that the IM_α-based BNC is superior to the other two classifiers (learned to minimize IM and maximize ACC) with respect to IM_α score. The superiority is obvious in the samples analysis (Fig. 8b, e) and proportion analysis (Fig. 8c, f) scenarios. However, a closer look reveals that also in the class analysis scenario (Fig. 8a, d), none of the points (each represents a database) are below the red line, which means that the IM_α-based BNC is also superior to the other two classifiers regarding class analysis (Databases 1–8). The third row in Fig. 8, which presents the comparison of IM- and ACC-based BNCs, shows that the IM-based BNC is superior to the ACC-based BNC in the case of proportion analysis, but is slightly inferior for the case of samples analysis.

In addition, we examine in Fig. 9 how the performance measures for each category of the databases change with the number of classes, number of samples, and balance in the samples among the classes (data proportion). The first row (Fig. 9a–c) shows IM scores, while the second (Fig. 9d–f) shows ACC scores. As can be seen in Fig. 9, the IM score is

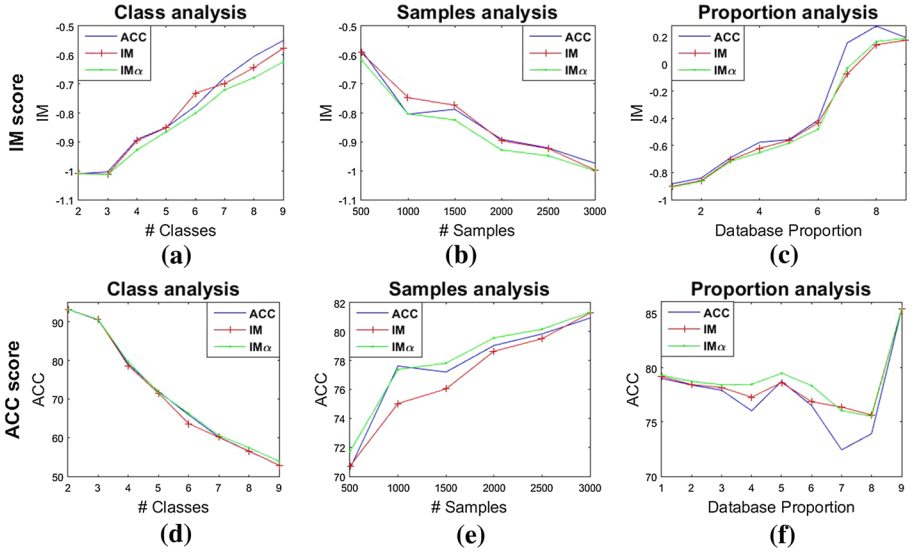

Fig. 9 ACC and IM measured for BNCs initialized by an empty graph and learned using the ACC-based (blue), IM-based (red), and IM_α-based (green) BNCs for 23 artificial databases (Color figure online)

monotone for each analysis (Fig. 9a–c), whereas ACC is not monotone in the case of the proportion analysis (Fig. 9f). As the number of classes increases, ACC decreases (Fig. 9d) because the classification task becomes more difficult, and IM increases (Fig. 9a) for the same reason. As the number of samples increases, ACC increases (Fig. 9e) and IM decreases (Fig. 9b) (i.e., both performance measures are improved). The reason is that as the number of samples in the dataset increases, the number of samples for each combination of variables increases, which makes the estimated probabilities more reliable and thereby also increases ACC. In the case of proportion analysis, ACC is unstable for all algorithms in contrast to the IM score, which increases as the database becomes imbalanced. This can be attributed to the accuracy limitation that was described in Sect. 5; the accuracy is not sensitive to changes in the level of imbalance. Finally, we see that in terms of IM (Fig. 9a–c) and ACC (Fig. 9d–f), the IM_α-based BNC is the best algorithm.

Figure 10 reveals that the number of neighbors of the IM- and ACC-based BNCs is similar where the initial graph is empty (Fig. 10a–c) with a slight tendency towards the ACC-based BNC (the ACC line is almost always beneath the IM line, which means fewer neighbors). However, for the NBC initial graph, the ACC-based BNC is significantly superior to the IM-based BNC. In the proportion analysis, for either the empty or NBC initial graphs (Fig. 10c, f), there is a sharp decline starting from Database 6. The explanation for this break point is that from the sixth database (i.e., Database 20), we created very imbalanced databases. Note that we excluded IM_α to keep the graph scale. Each iteration of the IM_α -based BNC includes examining several alphas; hence, the number of neighbors is a function of the number of alphas and the number of iterations, and IM_α is inferior to all seven proposed algorithms with respect to run time. The CEN-based BNC has the lowest number of iterations, which is constant regardless of the scenario and coheres with the explanation described above about CEN limitation and poor performance. More details about run-times are given in Table 43 in "Run time measured by number of neighbors for artificial BNs" section of Appendix.

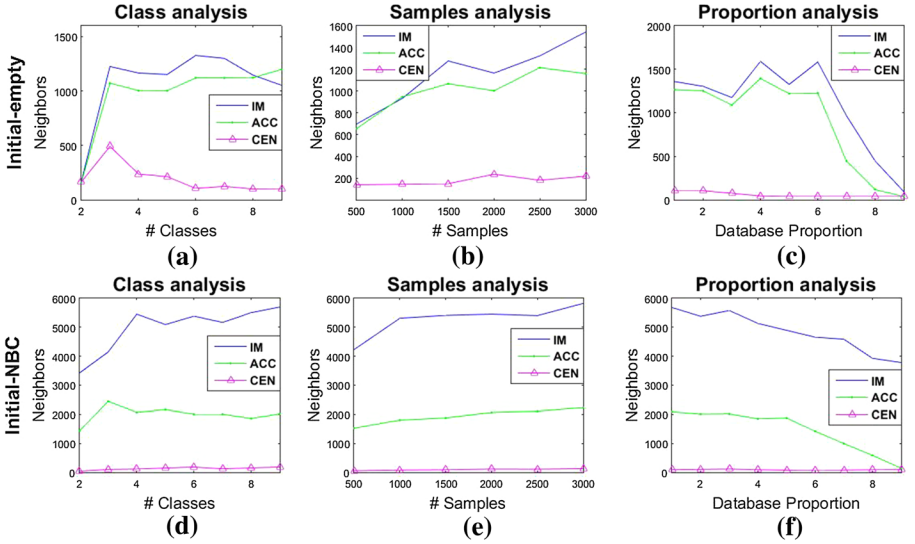

Fig. 10 Number of algorithm's neighbors for artificial databases (Color figure online)

Table 18 Mean (std) ACC values of BNCs learned using seven measures and the RMCV algorithm that is initialized by the empty graph for the non-major classes of nine imbalanced artificial databases

DB	IM	IM_α	MI	CEN	MCC	MAE	ACC
15	78.65 (1)	**78.90 (1)**	77.72 (1)	*13.22 (6)*	78.70 (1)	77.46 (1)	78.44 (1)
16	77.51 (1)	**78.06 (1)**	76.83 (2)	*29.97 (10)*	77.57 (1)	77.89 (1)	77.91 (1)
17	77.83 (1)	**77.96 (0)**	72.83 (3)	*17.38 (11)*	75.79 (1)	76.94 (1)	76.49 (1)
18	70.10 (2)	**72.31 (2)**	60.34 (6)	*2.41 (2)*	55.24 (7)	68.28 (3)	64.75 (5)
19	72.48 (1)	**72.91 (1)**	72.58 (1)	*0.00 (0)*	71.45 (2)	72.90 (1)	72.35 (1)
20	58.98 (3)	**61.69 (2)**	51.15 (5)	*0.00 (0)*	60.36 (3)	57.79 (3)	60.09 (3)
21	37.40 (2)	32.62 (3)	**41.37 (2)**	*0.00 (0)*	34.22 (4)	20.42 (5)	20.03 (8)
22	15.91 (4)	14.86 (4)	**25.30 (4)**	*0.00 (0)*	23.50 (4)	6.95 (3)	4.54 (2)
23	4.50 (1)	4.14 (1)	**4.57 (2)**	*0.00 (0)*	4.07 (2)	*0.00 (0)*	*0.00 (0)*
Avg (std)	54.82 (28)	**54.83** (29)	53.63 (26)	*7.00 (10)*	53.43 (27)	50.96 (32)	50.51 (33)

To demonstrate that the advantage of the IM_α-based BNC is not biased by the majority classes at the expense of the minority classes in imbalance problems, and that the measure is indeed advantageous to the minority classes, we repeated the experiment with only the non-major classes in Databases 15–23, each having a different degree of imbalance from zero (15) to large (23). Table 18 shows the average ACC value over the non-major classes after excluding the major class for each of the imbalanced databases. The table reveals that indeed the IM_α-based BNC outperforms all other algorithms for all databases except for the three most imbalanced (21–23) for which the MI-based BNC is superior (and the IM-based BNCs are usually second best). The latter result demonstrates that, for a very highly imbalanced dataset, the MI component in the IM_α measure is more important than the ES component (we already saw the MI's supreme sensitivity to class imbalance in Sect. 5.1).

Table 19 Mean (std $\times 10^{-1}$) MAE values of BNCs learned using seven measures and the RMCV algorithm that is initialized by the empty graph for 14 balanced artificial databases

DB	IM	IM_α	MI	CEN	MCC	MAE	ACC
1	**0.067 (0)**	**0.067 (0)**	**0.067 (0)**	**0.067 (0)**	**0.067 (0)**	**0.067 (0)**	**0.067 (0)**
2	**0.099 (0)**	**0.099 (0)**	0.102 (0)	*0.452 (4)*	0.101 (0)	0.100 (0)	0.101 (0)
3	0.241 (0)	**0.232 (0)**	0.268 (1)	*0.743 (4)*	0.246 (0)	0.234 (0)	0.242 (0)
4	0.359 (0)	**0.353 (0)**	0.365 (0)	*0.978 (4)*	0.358 (0)	0.357 (1)	0.359 (0)
5	0.514 (1)	**0.485 (1)**	0.535 (1)	*1.374 (5)*	0.491 (1)	0.495 (1)	0.509 (1)
6	0.625 (1)	**0.616 (0)**	0.636 (1)	*1.833 (7)*	0.626 (1)	0.665 (1)	0.633 (1)
7	0.761 (1)	**0.745 (1)**	0.802 (1)	*1.773 (5)*	0.783 (1)	0.832 (2)	0.785 (1)
8	0.901 (1)	**0.869 (1)**	1.012 (3)	*2.321 (8)*	0.906 (1)	0.902 (3)	1.107 (1)
9	0.356 (1)	**0.345 (1)**	0.390 (1)	*0.781 (3)*	0.377 (1)	0.356 (1)	0.360 (1)
10	0.293 (1)	**0.266 (0)**	0.318 (1)	*0.696 (3)*	0.285 (1)	0.275 (1)	0.276 (0)
11	0.279 (1)	**0.259 (0)**	0.278 (1)	*0.802 (3)*	0.271 (1)	0.266 (0)	0.271 (0)
12	0.241 (0)	**0.232 (0)**	0.268 (1)	*0.743 (4)*	0.246 (0)	0.234 (0)	0.242 (0)
13	0.231 (0)	**0.223 (0)**	0.224 (0)	*0.778 (3)*	0.229 (0)	0.226 (0)	0.230 (0)
14	**0.207 (0)**	**0.207 (0)**	0.212 (0)	*0.763 (3)*	0.212 (0)	0.210 (0)	0.213 (0)
Avg (std)	0.370 (2)	**0.357 (2)**	0.391 (3)	*1.007 (6)*	0.371 (2)	0.373 (3)	0.385 (3)

Table 20 Average Friedman's ranks according to ACC, IM, MAE, and MI of BNCs learned using seven measures and two initializations for 23 artificial databases

	Initial	IM	IM_α	MI	CEN	MCC	MAE	ACC
1. ACC score	Empty	3.2	**1.5**	4.7	*6.9*	3.9	3.8	3.8
	NBC	4.0	**1.8**	*6.4*	5.7	3.6	3.3	3.1
2. IM score	Empty	2.8	**1.5**	4.3	*6.9*	3.8	4.1	4.5
	NBC	3.0	**1.6**	*5.4*	5.1	5.3	2.8	4.7
3. MAE score	Empty	3.1	**1.6**	4.7	*6.7*	3.6	3.9	4.3
	NBC	3.1	**1.6**	*6.0*	5.3	4.8	2.8	4.3
4. MI score	Empty	2.9	**1.5**	4.0	*6.9*	3.9	4.2	4.6
	NBC	2.5	**1.8**	4.9	4.8	*5.7*	3.1	5.2

For comparison, for these three databases, the MAE and ACC-based BNCs, and actually all BNCs, are very poor, demonstrating the inability of all measures to adequately accommodate a very high class imbalance.

To demonstrate that the advantage of the IM_α-based BNC for ordinal problems is not due to class imbalance, and that the measure is indeed advantageous when errors have different severities, we computed the MAE for the seven algorithms on the 14 non-imbalanced databases (1–14) (see "Artificial BN sampling" section of Appendix for details on how we created the ordinal problems). Table 19 shows that the IM_α-based BNC achieves better MAE results than all algorithms regardless of the class-variable cardinality (Databases 1–8) and sample size (Databases 9–14). This superiority applies even to the MAE-based BNC that was trained to minimize MAE, whereas the IM_α-based BNC was trained to minimize IM_α. In these scenarios, the ES component of IM_α is the dominant one (which is supported by the superiority of the MAE-based BNC to the MI-based BNC).

Finally, we proceed to Friedman's non parametric test followed by the Nemenyi post hoc test, as was suggested by Demšar (2006) in order to find which algorithms are superior. The Friedman test results are given for ACC, IM, MAE, and MI in Table 20, rows 1–4 respectively, and those of the Nemenyi post hoc test (with a 0.05 confidence level) for ACC and IM in Table 21, rows 1 and 2, respectively. The rows in Tables 20 and 21 refer to specific measures and initial graphs (empty or NBC), while the columns represent the seven algorithms/classifiers. In Table 20, we present the MAE and MI scores, in addition to IM, since they compose it, which allows us to see if the advantage of the IM-based BNC over the other algorithms is due to either or both of the measures. In Table 21, each column represents a baseline algorithm to which the rest of the algorithms were compared in the Nemenyi post hoc test.

First, we can see that the IM_α-based BNC has the lowest (best) average rank regardless of the initial graph or the measure (Table 20). The fact that the IM_α-based BNC shows better results with respect to both MAE and MI (that both compose IM) demonstrates that it *simultaneously* minimizes error severity and maximizes the information provided by the classifier. Second, as can be seen from the Nemenyi post hoc tests (Table 21), all algorithms were significantly better than the CEN-based BNC almost always, and the IM_α-based BNC was almost always significantly superior to all other algorithms regardless of the initial graph. With respect to the differences between the IM- and IM_α-based BNCs, we expanded our evaluation and performed Wilcoxon tests between these BNCs for the two initializations (Empty and NBC) and two measures (IM and ACC) and found, based on all four tests, that IM_α is superior to IM (with a 0.05 confidence level).

Discussion of the artificial-dataset experiment

The goal of this experiment was to demonstrate using 23 artificial databases the sensitivity of the different measures to the issues that motivated the development of IM_α. This experiment shows that the IM_α-based BNC (and usually also the IM-based BNC) are superior to the ACC-based BNC. This is especially remarkable since IM and IM_α are not trained to maximize the classification accuracy as ACC does, yet they achieved better ACC results. This is explained by the fact that IM contains ACC components in both the MI and ES terms and because IM_α trades IM and ACC, enjoying the benefits from both. Moreover, the experiment shows that the IM_α-based BNC simultaneously minimizes the error severity and maximizes the amount of information in the classification as is revealed in the MAE and MI scores achieved by the algorithm that outperform those of the MAE and MI-based BNCs, respectively.

The IM_α superiority as reflected based on the ACC, IM, MI, and IM_α scores can also be seen through the confusion matrices, which give us insight into additional information. For example, we can see the resultant confusion matrix of the ACC-based BNC (Table 22a) compared to that of the IM_α-based BNC (Table 22b) over a specific test set of Database 22. The ACC-based BNC totally fails in classifying the minority class (C_4), whereas the IM_α-based BNC achieves a 50% accuracy on this minority class. Also, the confusion matrix of the IM_α-based BNC is superior to that of the ACC-based BNC in terms of MAE (0.22 vs. 0.27) and MI (0.34 vs. 0.22). These differences between the matrices of the two classifiers are typical also to the other sets in the other databases. However, this superiority comes at the expense of run time, which on average is six times higher for the IM_α-based BNC than for the IM- or ACC-based BNCs (as it examines this approximate number of alphas). Note that we did not run the wrapper in parallel with different alphas, which could reduce the average run time to that of the IM-based BNC.

Table 21 Nemenyi post hoc test according to ACC and IM of BNCs learned using seven measures and two initializations for 23 artificial databases. Values in the table stand for algorithms for which the column headline is superior

	Initial	IM	IM_α	MI	CEN	MCC	MAE	ACC
1. ACC score	Empty	CEN	MI, CEN, MCC, MAE, ACC	CEN	–	CEN	CEN	CEN
	NBC	MI,CEN	IM, MI, CEN, MCC	–	–	MI, CEN	MI, CEN	MI, CEN
2. IM score	Empty	CEN	MI, CEN, MCC, MAE, ACC	CEN	–	CEN	CEN	CEN
	NBC	MI, CEN, MCC	MI, CEN, MCC, ACC	–	–	–	MI, CEN, MCC	–

Table 22 Confusion matrices achieved by ACC and IM_α-based BNCs initialized with NBC for a single test set of Database 22

Predicted class (X)	True class (Y)			
	C_1	C_2	C_3	C_4
(a) ACC				
C_1	360	77	15	2
C_2	8	8	0	4
C_3	1	0	19	4
C_4	0	0	2	0
(b) IM_α				
C_1	359	75	7	2
C_2	9	10	0	0
C_3	1	0	27	3
C_4	0	0	2	5

The proportion analysis (class imbalance) has shown that the ACC measure is noisy compared to the IM measure, which can be explained by the experiments that were conducted in Sect. 5, which demonstrated the ACC limitations, among them the insensitivity to class balance.

In this experiment, the CEN-based BNC was the least accurate. The reason for its poor performance seems to be that it is not sensitive enough to changes (e.g., number of classes, class proportions); hence, it is terminated too quickly. This was demonstrated with an example and was supported by Fig. 10a–f where the CEN-based BNC had on average not more than 200 neighbors regardless of the scenario. Another shortcoming of the CEN-based BNC is that as the sample size increases (500–3000), its accuracy does not increase as is expected from a classifier.

6.2 UCI and real-world databases

This experiment included 17 ordinal databases (Table 23), 14 of them are UCI databases (Lichman 2013), while the other three are original: ALS[3], Missed due date,[4] and Motorcycle.[5] The selected problems show diversity with respect to the sample size, number of

[3] The amyotrophic lateral sclerosis (ALS) database (Gordon and Lerner 2019) consists of patients' static data (e.g., sex, age at onset of disease), temporal/longitudinal data (e.g., blood pressure, laboratory test results), and ALSFRS (class variable) values, which are documented at every clinic meeting. ALSFRS scores take five values (classes) between 0 and 4, where 0 is complete loss of function and 4 corresponds to normal ability, that are distributed 1%, 5.5%, 17.2%, 42%, and 34.3% respectively.

[4] The Missed due date database contains information about Teleco orders. After submitting an order, the company has x days to deliver the product; however, if the due date is not met, then the order is flagged as a missed due date. Each order is characterized by the product (e.g., its price), type (e.g., whether it includes shipment), and assignments (e.g., their number and complexity). The target variable is due date delay level, which consists of three classes: No delay (1), 3–5 days of delay (2), and more than 5 days of delay (3), that are distributed 89%, 9.5%, and 1.5%, respectively.

[5] The Motorcycle data (Halbersberg and Lerner 2019) include motorcycle injury accidents of young drivers (YDs) who received their driving license in Israel between 2002 and 2008. Each accident is characterized by 73 variables of the driver, road, car, accident, and environment. The class variable is accident severity: Fatal (1), Severe (2), and Minor (3), which are distributed 1%, 12.5%, and 86.5%, respectively. After performing a Spearman test (0.05 confidence level) between each of the 73 variables and the class variable, 19 features were selected.

Table 23 Characteristics of selected UCI and real-world ordinal databases

	Database name	# Classes	# Variables	# Samples
1	ALS	5	29	2531
2	Australian	2	15	690
3	Autombp	10	8	392
4	Bostonhousing	10	14	506
5	Car	4	7	1728
6	Cleve	2	12	296
7	Corral	2	7	128
8	Glass2	2	10	163
9	Hepatitis	2	20	80
10	Machinecpu	10	7	209
11	Missed due date	3	20	10,500
12	Mofn	2	11	1324
13	Motorcycle	3	19	3653
14	Mushroom	2	22	8124
15	Shuttle	6	9	5800
16	Stocksdomain	10	10	950
17	Voting	2	17	232

variables and classes, and degree of imbalance, posing a range of challenges the classifiers should meet. Similar to the previous experiment, 10 random permutations were made to each database, which were each separated by CV5. That is, a total of 850 datasets are used in this experiment. Again, each of the seven algorithms trains and tests two classifiers (one for each initial graph) on each of the 850 datasets of the 17 databases (i.e., 11,900 classifiers). For each database, algorithm, and initial graph, we calculate all scores as averages over the 50 derived datasets.

Tables 24 and 25 show the average accuracies and IM_α scores achieved by the seven algorithms (all are initialized by the NBC), respectively. Table 44 in "IM scores for UCI databases" section of Appendix shows similar results for the IM score. Table 24 reveals that CEN on average is the worst method to learn a classifier. The algorithm based on IM_α (α is optimized according to Algorithm 1) has the highest ACC score for most databases (10 out of 17) and also the highest average score. Moreover, the IM_α-based BNC has a slight advantage over IM with respect to ACC for almost all databases with more than two classes (Databases 1, 3, 4, 5, 10, 11, 13, 15, and 16). This result gives another empirical justification to the development of the IM_α score that was targeted towards multiclass classification problems with a wide range of class number.

Figure 11(a) shows the IM_α-based BNC superiority to the ACC-based BNC (9 out of 17 points—a point represents a database—are above the red line, four are below the line, and four are on the line), and Fig. 11b shows superiority to the CEN-based BNC for all databases. Table 25 reveals that CEN is the poorest in terms of the IM_α measure, and the algorithms based on IM/IM_α have the highest values of this measure.

Area under curve (AUC) is a performance measure considered by many to be an alternative to accuracy because it trades between a true and false positive. The AUC has an important statistical property: the AUC of a classifier is equivalent to the probability that the classifier will rank a randomly chosen positive instance higher than a randomly chosen negative one (Fawcett 2006). In Table 26, we present the average AUC

Table 24 Mean (std) ACC values of BNCs learned using seven measures and the RMCV algorithm that is initialized by the NBC graph for 17 UCI and real-world databases

DB	IM	IM_α	MI	CEN	MCC	MAE	ACC
1	49.82 (2)	**50.53 (1)**	50.14 (1)	*46.51 (4)*	50.22 (1)	50.14 (1)	49.74 (2)
2	**86.03 (3)**	**86.03 (3)**	**86.03 (3)**	*85.74 (2)*	**86.03 (3)**	**86.03 (3)**	**86.03 (3)**
3	48.95 (5)	49.18 (6)	48.60 (5)	*48.47 (5)*	49.25 (5)	49.27 (5)	**49.43 (5)**
4	72.02 (4)	**72.14 (5)**	*70.61 (5)*	71.43 (4)	70.90 (5)	71.92 (5)	72.02 (5)
5	95.39 (2)	**95.43 (2)**	*94.74 (2)*	95.37 (2)	94.93 (2)	95.20 (2)	95.34 (2)
6	81.60 (4)	81.60 (4)	81.60 (4)	*81.53 (4)*	81.60 (4)	**81.73 (4)**	**81.73 (4)**
7	**99.57 (3)**	**99.57 (3)**	**99.57 (3)**	**99.57 (3)**	*99.07 (4)*	**99.57 (3)**	**99.57 (3)**
8	**75.09 (9)**	**75.09 (9)**	**75.09 (9)**	*64.80 (14)*	**75.09 (9)**	75.03 (9)	75.03 (9)
9	84.50 (9)	*84.25 (9)*	85.25 (9)	*84.25 (9)*	**86.13 (9)**	85.13 (9)	85.13 (9)
10	64.89 (6)	65.38 (6)	**65.60 (6)**	*64.44 (6)*	64.46 (6)	65.24 (5)	64.67 (7)
11	91.25 (1)	**92.71 (1)**	91.11 (1)	91.24 (1)	*84.55 (4)*	92.15 (1)	92.46 (1)
12	94.22 (7)	94.27 (7)	94.22 (7)	*94.19 (7)*	**94.53 (7)**	94.27 (7)	94.27 (7)
13	84.37 (1)	84.69 (1)	*84.28 (1)*	**86.65 (1)**	84.37 (1)	85.37 (1)	85.39 (1)
14	**100.00**	**100.00**	**100.00**	**100.00**	*99.97 (0)*	**100.00**	**100.00**
15	**99.60 (0)**	**99.60 (0)**	**99.60 (0)**	99.59 (0)	99.59 (0)	*99.57 (0)*	99.59 (0)
16	85.05 (3)	85.42 (3)	84.95 (3)	*84.75 (3)*	**85.76 (3)**	85.08 (3)	85.25 (3)
17	**95.08 (3)**	**95.08 (3)**	**95.08 (3)**	**95.08 (3)**	94.79 (3)	*94.75 (3)*	*94.75 (3)*
Avg (std)	82.79 (16)	**83.00 (16)**	82.73 (16)	*81.98 (17)*	82.43 (16)	82.97 (16)	82.96 (16)

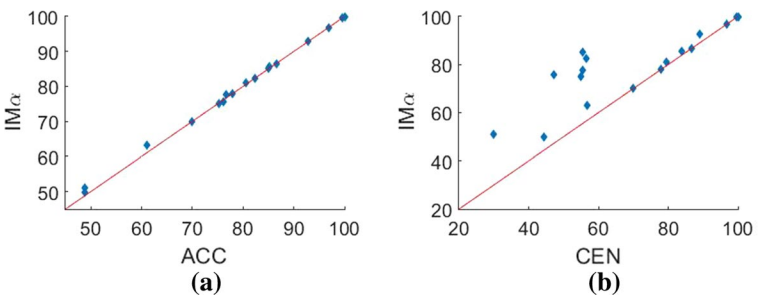

Fig. 11 Accuracies of IM_α-based BNC vs. the ACC- and CEN-based BNCs. All are initialized by the empty graph for the 17 UCI and real-world databases

results accomplished by each of the seven algorithms. For non binary databases, we use an extension for AUC to a multiclass problem as introduced by Hand and Till (2001). Table 26 shows that MAE- and IM_α-based BNCs have the highest average AUC. However, the average results of all algorithms are very similar, and the advantage is not significant. While considering only binary databases (2, 6, 7, 8, 9, 12, 14, and 17), IM_α -based BNCs ranks first for all; however, in 4 out of 8 of the binary databases, all algorithms achieved the same results, so the advantage of IM_α regarding AUC in the binary databases is also not significant.

Two other well known measures in statistics and machine learning are precision (positive predictive value) and recall (true positive rate), which are mainly used for binary

Table 25 Mean (std) normalized IM_α (multiplied by 100) values of BNCs learned using seven measures and the RMCV algorithm that is initialized by the NBC graph for 17 UCI and real-world databases

DB	IM	IM_α	MI	CEN	MCC	MAE	ACC
1	50.54 (5)	50.57 (5)	50.57 (5)	*40.89 (3)*	50.55 (5)	**50.59 (5)**	50.52 (5)
2	71.14 (4)	71.14 (4)	71.14 (4)	*70.67 (4)*	71.14 (4)	**71.16 (4)**	**71.16 (4)**
3	59.83 (3)	59.99 (4)	59.46 (3)	*59.44 (4)*	59.93 (3)	59.91 (4)	**60.15 (3)**
4	65.76 (4)	**65.78 (5)**	*64.91 (5)*	65.25 (4)	64.92 (5)	65.51 (5)	65.62 (5)
5	78.26 (5)	**78.29 (5)**	*77.61 (4)*	78.27 (5)	77.77 (4)	78.02 (4)	78.21 (5)
6	65.05 (6)	65.05 (6)	65.05 (6)	*64.92 (6)*	65.05 (6)	**65.24 (6)**	**65.24 (6)**
7	**98.32 (5)**	**98.32 (5)**	**98.32 (5)**	**98.32 (5)**	*97.29 (7)*	**98.32 (5)**	**98.32 (5)**
8	**57.49 (10)**	**57.49 (10)**	**57.49 (10)**	*47.65 (13)*	**57.49 (10)**	57.43 (10)	57.43 (10)
9	62.94 (11)	*62.58 (11)*	64.02 (11)	62.61 (11)	**65.25 (12)**	63.86 (11)	63.86 (11)
10	60.03 (3)	60.20 (3)	**60.30 (3)**	*59.70 (3)*	59.82 (4)	60.02 (3)	60.04 (4)
11	71.33 (1)	**72.00 (1)**	68.36 (2)	69.55 (2)	*61.79 (3)*	71.20 (1)	70.84 (2)
12	80.09 (12)	80.18 (12)	80.09 (12)	*80.04 (12)*	**80.72 (12)**	80.18 (12)	80.18 (12)
13	60.29 (5)	**60.45 (5)**	59.92 (5)	*59.31 (5)*	59.37 (5)	59.65 (5)	59.66 (5)
14	**99.94 (0)**	**99.94 (0)**	**99.94 (0)**	**99.94 (0)**	*99.85 (0)*	**99.94 (0)**	**99.94 (0)**
15	74.08 (4)	74.07 (4)	**74.10 (4)**	*74.02 (4)*	74.06 (4)	*74.02 (4)*	74.05 (4)
16	81.49 (3)	81.73 (3)	81.47 (3)	*81.34 (3)*	**82.07 (3)**	81.51 (3)	81.49 (3)
17	**88.21 (6)**	**88.21 (6)**	**88.21 (6)**	88.20 (6)	87.65 (7)	*87.60 (7)*	*87.60 (7)*
Avg (std)	72.05 (14)	**72.12 (14)**	71.82 (14)	*70.60 (16)*	71.45 (14)	72.01 (14)	72.02 (14)

Table 26 Mean (std $\times 10^{-1}$) AUC values of BNCs learned using seven measures and the RMCV algorithm that is initialized by the NBC graph for 17 UCI and real-world databases

DB	IM	IM_α	MI	CEN	MCC	MAE	ACC
1	**0.77 (0)**	0.76 (0)	**0.77 (0)**	*0.62 (1)*	0.76 (0)	0.75 (0)	**0.77 (0)**
2	0.92 (0)	0.92 (0)	0.92 (0)	0.92 (0)	0.92 (0)	0.92 (0)	0.92 (0)
3	0.77 (1)	0.77 (1)	0.77 (1)	0.77 (1)	0.77 (1)	0.77 (1)	0.77 (1)
4	0.95 (0)	0.95 (0)	0.95 (0)	0.95 (0)	0.95 (0)	0.95 (0)	0.95 (0)
5	*0.99 (0)*	**1.00 (0)**	*0.99 (0)*	**1.00 (0)**	**1.00 (0)**	**1.00 (0)**	**1.00 (0)**
6	0.89 (0)	0.89 (0)	0.89 (0)	0.89 (0)	0.89 (0)	0.89 (0)	0.89 (0)
7	**1.00 (0)**	**1.00 (0)**	**1.00 (0)**	**1.00 (0)**	*0.97 (0)*	**1.00 (0)**	**1.00 (0)**
8	**0.77 (1)**	**0.77 (1)**	**0.77 (1)**	*0.70 (1)*	**0.77 (1)**	**0.77 (1)**	**0.77 (1)**
9	*0.62 (4)*	**0.82 (3)**	0.63 (4)	0.65 (4)	0.81 (4)	0.80 (3)	0.64 (4)
10	**0.51 (2)**	*0.50 (2)*	**0.51 (2)**	**0.51 (2)**	*0.50 (2)*	*0.50 (2)*	*0.50 (2)*
11	*0.73 (1)*	0.77 (1)	0.76 (1)	0.75 (1)	0.74 (1)	0.75 (1)	**0.79 (1)**
12	0.98 (0)	0.98 (0)	0.98 (0)	0.98 (0)	0.98 (0)	0.98 (0)	0.98 (0)
13	**0.66 (1)**	*0.63 (1)*	0.64 (1)	**0.66 (1)**	0.61 (1)	0.65 (1)	0.65 (1)
14	1.00 (0)	1.00 (0)	1.00 (0)	1.00 (0)	1.00 (0)	1.00 (0)	1.00 (0)
15	**0.51 (1)**	**0.51 (1)**	**0.51 (1)**	**0.51 (1)**	*0.50 (1)*	**0.51 (1)**	**0.51 (1)**
16	*0.98 (0)*	**0.99 (0)**	*0.98 (0)*	*0.98 (0)*	*0.98 (0)*	*0.98 (0)*	**0.99 (0)**
17	*0.86 (3)*	**0.87 (3)**	*0.86 (3)*	*0.86 (3)*	**0.87 (3)**	**0.87 (3)**	*0.86 (3)*
Avg (std)	0.82 (2)	**0.83 (2)**	0.82 (2)	*0.81 (2)*	0.82 (2)	**0.83 (2)**	0.82 (2)

Table 27 Mean (std×10^{-1}) F-measure values of BNCs learned using seven measures and the RMCV algorithm that is initialized by the NBC graph for 17 UCI and real-world databases

DB	IM	IM_α	MI	CEN	MCC	MAE	ACC
1	0.90 (2)	**0.91 (2)**	0.90 (2)	*0.88 (3)*	0.90 (2)	0.90 (2)	0.90 (2)
2	0.85 (0)	**0.86 (0)**	**0.86 (0)**	*0.36 (0)*	0.85 (0)	0.85 (0)	0.85 (0)
3	**0.41 (1)**	**0.41 (1)**	0.40 (1)	*0.12 (1)*	0.37 (1)	0.37 (1)	0.36 (1)
4	0.74 (1)	0.74 (1)	0.74 (1)	*0.13 (0)*	**0.75 (1)**	**0.75 (1)**	0.74 (1)
5	0.21 (0)	0.21 (0)	0.21 (0)	0.21 (0)	0.21 (0)	0.21 (0)	0.21 (0)
6	**0.78 (1)**	**0.78 (1)**	0.77 (1)	*0.39 (1)*	0.77 (1)	0.77 (1)	0.77 (1)
7	**0.82 (1)**	**0.82 (1)**	0.81 (1)	*0.38 (1)*	**0.82 (1)**	**0.82 (1)**	**0.82 (1)**
8	**0.76 (1)**	**0.76 (1)**	**0.76 (1)**	*0.43 (2)*	**0.76 (1)**	**0.76 (1)**	**0.76 (1)**
9	0.69 (2)	0.68 (2)	**0.71 (2)**	*0.64 (2)*	0.68 (2)	*0.64 (2)*	*0.64 (2)*
10	**0.23 (1)**	**0.23 (1)**	**0.23 (0)**	*0.11 (0)*	0.22 (0)	0.22 (1)	0.20 (1)
11	**0.56 (0)**	**0.56 (0)**	**0.56 (0)**	*0.31 (0)*	**0.56 (0)**	**0.56 (0)**	0.55 (0)
12	0.44 (0)	0.44 (0)	0.44 (1)	0.44 (0)	0.44 (0)	0.44 (0)	0.44 (0)
13	0.33 (0)	0.32 (0)	**0.40 (0)**	*0.31 (0)*	0.36 (0)	*0.31 (0)*	*0.31 (0)*
14	1.00 (0)	1.00 (0)	1.00 (0)	1.00 (0)	1.00 (0)	1.00 (0)	1.00 (0)
15	*0.51 (0)*	**0.60 (1)**	*0.51 (0)*	0.52 (0)	**0.60 (1)**	0.59 (1)	*0.51 (0)*
16	**0.86 (0)**	**0.86 (0)**	**0.86 (0)**	*0.85 (0)*	**0.86 (0)**	**0.86 (0)**	**0.86 (0)**
17	0.97 (0)	0.97 (0)	0.97 (0)	0.97 (0)	0.97 (0)	0.97 (0)	0.97 (0)
Avg (std)	0.65 (3)	**0.66 (3)**	0.65 (3)	*0.47 (3)*	0.65 (3)	0.65 (3)	0.64 (3)

classification problems (Baccianella et al. 2009), but have an extended version for multiclass problems (Sokolova and Lapalme 2009): $\text{recall}_i = \frac{M_{ii}}{\sum_j M_{ij}}$ and $\text{precision}_i = \frac{M_{ii}}{\sum_j M_{ji}}$. These measures are calculated for each class separately. The results for all classes can then be averaged as micro (favors bigger classes) or macro (treats all classes equally) and merged to $Fmeasure = \frac{2 \cdot precision \cdot recall}{precision + recall}$ (Sokolova and Lapalme 2009). In precision, recall, and F-measure, all error types are equal; hence, no information about the error distribution is taken into consideration (fits to nominal multiclass problems). Table 27 presents the performance of the seven algorithms when the classifier is evaluated using the F-measure. According to Table 27, the IM_α-based BNC has the best average F-measure and is also ranked first in all but three databases.

We further analyze the time complexity of each algorithm [full results are in Table 45 ("Run Time measured by number of neighbors for UCI BNs" section of Appendix)]. The IM_α-based BNC suffers from the worst time complexity, which is the number of α's checked times longer than that of the other classifiers.

Table 28 summarizes the average Friedman's ranks according to ACC, IM, MAE, and MI scores in rows 1–4, respectively. Because results regarding the average rank of AUC had shown no superiority to any of the algorithms, they were excluded. Each column represents an algorithm and each row stands for a different evaluation measure and initial graph. Table 28 shows that the IM_α-based BNC has the lowest (best) average rank followed by IM, regardless of the initial graph or the evaluation measure (again, in addition to the IM measure, we present the MAE and MI measures that compose IM and IM_α to show that the success of our proposed measure is attributed to the improvement in both measures).

Table 28 Average Friedman's ranks according to the ACC, IM, and MAE of BNCs learned using seven measures and two initializations for 17 UCI and real-world databases

	Initial	IM	IM_α	MI	CEN	MCC	MAE	ACC
1. ACC score	Empty	2.6	**2.1**	4.1	*6.0*	4.1	4.3	4.8
	NBC	3.1	**2.3**	4.6	*5.6*	4.5	3.9	4.0
2. IM score	Empty	2.3	**1.8**	3.6	*6.2*	4.4	4.5	5.3
	NBC	2.6	**2.3**	3.8	*5.5*	4.8	4.3	4.8
3. MAE score	Empty	2.5	**1.9**	4.4	*5.9*	4.2	4.2	5.0
	NBC	3.1	**2.4**	4.3	*5.3*	4.7	3.7	4.5
4. MI score	Empty	2.2	**2.1**	3.0	*6.0*	4.4	4.5	5.6
	NBC	2.9	**2.7**	3.8	*5.7*	4.3	4.0	4.8

Table 29 Nemenyi post hoc test according to ACC of BNCs learned using seven measures and two initializations for 17 UCI and real-world databases. Values in the table stand for algorithms for which the column headline is superior

Initial	IM	IM_α	MI	CEN	MCC	MAE	ACC
Empty	CEN, ACC	MI, CEN, MCC, MAE, ACC	–	–	–	–	–
NBC	CEN	MI, CEN, MCC	–	–	–	–	–

Table 30 Wilcoxon post hoc test according to ACC of BNCs learned using IM, IM_α, and ACC and two initializations for 17 UCI and real-world databases

Initial	IM_α vs. IM	IM_α vs. ACC
Empty	×	√
NBC	√	×

Tick stands for statistically superiority of IM_α and 'X' stands for non-statistical significant

Table 31 Nemenyi post hoc test according to IM of BNCs learned using seven measures and two initializations for 17 UCI and real-world databases

Initial	IM	IM_α	MI	CEN	MCC	MAE	ACC
Empty	CEN, MCC, MAE, ACC	CEN, MCC, MAE, ACC	CEN	–	–	–	–
NBC	CEN, MCC, ACC	CEN, MCC, MAE, ACC	–	–	–	–	–

Values in the table stand for algorithms for which the column headline is superior

Next, we proceeded to conduct Nemenyi and Wilcoxon post hoc tests. Table 29 summarizes the Nemenyi post hoc test for ACC. Each column represents a baseline algorithm to which the rest of the algorithms are compared. For BNCs initialized by an empty graph, the algorithm based on IM_α significantly outperforms all other BNCs except for IM. For the NBC initial graph, BNC-ACC and BNC-IM were not dominant by the IM_α-based BNC; however, according to the Wilcoxon test (Table 30), the IM_α-based BNC is superior to the IM-based BNC for the NBC initialization and to the ACC-based BNC for the empty graph

Table 32 Wilcoxon post hoc test according to IM of BNCs learned using IM, IM_α, and ACC and two initializations for 17 UCI and real-world databases

Initial	IM_α vs. IM	IM_α vs. ACC
Empty	×	√
NBC	√	√

Tick stands for statistically superiority of IM_α and 'X' stands for non-statistical significant

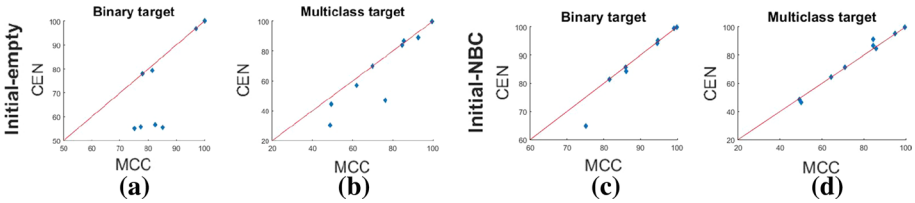

Fig. 12 Accuracies of CEN- versus MCC-based BNCs for 17 UCI and real-world databases. **a** all binary databases and empty initial graph, **b** all multiclass databases and empty initial graph, **c** all binary databases and NBC-based initialization, and **d** all multiclass databases and NBC-based initialization

initialization. Table 31 reveals that in contradiction to the comparison of ACC values, while comparing the IM scores, the IM-based BNC is statistically significantly superior to CEN-, MCC-, MAE-, and ACC-based BNCs regardless of the initial graph. Also, it reveals that the IM_α-based BNC is not superior to the MI-based classifier, but is superior to all other classifiers. The Wilcoxon test displayed in Table 32 shows that for the NBC initial graph, the IM_α-based BNC is superior also to the IM-based BNC, which was not the case in the Nemenyi test, besides being significantly superior to the ACC-based BNC, which was not the case when compared based on ACC in Table 30. If we allow the confidence level in the Nemenyi test to be 0.1, then the IM_α-based BNC is superior to the IM-based BNC also in the Nemenyi post hoc test.

Discussion of UCI and real-world databases experiments

After showing the advantages of our proposed algorithm on artificial databases, in this experiment, we focused on UCI and real-world databases from a variety of problems/domains. The experiment showed that IM and IM_α did not fall behind the ACC-based BNC, and were even better with respect to classification accuracy (not significant) and IM score (significant). In most of the databases with a high number of classes, such as ALS, Bostonhousing, Car, Missed due date, and Shuttle, the IM_α-based BNC outperforms all other classifiers (even) with respect to accuracy. Also, on average, it has the best ACC, IM, and IM_α, and the lowest (best) rank regardless of the initial graph. Also, the IM_α-based BNC showed better AUC and F-measure results. However, the fact that AUC scores for all algorithms were very similar may indicate that there is a lack of ability to compare imbalanced-ordinal databases based on this measure.

The IM_α-based BNC's better results come at the expense of run time, which on average is five times higher than that of the ACC/IM-based BNCs (we did not run the wrapper IM_α in parallel with different values of alpha, but rather in sequential mode). This is because of the need to optimize α.

Once again, the CEN-based BNC was found to be the least efficient. It has already been argued by Jurman et al. (2012) that CEN is not reliable in the binary case and that MCC

Table 33 Mean (std) ACC values of eight state-of-the-art algorithms and BNC-IM$_\alpha$ that is initialized by the NBC graph for 23 artificial databases

DB	DT-ord	DT-cost	NN	RF	SVM	SVM-smt	TAN	IM$_\alpha$
1	95.97 (1)	95.97 (1)	*92.44 (2)*	96.75 (1)	93.02 (1)	93.02 (1)	96.20 (1)	**97.44 (1)**
2	86.75 (2)	86.53 (2)	*69.84 (1)*	87.25 (2)	78.58 (4)	78.58 (4)	83.91 (2)	**90.75 (1)**
3	71.85 (3)	71.74 (3)	58.24 (2)	72.24 (2)	*55.82 (3)*	*55.82 (3)*	63.93 (5)	**79.81 (2)**
4	63.75 (3)	63.27 (3)	*52.04 (2)*	65.86 (3)	60.66 (3)	60.66 (3)	63.48 (3)	**71.26 (3)**
5	56.96 (3)	55.60 (3)	*42.08 (3)*	56.60 (3)	46.00 (4)	46.00 (4)	48.55 (4)	**65.96 (3)**
6	49.98 (3)	49.60 (3)	*39.68 (3)*	51.70 (3)	41.58 (4)	41.58 (4)	44.52 (4)	**59.99 (3)**
7	46.53 (3)	45.83 (3)	36.92 (2)	46.84 (3)	*34.46 (3)*	*34.46 (3)*	37.87 (3)	**55.71 (3)**
8	42.30 (2)	40.70 (3)	33.48 (2)	42.69 (3)	*33.28 (3)*	*33.28 (3)*	35.86 (3)	**52.31 (2)**
9	63.95 (6)	64.42 (5)	*48.80 (7)*	64.67 (5)	54.08 (6)	54.08 (6)	57.25 (6)	**68.83 (5)**
10	68.25 (4)	67.98 (4)	*56.08 (4)*	68.83 (3)	56.14 (5)	56.14 (5)	59.77 (5)	**76.11 (3)**
11	70.34 (3)	69.94 (3)	55.79 (2)	71.79 (2)	*55.48 (5)*	*55.48 (5)*	61.23 (4)	**78.05 (2)**
12	71.85 (3)	71.74 (3)	58.24 (2)	72.24 (2)	*55.82 (3)*	*55.82 (3)*	63.93 (5)	**79.81 (2)**
13	72.37 (2)	72.27 (2)	58.08 (1)	73.11 (2)	*56.89 (5)*	*56.89 (5)*	62.78 (4)	**80.54 (2)**
14	73.49 (2)	73.54 (2)	58.67 (2)	73.97 (2)	*56.48 (4)*	*56.48 (4)*	65.35 (3)	**81.55 (1)**
15	71.32 (3)	71.16 (3)	58.92 (1)	71.91 (3)	*56.56 (5)*	*56.56 (5)*	60.27 (4)	**79.98 (2)**
16	71.24 (2)	70.76 (2)	58.96 (2)	71.94 (3)	56.38 (3)	*56.34 (3)*	61.60 (4)	**78.79 (2)**
17	71.00 (3)	70.54 (2)	59.08 (2)	70.90 (2)	54.68 (4)	*54.48 (4)*	58.17 (4)	**78.65 (2)**
18	73.23 (3)	73.02 (3)	62.96 (3)	74.18 (3)	*60.97 (3)*	61.27 (3)	66.00 (4)	**79.13 (2)**
19	73.34 (3)	73.01 (2)	62.48 (2)	74.31 (2)	*61.29 (3)*	61.33 (3)	69.22 (3)	**78.94 (2)**
20	71.95 (2)	71.89 (2)	64.76 (3)	74.04 (2)	66.58 (2)	*63.86 (3)*	68.85 (3)	**78.64 (2)**
21	73.68 (2)	73.66 (3)	72.84 (2)	77.21 (2)	73.63 (2)	*67.95 (2)*	75.88 (2)	**78.89 (3)**
22	73.86 (3)	73.72 (3)	74.88 (6)	78.37 (2)	76.74 (2)	*69.50 (3)*	78.47 (2)	**79.47 (2)**
23	83.83 (1)	83.70 (1)	81.64 (2)	86.21 (1)	85.77 (2)	*60.76 (6)*	86.78 (1)	**86.92 (2)**
Avg (std)	69.47 (12)	69.16 (12)	59.00 (14)	70.59 (12)	59.60 (15)	*57.84 (13)*	63.91 (14)	**76.41 (10)**

should be preferred as an optimal off-the-shelf tool in practical tasks. In this experiment, we showed that this claim stands true also in multiclass problems. A comparison showing the superiority of MCC-based BNCs over CEN-based ones can be seen in Fig. 12, which demonstrates the differences between MCC- and CEN-based BNCs with respect to: (1) empty graph (Fig. 12a, b) versus NBC-based initializations (Fig. 12c, d), and (2) binary (Fig. 12a, c) versus multiclass (Fig. 12b, d) classifications. It can be seen that for multiclass problems, when the initial graph is empty (Fig. 12b), MCC is superior to CEN (this is supported by the Wilcoxon test with a 0.05 confidence level). For an NBC-based initial graph (Fig. 12d), although there is no statistical superiority: in five databases, MCC has higher accuracy, in three CEN leads, and one ends in a tie. For the binary databases, the results are similar (significance superiority in favor of MCC for an empty initial graph, and non-significance superiority for the NBC-based initialization).

6.3 Comparison to state-of-the-art algorithms

In this section, we compare the proposed algorithm to other state-of-the-art machine learning algorithms. Tables 33 and 34 summarize the ACC results for seven algorithms compared with our proposed algorithm to artificial (Sect. 6.1) and real-world (Sect. 6.2)

Table 34 Mean (std) ACC values of eight state-of-the-art algorithms and BNC-IM$_\alpha$ that is initialized by the NBC graph for 17 UCI and real-world databases

DB	DT-ord	DT-cost	NN	RF	SVM	SVM-smt	TAN	IM$_\alpha$
1	*44.22 (2)*	44.99 (2)	45.60 (3)	50.07 (2)	**50.94 (2)**	48.71 (2)	50.01 (2)	50.53 (1)
2	84.00 (3)	84.00 (3)	*65.36 (6)*	85.70 (3)	85.07 (3)	85.07 (3)	83.33 (3)	**86.03 (3)**
3	48.88 (5)	47.05 (5)	*41.25 (8)*	**50.20 (5)**	48.85 (5)	45.35 (5)	49.00 (6)	49.18 (6)
4	73.06 (4)	73.06 (4)	*71.57 (5)*	75.71 (5)	**75.92 (4)**	75.51 (4)	73.22 (5)	72.14 (5)
5	95.39 (1)	95.13 (1)	**97.93 (1)**	97.41 (1)	86.79 (1)	*86.11 (2)*	87.70 (4)	95.43 (2)
6	78.60 (5)	78.60 (5)	*72.33 (8)*	82.30 (4)	**83.47 (4)**	**83.47 (4)**	80.20 (4)	81.60 (4)
7	91.07 (6)	91.07 (6)	**100.00**	98.50 (3)	88.86 (6)	88.86 (6)	*85.21 (8)*	99.57 (3)
8	76.17 (8)	76.17 (8)	75.43 (10)	**76.34 (8)**	75.66 (7)	75.66 (7)	*73.54 (9)*	75.09 (9)
9	84.50 (9)	84.50 (9)	81.25 (10)	**86.50 (8)**	82.63 (9)	81.13 (10)	*77.63 (11)*	84.25 (9)
10	64.22 (7)	61.51 (7)	*57.78 (6)*	62.44 (6)	60.27 (7)	61.96 (8)	64.84 (5)	**65.38 (6)**
11	**93.06 (0)**	92.98 (0)	92.16 (1)	93.05 (1)	92.73 (0)	*89.92 (1)*	91.12 (1)	92.71 (1)
12	99.85 (1)	99.85 (1)	**100.00**	99.25 (1)	99.93 (0)	99.96 (0)	*92.66 (3)*	94.27 (7)
13	82.88 (1)	*82.64 (1)*	**86.66 (2)**	85.54 (1)	**86.66 (1)**	78.88 (3)	84.89 (1)	84.69 (1)
14	**100.00**	**100.00**	**100.00**	**100.00**	*95.64 (1)*	*95.64 (1)*	**100.00**	**100.00**
15	99.65 (0)	99.64 (0)	*95.40 (3)*	**99.78 (0)**	98.84 (0)	97.21 (1)	99.57 (0)	99.60 (0)
16	88.89 (2)	89.01 (2)	86.11 (2)	**90.28 (2)**	*84.01 (2)*	*84.04 (2)*	85.78 (2)	85.42 (3)
17	95.75 (3)	95.75 (3)	*92.08 (5)*	**96.13 (3)**	94.67 (3)	94.67 (3)	95.37 (3)	95.08 (3)
Avg (std)	82.36 (17)	82.11 (17)	*80.05 (19)*	**84.78** (16)	81.82 (15)	80.71 (16)	80.83 (15)	83.00 (16)

databases, respectively. The experiments were performed using LIBSVM (Chang and Lin 2011) (SVM), PRTOOLS (Duin et al. 2000) (NN), and the Matlab Statistics and Machine Learning Toolbox (DT and RF). We used a linear kernel for SVM since among the linear, polynomial, and Gaussian kernels, it was found in Kelner and Lerner (2012) to have the highest average accuracy over 22 UCI databases. We used ordinal classification implementation for DT (Frank and Hall 2001) (denoted as DT-ord). We also added to the comparison DT with the equivalent cost matrix that was derived by IM and IM$_\alpha$ (denoted as DT-cost), where each cell in the cost matrix is equal to $|x - y|$. These two DTs are advantageous to the conventional DT for the examined scenarios of ordinal nature and different error severities. In addition, we included the tree augmented naïve Bayes (TAN) (Friedman et al. 1997), which is a supreme BNC (and therefore saw no need to include the inferior NBC). Also included in the comparison is an SVM that was trained after synthetically balancing each dataset using the synthetic minority over-sampling technique, SMOTE, denoted as SVM-smt. SMOTE (Chawla et al. 2002), as opposed to random sampling, uses a more educated sampling technique to combine both downsampling the majority class and creation of synthetic minority class examples (upsampling) by introducing synthetic examples to each minority sample according to the feature space of its k nearest neighbors. We chose to compare the state-of-the-art algorithms to the IM$_\alpha$-based BNC that is initialized with NBC since it achieved the highest performances in previous experiments.

As can be seen from Table 33 for the artificial databases, the IM$_\alpha$-based BNC is ranked first for all databases (and therefore also has the highest average accuracy) and is obviously superior to all other algorithms with no need for any statistical tests. Notice that SVM and SVM-smt have different accuracies only for the imbalanced databases 16–23 (see Table 14). Note also, that by focusing on the minority classes, SVM-smt misses the

Table 35 Average Friedman's ranks according to ACC and IM measures of state-of-the-art algorithms for 23 artificial databases

	DT-ord	DT-cost	NN	RF	SVM	SVM-smt	TAN	IM_α
1. ACC score	3.5	4.0	6.6	2.3	6.4	*7.6*	4.5	**1**
2. IM score	3.7	3.4	*7.6*	2.2	6.8	6.5	4.6	**1**

Table 36 Average Friedman's ranks according to ACC and IM measures of state-of-the-art algorithms for 17 UCI and real-world databases

	DT-ord	DT-cost	NN	RF	SVM	SVM-smt	TAN	IM_α
1. ACC score	3.7	4.6	5.4	**2.3**	4.5	*5.9*	5.5	4.1
2. IM score	3.6	4.2	*5.6*	**2.8**	5.5	5.0	5.3	3.0

majority classes with the consequence of lower overall accuracy than the conventional SVM. By Table 34, RF outperforms the other algorithms for six of the 17 databases and gains the highest average accuracy. Although ranked first for only three of the databases, the IM_α-based algorithm has the second highest average accuracy (above NN and SVM and only second to RF), and is never ranked last (an achievement that is shared only by RF).

We compared algorithm performances for the artificial as well as the UCI and real-world databases for each score function separately using Friedman's non parametric test and a Nemenyi post hoc test. Table 35 shows that, for the artificial databases, IM_α-based BNC is ranked first with a large margin from the RF and DT-ord algorithms that follow. The superiority of the IM_α-based BNC to all other algorithms is significant. Table 36 reveals that, for the UCI and real-world databases, RF has the highest average ranks followed by either DT-ord (if measured according to ACC) or the IM_α-based BNC (if measured according to the IM score). The difference between the RF and IM_α-based BNC is vivid regarding ACC, but negligible regarding IM. Regarding IM—the more important measure of the two—the Nemenyi post hoc test (performed with a 0.05 confidence level) shows that the RF and IM_α-based BNC are superior to NN. Further, a Wilcoxon post hoc test (with a 0.05 confidence level) found the RF and IM_α-based BNC to also be significantly superior to SVM and TAN. In addition, Table 36 shows the impact of SMOTE on the SVM. Interestingly, SVM is superior with respect to ACC, and SVM-smt is significantly better with respect to the IM score. Nevertheless, both are behind the IM_α-based algorithm regardless of the score metric.

As a concluding evaluation of the classifiers, let us analyze their confusion matrices for the two real-world problems we described in Sect. 1, which helped motivate this study: prediction of the severity of a YD motorcycle accident and prediction of the disease state of an ALS patient (Databases 13 and 1, respectively, in Table 23). As we recall, these are ordinal class-imbalance problems for which the severity of the error should be accounted. Table 34 shows that the SVM and NN achieved the best ACC performance (86.66%) for Database 13 (YD accidents). However, considering their confusion matrices, we see that the SVM (Table 37a) predicted all samples to the majority class of minor accidents, and the NN (Table 37b) did not predict even a single fatal accident and only very few severe accidents, which make both

Table 37 Confusion matrices for the YD motorcycle accident database of SVM, NN, and IM_α

Predicted class (X)	True class (Y)		
	Fatal	Severe	Minor
(a) Confusion matrix for SVM			
Fatal	0	0	0
Severe	0	0	0
Minor	6.6	91.2	635.2
(b) Confusion matrix for NN			
Fatal	0	0	0
Severe	1.1	9.8	9.8
Minor	5.4	81.4	625.4
(c) Confusion matrix for SVM-smt			
Fatal	0.9	7.4	26.5
Severe	1.7	16.8	48.2
Minor	4.0	67.1	560.5
(d) Confusion matrix for IM_α-based BNC			
Fatal	0.3	1.4	3.2
Severe	2.2	22.0	35.8
Minor	4.1	67.8	596.2

classifiers uninformative and practically not useful. The SVM that is based on SMOTE (SVM-smt) slightly improved the prediction of the minority class of fatal accidents (Table 37c), but at the expense of too many false alarms (e.g., on average, 26.5 and 48.2 minor accidents were misclassified as fatal and severe, respectively, compared to the conventional SVM (Table 37a)). This is a common disadvantage of all sampling techniques. The differences between Tables 37(b) and 37(c) also demonstrate the disparity that was revealed in Table 36 between SVM and SVM-smt, where the former was ranked higher than the latter in accuracy, but lower with respect to IM. Although the IM_α-based BNC is less accurate than the SVM and NN for the YD database in 2% (34), its confusion matrix (Table 37d) shows more accurate predictions for the two minority classes of severe and fatal accidents that make the classifier more informative and valuable practically. SVM-smt, which is more accurate in the prediction of fatal accidents, is less accurate for severe and minor accidents, which makes it, overall, inferior to the IM_α-based BNC. Other traditional methods in addition to DT-cost, DT-ord, and SVM-smt to tackle the ordinal class imbalance problem represented in this database, such as upsampling the fatal accidents for DT and ordinal regression by the logit model, were evaluated and found inferior to the IM_α-based BNC in Halbersberg and Lerner (2019).

Similarly, for the ALS problem, the SVM, RF, and IM_α-based BNC show exactly the same accuracy (50%), but comparison of their confusion matrices (Table 38) shows that the IM_α-based BNC is the most or second-most accurate classifier for all disease states except the state describing patient's "full functionality" (State 4). The RF is never the best disease-state predictor (the IM_α-based BNC is superior to RF for all classes except for State 4), the SVM is the most accurate classifier twice (States 2 and 4), but also the least accurate three times, and the SVM-smt causes once again too many false alarms for the minority class that describes patient's "non-functionality"

Table 38 Confusion matrices for the ALS database of SVM, RF, and IM_α

Predicted Class (X)	True Class (Y)				
	0	1	2	3	4
(a) Confusion matrix for SVM					
0	3.9	2.3	2.8	1.9	0
1	0.8	0.2	0.7	0.2	0
2	14.7	19.1	25.9	22.2	1.4
3	1.8	2.6	6.7	7.5	5.2
4	3.9	9.4	44.2	108.9	220.7
(b) Confusion matrix for RF					
0	5.2	4.7	4.1	2.9	0.1
1	5.1	4.5	5.7	3.9	0.7
2	7.3	10.5	17.8	16.6	5.2
3	5.7	9.3	25.4	40.9	35.9
4	1.8	4.5	27.4	76.4	185.6
(c) Confusion matrix for SVM-smt					
0	11.8	11.9	13.3	12.9	3.6
1	7.2	8.0	16.1	14.5	4.1
2	3.9	7.0	15.2	17.6	8.7
3	0.2	1.1	3.4	5.2	4.1
4	1.8	5.6	32.4	90.5	206.8
(d) Confusion matrix for IM_α-based BNC					
0	7.1	6.1	4.9	4.9	3.0
1	6.2	5.6	6.3	6.8	0.9
2	5.5	12.1	19.3	18.2	11.3
3	5.2	6.1	27.8	43.6	31.5
4	1.0	3.2	22.0	67.3	180.6

(State 0). Moreover, SVM-smt (similar to SVM) shows poor results for State 3, with only 5.2 patients on average that were correctly classified (compared to 43.6 by the IM_α-based BNC).

7 Discussion

By minimizing the 0/1 loss function, the BNC, which is a powerful tool in knowledge representation, can also guarantee accurate classification. However, similar to other classifiers, the BNC focuses on the majority class, and therefore, misclassifies minority classes; is usually uninformative about the distribution of misclassifications; and is insensitive to error severity (making no distinction between misclassification types).

We have proposed a measure—the information measure (IM)—that is more appropriate for learning and evaluating the BNC because it jointly maximizes the classification accuracy and information, and accounts for the error distribution, class imbalance, and error severity in the domain. We motivated this measure theoretically. We then extended it using a control parameter that provides more flexibility in meeting the problem requirements. This parameter can be user defined or be set using a wrapper and a validation set.

To expedite the search for the optimal value of the parameter using a wrapper, we suggest parallelizing the search. Alternatively, setting the parameter can be performed in a Bayesian setting.

We evaluated the measure in comparison to seven common measures using synthesized confusion matrices, twenty-three artificial databases, seventeen UCI and real-world databases, and different performance measures. We showed that an IM-based BNC is superior to BNCs learned using the other measures for ordinal classification and/or imbalanced problems, and is not inferior to state-of-the-art classifiers with respect to accuracy. More importantly, this BNC provides vital information about the distribution of errors and classifies well all classes and not just the majority one. Our experiments encourage application of the IM-based BNC to other problems for which joint maximization of accuracy and information is needed, the data is imbalanced, and/or the problem is ordinal, whether the classifier is a BNC or not.

In addition, we demonstrated the advantages of the IM_α-based BNC in better analyzing real-world complex problems, such as in road safety and medical diagnosis. In further research, this classifier can be applied to other domains for which both accuracy and information are needed, the classes are imbalanced, and/or the cost of different misclassifications is different. Also for further research is the application of the information measure to other classifiers, e.g., to determine the splitting variable in each level of training a decision tree.

Appendix

Information measure with alpha

$$
\begin{aligned}
IM &= -MI + ES \\
&= \sum_x \sum_y -P(x,y)\log\left(\frac{P(x,y)}{P(x)P(y)}\right) + \sum_x \sum_y P(x,y)\log(1+|x-y|)) \\
IM_\alpha &= \sum_x \sum_y -P(x,y)\log\left(\frac{\alpha P(x,y)}{P(x)P(y)}\right) + \sum_x \sum_{y\neq x} P(x,y)\log\left(\alpha(1+|x-y|)\right) \\
&= \sum_x \sum_y \left(-P(x,y)\log(\alpha) - P(x,y)\log\left(\frac{P(x,y)}{P(x)P(y)}\right)\right) \\
&\quad + \sum_x \sum_{y\neq x} \left(P(x,y)\log(\alpha) + P(x,y)\log(1+|x-y|)\right) \\
&= \sum_x \sum_y -P(x,y)\log(\alpha) \\
&\quad + \sum_x \sum_y -P(x,y)\log\left(\frac{P(x,y)}{P(x)P(y)}\right) + \sum_x \sum_{y\neq x} \left(P(x,y)\log(\alpha)\right) \\
&\quad + \sum_x \sum_{y\neq x} P(x,y)\log\left(1+|x-y|\right) \\
&= -\log(\alpha) + \log(\alpha)\sum_x \sum_{y\neq x} P(x,y) + \sum_x \sum_y P(x,y)\left(-\log\left(\frac{P(x,y)}{P(x)P(y)}\right) + \log(1+|x-y|)\right) \\
&= IM - \log(\alpha) + \log(\alpha)(1-ACC) \\
&= IM - \log(\alpha)ACC
\end{aligned}
\tag{20}
$$

Sensitivity analysis

In this section, we give theoretical support for the experiments presented in Sect. 5 and particularly to Table 13. Since Table 13 consists of 7 measures × 5 properties = 35 cases, we concentrate here only on the most interesting or unexplored combinations of measure and property. This appendix is organized according to the order by which the measures are presented in Table 13.

In most of the cases for which we wish to show insensitivity of a measure to a property, we give an example by which we make a single change to the property, as reflected in a classifier confusion matrix[6], and show that the measure does not change (i.e., manifests insensitivity). Thus, it is necessary to require that the sum over the confusion matrices (i.e., the total number of samples in the test set) before and after the change remains fixed in order to analyze the sensitivity to the property. The notation we use is that (similarly to Sect. 5) y and x are the true and predicted values for a class, and Y and X are these values for all M classes, respectively. In addition, i and j are assignments to specific classes (i for x and j for y) we are interested in.

Accuracy

According to Table 13, accuracy is not sensitive to any of the properties except partially to the number of classes. To show this, recall that accuracy is the confusion matrix trace (sum of the matrix diagonal) divided by the total number of samples, and thus, first, is insensitive to the diagonal distribution (i.e., insensitive to *class imbalance*). Second, if the number of classes changes (say, by joining two existing classes i and j and not by adding/removing a class, which changes the problem), accuracy remains insensitive to this number if there were no errors in misclassifying class i as class j or vice versa (keeping the trace/accuracy unchanged). However, if this is not the case, accuracy becomes sensitive, and thus, overall, it is only partially sensitive to the *number of classes*. In addition, since accuracy is also defined as one minus the total number of errors, it is insensitive to the *error distribution* and *severity*, and therefore cannot also *trade accuracy and information*.

Mean absolute error

1. *Class imbalance*: Table 13 indicates that the MAE is only partially sensitive to class imbalance. We demonstrate MAE insensitivity in a special case where imbalance is reflected only on-diagonal (as in the introduction to this appendix, this is enough to demonstrate insensitivity for a single case). Consider two confusion matrices for the same number of samples and the same off-diagonal elements, but with different on-diagonal class distributions. Although this class imbalance is along the diagonal, the two matrices have the same MAE, which means that for this case, MAE is not sensitive to class imbalance (of course if the imbalance was reflected also off diagonal, then the MAE could have been changed accordingly).

[6] Recall that using the confusion matrix of a classifier already trained according to a certain measure can directly exhibit the measure properties without really training the classifier, and as we exercised this approach already in Sect. 5, we also do it here.

2. *Number of classes*: Table 13 also indicates that the MAE is only partially sensitive to the number of classes. As we mentioned for accuracy, there are two approaches to demonstrate (in)sensitivity to the number of classes: (1) removal/addition of a class from/to the confusion matrix, and (2) merging two classes into one. Since the total number of samples should be kept between the scenarios, the latter approach is more realistic than the former, which also defines a new problem. Thus, let's consider a confusion matrix A with $M > 2$ classes. Also, let's merge the ith and jth true classes of A (without loss of generality, assume $j > i$) to form a confusion matrix B. Assume there were no misclassifications in A with respect to the j class (i.e., the j class is not "involved" in any misclassification), the MAE elements of the merged class in B and those of the two original classes in A have the same error severity. That is, the MAE of the two matrices is equal, which means that the MAE is insensitive to the number of classes. Note that Sect. 5.2 demonstrates an example for this insensitivity using the first approach above (introducing a new class).
3. *Error distribution*: Table 13 indicates that the MAE is only partially sensitive to the error distribution. It is very easy to change the error distribution of true class j but to keep the MAE intact by changing the error distribution of another true class i to compensate for the change in class j. It is more challenging, though, to show the MAE indifference to the error distribution by changing only the distribution of a single class, but without changing the sum of errors of that class. To show this, we use the case of symmetrical error distributions (e.g., uniform, normal, Laplace). For example, consider the two error severity frequency distributions (with an equal $MAE = 3.5$): $V_{A_j} = \{10, 10, 10, 10, 10, 10\}$ and $V_{B_j} = \{5, 10, 15, 15, 10, 5\}$ representing uniform and normal distributions of the error severity for true class j and confusion matrices A and B, respectively. V_{B_j}, e.g., demonstrates that there are in B five samples of true class j wrongly classified with error severity of one, ten samples of true class j wrongly classified with error severity two, 15 with error severity three, etc. until error severity six (the dimension of V_{B_j}). This example will inspire us in the proof of the following lemma that the MAE is insensitive to a change in error distribution of a single class if the distributions are symmetrical and the sum of errors for that class (and since this is the only class to change also for the entire confusion matrix) is kept intact.

Lemma 3 *Two confusion matrices of two classifiers induced using the same data have the same MAE if their corresponding error severity distributions per class are either equal or each is symmetrical.*

Proof Without loss of generality, we change the error distribution of class j (of M classes) between two confusion matrices A and B, but without changing their sum of errors (i.e., the total number of errors for class j in A and B is equal). Assume that error severities 1 to m_j for true class j are symmetrical in A and B (recall V_{A_j} and V_{B_j} in the example above) and distributed, respectively:

$$P_{A_{y=j}} = \frac{1}{S_j}\{e^1_{A_{y=j}}, e^2_{A_{y=j}}, ..., e^{m_j}_{A_{y=j}}\} \quad and \quad P_{B_{y=j}} = \frac{1}{S_j}\{e^1_{B_{y=j}}, e^2_{B_{y=j}}, ..., e^{m_j}_{B_{y=j}}\},$$

where m_j is the maximal error severity for class j ($m_j \leq M - 1$), S_j is the number of samples of true class j, and $e^k_{A_{y=j}}$ is the number of samples of true class j in A that were wrongly clas-

sified to class x s.t. $|x-j|=k$, and $1 \leq k \leq m_j$. Note, that $e^k_{A_{y=j}}$ should not necessarily be equal to $e^k_{B_{y=j}}$.

Since both error severity frequency distributions are symmetrical (recall V_{A_j} and V_{B_j}), we get for $P_{A_{y=j}}$ and an even m_j (and similarly for $P_{B_{y=j}}$ and/or an odd m_j):

$$e^1_{A_{y=j}} = e^{m_j}_{A_{y=j}},\ e^2_{A_{y=j}} = e^{m_j-1}_{A_{y=j}}, ...,\ e^{m_j/2}_{A_{y=j}} = e^{m_j/2+1}_{A_{y=j}}.$$

Due to this symmetry, we get that:

$$\begin{aligned} MAE_{A_{y=j}} &= 1/S_j\left([1+m_j]e^1_{A_{y=j}} + [2+m_j-1]e^2_{A_{y=j}} + ... + [\frac{m_j}{2}+\frac{m_j}{2}+1]e^{m_j/2}_{A_{y=j}}\right) \\ &= 1/S_j\left([m_j+1]e^1_{A_{y=j}} + [m_j+1]e^2_{A_{y=j}} + ... + [m_j+1]e^{m_j/2}_{A_{y=j}}\right) \\ &= (m_j+1)/S_j\left(e^1_{A_{y=j}} + e^2_{A_{y=j}} + ... + e^{m_j/2}_{A_{y=j}}\right) = (m_j+1)/S_j \sum_{i=1}^{m_j/2} e^i_{A_{y=j}}. \end{aligned}$$

And since the sums of errors for class j in A and B are equal, we get that:

$$MAE_{A_{y=j}} = (m_j+1)/S_j \sum_{i=1}^{m_j/2} e^i_{A_{y=j}} = (m_j+1)/S_j \sum_{i=1}^{m_j/2} e^i_{B_{y=j}} = MAE_{B_{y=j}}.$$

□

4. *Error severity*: MAE tackles error severities by definition.
5. *Accuracy and information tradeoff*: Generally, a tradeoff is a balancing of factors, all of which are not attainable at the same time. In our case, we see a tradeoff as balancing between two measures with opposite trends, e.g., one increases, and the other decreases. In ranges where the measures do not demonstrate such a relation, they show no tradeoff. In Table 13, we stated that the MAE does not trade accuracy and information. To show that, we first assume by negation that there is a tradeoff between them. If the MAE balances between accuracy and the MI, then in ranges where one increases while the other decreases, we expect the MAE to be monotonic with one of them, but with a smaller change. We will check the corresponding changes and show that this is not the case.

Lemma 4 *The MAE does not balance between accuracy and information.*

Proof Let A be a confusion matrix of size M that holds zero information (i.e., representing a random classifier showing a uniform error distribution per class), and let B be a confusion matrix of a classifier trained over the same data, but with a single change from A. According to the information theory, B holds more information than A, i.e., $MI_B > MI_A$.

There could be three types of change A has undergone:

(i) Moving samples between two (on- or) off-diagonal cells in A and B.
(ii) Moving samples from an off-diagonal cell in A to a diagonal cell in B.

Table 39 With correspondence to Eq. (21), an example of a single change between two $M \times M$ confusion matrices (a) A and (b) B, in which c samples of class M that were correctly predicted in this class ($x = M$) in A are now wrongly predicted in class 1 ($x = 1$) in B (recall that each true class in A is uniformly distributed) (Color table online)

(a) A

		True Class (Y)			
Predicted Class (X)		C_1	C_2	...	C_M
	C_1	**a**	**b**		**c**
	C_2	a	b		c
	...				
	C_M	**a**	**b**		**c**

(b) B

		True Class (Y)			
Predicted Class (X)		C_1	C_2	...	C_M
	C_1	**a**	**b**		**2c**
	C_2	a	b		c
	...				
	C_M	**a**	**b**		**0**

(iii) Moving samples from a diagonal cell in A to an off-diagonal cell in B.

□

Since MI increases when moving from A to B following a single change, we are interested in cases in which accuracy decreases for this change, i.e., cases that demonstrate a tradeoff between the two measures. In the first two cases, there is no tradeoff since accuracy does not decrease. In the third case, however, accuracy decreases and, therefore, there is a potential for a tradeoff between accuracy and the MI. If the MAE trades between the two measures, we expect the change in its value to account for the opposite trends in both measures and not only for one of them.

Let us denote in k the number of samples of true class $y = j$ that moved from predicted class $x = j$ to predicted class $x = i$ in a single change when moving from A to B (i.e., samples predicted as j in A and as i in B), n the total number of samples, and s the change in severity $|j - i|$ due to the move. Thus, when moving between A and B, the accuracy due to this single change decreases by k/n, and the MAE increases by $(ks)/n$.

To prove the lemma, we will show that although the MAE is monotone with the MI, the MAE's change is higher than the MI's change, which means accuracy did not reduce the MAE, and there is no balance between the MI and accuracy.

We first compute element-wise the change in the MI due to a single change in moving between confusion matrices A and B:

$$\begin{aligned}
\Delta MI = &- \sum_{x=i,j} \sum_{y} P_A(x,y) \log\left(\frac{P_A(x,y)}{P_A(x)P_A(y)}\right) \\
&+ \sum_{y \neq j} P_B(x=i,y) \log\left(\frac{P_B(x=i,y)}{P_B(x=i)P_B(y)}\right) \\
&+ \sum_{y \neq j} P_B(x=j,y) \log\left(\frac{P_B(x=j,y)}{P_B(x=j)P_B(y)}\right) \\
&+ P_B(x=i,y=j) \log\left(\frac{P_B(x=i,y=j)}{P_B(x=i)P_B(y=j)}\right) \\
&+ P_B(x=j,y=j) \log\left(\frac{P_B(x=j,y=j)}{P_B(x=j)P_B(y=j)}\right).
\end{aligned} \tag{21}$$

Since ΔMI between A and B is only due to the change in elements of the i*th* and j*th* rows (predicted classes), we can calculate ΔMI by first removing the MI's contribution of these

rows in A—the first term in Eq. (21), in red in Table 39(a) in which $i = 1$ and $i = M$. Second, we add the MI contribution of these rows in B—the second and third terms in Eq. (21) for all true classes but the j*th* one, in green in Table 39(b), and the fourth and fifth terms in Eq. (21) for the j*th* true class, in blue in Table 39(b) in which $j = M$, respectively.

Note that the first term in Eq. (21) is canceled off since A is uniformly distributed per class and, thus, $log\left(\frac{P(x,y)}{(1/M)(MP(x,y))}\right) = 0\ \forall x, y$, and the fifth term is canceled off because $P_B(x = j, y = j) = 0$ (all samples of class j that were correctly classified as j in A are classified as i in B).

The highest value MI can take due to the change between A and B is when $k = nP(y_j)/M$ (i.e., all samples of class j that were classified as j in A are classified as i in B).

Since: (1) $P_B(x = i) = P_A(x = i) + P(k) = 1/M + k/n$, (2) $P_B(y = j) = Mk/n$, and (3) $P_B(x = i, y = j) = 2k/n$, the fourth term in Eq. (21) can be written as:

$$\frac{2k}{n} log\left(\frac{(2k)/n}{(1/M + k/n)(Mk/n)}\right) = \frac{2k}{n} log\left(\frac{2n}{n + Mk}\right)$$

Next, since: (1) $P_B(x, y) = P_A(x, y)\ \forall x, \forall y \neq j$, and (2) $P_B(y) = P_A(y) = MP_A(x = i, y) = MP_A(x = j, y)\ \forall y$, the sum of the second and third terms in Eq. (21) can be written as

$$\begin{aligned}
&\sum_{y \neq j} P_A(x = i, y) log\left(\frac{P_A(x = i, y)}{(1/M + k/n)MP_A(x = i, y)}\right) \\
&\quad + \sum_{y \neq j} P_A(x = j, y) log\left(\frac{P_A(x = j, y)}{(1/M - k/n)MP_A(x = j, y)}\right) \\
&= \sum_{y \neq j} P_A(x = i, y) log\left(\frac{n}{n + Mk}\right) \\
&\quad + \sum_{y \neq j} P_A(x = j, y) log\left(\frac{n}{n - Mk}\right) \\
&= \sum_{y \neq j} P_A(x = i, y) log\left(\frac{n^2}{n^2 - (Mk)^2}\right).
\end{aligned}$$

And now Eq. (21) can be written as

$$\Delta MI = \frac{2k}{n} log\left(\frac{2n}{n + Mk}\right) + \sum_{y \neq j} P_A(x = i, y) log\left(\frac{n^2}{n^2 - (Mk)^2}\right) \tag{22}$$

To prove the lemma, we need to show that $\Delta MAE = ks/n$ is larger than ΔMI (Eq. (22)), which will contradict the assumption that the MAE lies between accuracy and the MI in a range where both measures have opposite trends. The two cases to consider are for the highest and lowest vales k can take:

(i) $k \to n/M$: In this case, (almost) all samples are from class j, which is distributed uniformly in A, leading to the highest value of k samples moved from A to B. The first term in Eq. (22) goes to $log(2n/2n) = 0$, and the second term also goes to zero

because $P_A(x = i, y \neq j) \rightarrow 0$, as there are almost no samples of classes other than j [note we use the convention that $0log(0/0) = 0$Cover and Thomas (2012)], and thus $\Delta MI \rightarrow 0$. Since $\Delta MAE \rightarrow s/M$, $\Delta MAE > \Delta MI$.

(ii) $k \rightarrow 1$: In this case, we move down to the minimal number of samples, which is $k = 1$. For $n \gg M$, the first term in Eq. (22) goes to $(2k)/n$, as $log(2) = 1$, the second term goes to zero, and thus $\Delta MI \rightarrow 2/n$. Since $\Delta MAE \rightarrow s/n$, for $s > 2$, $\Delta MAE > \Delta MI$.[7]

□

Mutual Information

1. *Class imbalance*: According to Table 13, mutual information (MI) is sensitive to class imbalance. We prove that (Lemma 5) by showing that the MI bounds are sensitive to class imbalance, and if the bounds are sensitive to the balance between classes, then also the measure is.

Lemma 5 *Two confusion matrices with M classes and the same number of samples have different MI bounds if the balance between classes is different.*

Proof Let A and B be two confusion matrices with M classes, and let P_A and P_B be two probability distributions of $P(Y)$ in A and B, respectively (i.e., two class proportions). Note, that we do not consider here two reverse distributions as different (e.g., $P_A = \{a, b, c\}$ and $P_B = \{c, b, a\}$). We prove by showing that the bounds of MI are different between A and B if $P_A \neq P_B$. We examine the lower and upper bounds:

(i) For both A and B, the lower bound of MI is zero. This value is obtained when each class is uniformly distributed with respect to X. Therefore, we only have to show that there is a difference between the upper bounds of A and B.

(ii) For both A and B, the upper bound of MI is achieved for a perfect classification (i.e., all off diagonal entries are zero). Thus, the non-diagonal elements in Eq. (6) are canceled off, and since in this case $P(x, y) = P(x) = P(y)$, the upper bound is a function of $P(y)$:

$$MI = \sum_{x=y} \sum_{y} P(x, y) log\left(\frac{P(x, y)}{P(x)P(y)}\right) = \sum_{y} P(y) log\left(\frac{1}{P(y)}\right), \tag{23}$$

Because Eq. (23) is a strictly convex function (Cover and Thomas 2012), the upper bounds of MI (and thus also its values) for A and B are different if $P_A \neq P_B$.

□

[7] For $s = 2$, $\Delta MAE \rightarrow 2/n$ from above ($k > 1$ leads to $\Delta MAE > 2/n$), and $\Delta MI \rightarrow 2/n$ from below ($k > 1$ —more than a single sample is moved from A to B—leads to the first log in Eq. (22) to decrease faster than the increase in $2k/n$), i.e., $\Delta MI <$2/n), so the inequality holds also for $s = 2$. For $s = 1$, there is no meaning to the severity error, and this is the binary case, where MAE is replaced by the accuracy.

Table 40 Example of two flipped confusion matrices with the same MI (Color table online)

(a)

Predicted Class (X) \ True Class (Y)	C_1	C_2	C_3
C_1	0	b	b
C_2	b	0	b
C_3	b	b	0

(b)

Predicted Class (X) \ True Class (Y)	C_1	C_2	C_3
C_1	b	b	0
C_2	b	0	b
C_3	0	b	b

Table 41 Example of two confusion matrices with reverse error distribution for C_4 and the same MI (0.316) (Color table online)

(a)

Predicted Class (X) \ True Class (Y)	C_1	C_2	C_3	C_4
C_1	80	50	30	100
C_2	70	150	200	50
C_3	60	10	90	0
C_4	20	10	10	100

(b)

Predicted Class (X) \ True Class (Y)	C_1	C_2	C_3	C_4
C_1	80	50	30	0
C_2	70	150	200	50
C_3	60	10	90	100
C_4	20	10	10	100

2. *Number of classes*: Table 13 indicates that MI is sensitive to the number of classes. Again, we prove (Lemma 6) that MI is sensitive to the number of classes by showing that MI bounds are.

Lemma 6 *Two confusion matrices with a different number of classes have different MI bounds.*

Proof We prove that by showing that MI bounds are different for the two matrices. Let A and B be two confusion matrices with M_A and M_B classes ($M_A \neq M_B$). In general, the minimal MI value a confusion matrix can take is when all samples are uniformly distributed across the matrix, i.e., $MI = \sum_x \sum_y 1/M^2 log \frac{1/M^2}{1/M \cdot 1/M} = 0$, a value that is independent of M. The maximal value MI can take is when all samples are uniformly distributed across the diagonal, i.e., $MI = \sum_x 1/M log \frac{1/M}{1/M \cdot 1/M} = log(M)$. Since $M_A \neq M_B$ also $log(M_A) \neq log(M_B)$, i.e., different upper bounds to M_A and M_B. □

3. *Error distribution*: According to Table 13, MI is only partially affected by the error distribution. A simple case that demonstrates MI insensitivity to error distribution is a flipped confusion matrix. For example, the error distributions of class C_3 (in red) in Tables 40(a, b) (or those of class C_1) are different although the MI of the two confusion matrices is equal ($log(9/6) = 0.585$).

4. *Error severity* Table 13 indicates that MI is only partially affected by the error severity. In order to demonstrate the MI insensitivity to the total error severity, ES, we also have to change the error distribution because it is the interrelation of the error distribution and severity of error ($|x - y|$) that is expressed in ES (Eq. (16)). Consider the example

in Table 41 that shows two confusion matrices with the same entries except those of class C_4 (highlighted in red). These matrices demonstrate different error distributions for class $j = 4$, leading to harsher total error severity for Table 41(a) than for Table 41(b), yet they have the same MI score. A measure that pretends to account for error severity cannot score both Tables 41(a, b) equally.

In Lemma 7, we prove that MI is insensitive to error severity for the general (harsh) case, where the error distributions are reversed. For this lemma, we require that the total number of predictions for classes other than j with respect to true classes other than j are symmetrical. To demonstrate this, let's denote S_k as the total number of predictions for classes other than j for the k predicted class. For example, in Table 41, the sum of the first and third rows (without C_4) are $S_1 = 80 + 50 + 30 = 160$ and $S_3 = 60 + 10 + 90 = 160$, respectively (we arrange all $S_x \ \forall x \neq j$ in a vector $S = \{S_1, S_2, S_3\}$). We will show that MI is insensitive in the case where the errors of class j are reversed, and $S = \{S_1, \ldots, S_{M-1}\}$ is symmetrical, where $S_x = \sum_{y \neq j} P(A_{x,y}), \forall x \neq j$ (and similarly for B).

Lemma 7 *Two confusion matrices of two classifiers learned from the same data and with reverse error distributions for class j have the same MI if all their non-j entries are element-wise equal and $S_x = S_{M-x}$ $\forall x \neq j$ for both matrices.*

Proof Let A and B be two confusion matrices with M classes. Assume that the errors of class j in A and B have a reverse distribution, $S_x = S_{M-x}$ $\forall x \neq j$ (see C_4 in Table 41 for an example).

The MI of A can be written as:

$$MI(A) = \sum_{x \neq j} P(A_{x,y=j}) log\left(\frac{P(A_{x,y=j})}{P(A_x)P(A_{y=j})}\right) + \sum_{y} P(A_{x=j,y}) log\left(\frac{P(A_{x=j,y})}{P(A_{x=j})P(A_y)}\right) + \sum_{x \neq j} \sum_{y \neq j} P(A_{x,y}) log\left(\frac{P(A_{x,y})}{P(A_x)P(A_y)}\right). \tag{24}$$

The first term in Eq. (24) is the sum over all class predictions for true class j, the second term is the sum for predictions of class j over all true classes, and the third term is the sum over predictions for all classes other than j when the true classes are other than j. For example, in Table 41(a), the first term refers to all elements that in red, the second term to the elements of the fourth row, and the third term to all elements except those of the fourth row and the fourth column. We further develop the three terms of Eqs. (24) in (28) (the first term), (25) (the second term), and (27) (the third term).

The second term of Equation (24): Because the following class marginal probabilities of A and B are equal, $P(A_y) = P(B_y), \ \forall y$, and $P(A_{x=j}) = P(B_{x=j})$, we write this term as:

$$\sum_{y} P(A_{x=j,y}) log\left(\frac{P(A_{x=j,y})}{P(A_{x=j})P(A_y)}\right) = \sum_{y} P(B_{x=j,y}) log\left(\frac{P(B_{x=j,y})}{P(B_{x=j})P(B_y)}\right). \tag{25}$$

The third term of Equation (24) can be written as:

$$\sum_{x\neq j}\sum_{y\neq j} P(A_{x,y})\big(logP(A_{x,y}) - logP(A_x) - logP(A_y)\big).$$

First, since A and B are element-wise equal for $y \neq j$, $P(A_{x,y})logP(A_{x,y}) = P(B_{x,y})logP(B_{x,y})$ for $y \neq j$. Second, as above, $P(A_y) = P(B_y)$, $\forall y$. Third, since $P(A_{x,y=j}) = P(B_{M-x,y=j})$ for $x \neq j$ (the reverse distribution assumption), and, based on the lemma assumption:

$$\sum_{y\neq j} A_{x,y} = \sum_{y\neq j} A_{M-x,y}, \quad \forall\, x \neq j, \tag{26}$$

then $P(A_x) = P(B_{M-x})$. Therefore, the third term of Eq. (24) is:

$$\sum_{x\neq j}\sum_{y\neq j} P(B_{x,y})\big(logP(B_{x,y}) - logP(B_{M-x}) - logP(B_y)\big) = \sum_{x\neq j}\sum_{y\neq j} P(B_{x,y})log\left(\frac{P(B_{x,y})}{P(B_x)P(B_y)}\right), \tag{27}$$

where the last equality is due to the lemma definition, leading to $\sum_{y\neq j} P(B_{x,y})logP(B_{M-x}) = \sum_{y\neq j} P(B_{x,y})logP(B_x)$.

The first term of Equation (24): Since, as above, $P(A_x) = P(B_{M-x})$ and $P(A_{x,y=j}) = P(B_{M-x,y=j})$, the first term of Eq. (24) is:

$$\begin{aligned}\sum_{x\neq j} P(A_{x,y=j})log\left(\frac{P(A_{x,y=j})}{P(A_x)P(A_{y=j})}\right) &= \sum_{x\neq j} P(B_{M-x,y=j})log\left(\frac{P(B_{M-x,y=j})}{P(B_{M-x})P(B_{y=j})}\right)\\ &= \sum_{x\neq j} P(B_{x,y=j})log\left(\frac{P(B_{x,y=j})}{P(B_x)P(B_{y=j})}\right),\end{aligned} \tag{28}$$

where the last equality is between two sums over the same elements in different order.

Since we showed in Eqs. (25), (27), and (28) that in each term of Eq. (24) the element corresponding to A can be replaced with that corresponding to B, we get that $MI(A) = MI(B)$. □

Information measure

In this section, we refer to both the information measure (IM) and IM_α. Since IM and IM_α are a combination of MI and a variation of MAE, they are both sensitive to the same properties which either MI or MAE are sensitive to. Thus, IM and IM_α are sensitive to class imbalance, number of classes, and balance between accuracy and information (due to the MI part), and to error severity (due to the ES part of IM and IM_α). Although both MI and MAE are insensitive to the error distribution under several conditions, these conditions do not overlap, and thus IM and IM_α are sensitive to the error distribution.

Confusion entropy

Confusion entropy (CEN) showed poor results, and thus it is omitted from this theoretical analysis. Empirical results, similar to those in Sect. 5, showed that CEN is insensitive to

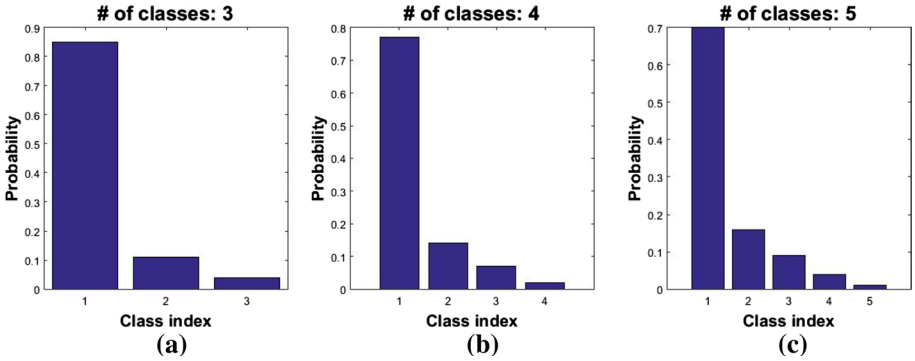

Fig. 13 Target variable distributions for a given parent combination and different number of classes

class imbalance, number of classes, and error severity, but a theoretical proof for this is not in the scope of this study.

Artificial BN sampling

The implementation to learn a BN—the structure and conditional probability table (CPT) parameters—was aided by the BNT (Murphy 2001) and SLP (Leray and Francois 2004) toolboxes. All CPTs (except that of the target variable) were sampled from a Dirichlet distribution with a parameter $\alpha = [1, 1, 1]$ (Geiger and Heckerman 1997; Ide and Cozman 2002). To perform a sensitivity analysis, we have to control the target variable (hence we cannot use the Dirichlet distribution). For each combination of the target variable parents, the target variable is sampled from the following distribution:

$$f(x) = x^3 + x, \quad 0 < x < 1.112 \tag{29}$$

where x is a continuous random variable.

We chose this function for several reasons: A polynomial of order three suits our purposes since it has a small area under the curve of high values of $f(x)$; however, other areas are not negligible, and the addition of x to x^3 increases the lowest probabilities to improve the representation of classes corresponding to low x.

For each combination of parents, we sampled X 10,000 times using decomposition Suzuki (1990) and created a histogram with bins as the number of classes of target variable. Figure 13 shows the distribution (in a discrete form) for three Fig. 13a, four Fig. 13b and five Fig. 13c class scenarios. It is important to maintain the same distribution so we could argue later that the differences in performance are due to changes in the number of classes and not due to the conditional probabilities.

IM scores for artificial databases

Table 42 shows the average IM scores achieved by the seven algorithms initialized by the empty graph. Recall that the IM scores are calculated according to Eq. (16) without normalization (as opposed to the IM_α scores that are normalized in order to compare different

Table 42 Mean (std $\times 10^{-1}$) IM values of BNCs learned using seven measures and the RMCV algorithm that is initialized by the empty graph for 23 artificial databases

DB	IM	IM_α	MI	CEN	MCC	MAE	ACC
1	*– 1.009 (1)*	**– 1.010 (1)**	**– 1.010 (1)**	*– 1.009 (1)*	*– 1.009 (1)*	**– 1.010 (1)**	**– 1.010 (1)**
2	– 1.012 (1)	**– 1.013 (1)**	– 1.007 (1)	*– 0.182 (8)*	– 1.007 (1)	– 1.009 (1)	– 1.004 (1)
3	– 0.896 (1)	**– 0.928 (1)**	– 0.833 (2)	*0.116 (6)*	– 0.877 (1)	– 0.913 (1)	– 0.891 (1)
4	– 0.851 (1)	**– 0.865 (1)**	– 0.831 (1)	*0.325 (7)*	– 0.844 (1)	– 0.806 (2)	– 0.850 (1)
5	– 0.732 (2)	**– 0.801 (2)**	– 0.694 (3)	*0.681 (6)*	– 0.769 (2)	– 0.718 (2)	– 0.776 (2)
6	– 0.699 (1)	**– 0.721 (1)**	– 0.676 (1)	*0.825 (7)*	– 0.696 (1)	– 0.611 (2)	– 0.678 (1)
7	– 0.644 (1)	**– 0.679 (1)**	– 0.582 (2)	*0.825 (5)*	– 0.608 (2)	– 0.511 (3)	– 0.605 (2)
8	– 0.577 (2)	**– 0.624 (1)**	– 0.433 (3)	*1.090 (7)*	– 0.551 (1)	– 0.261 (5)	– 0.550 (1)
9	– 0.592 (2)	**– 0.617 (2)**	– 0.533 (2)	*0.191 (5)*	– 0.549 (2)	– 0.592 (2)	– 0.583 (2)
10	– 0.747 (2)	– 0.803 (1)	– 0.682 (2)	*0.088 (5)*	– 0.763 (2)	– 0.754 (2)	**– 0.804 (1)**
11	– 0.773 (2)	**– 0.824 (1)**	– 0.789 (2)	*0.267 (6)*	– 0.792 (2)	– 0.808 (1)	– 0.787 (1)
12	– 0.896 (1)	**– 0.928 (1)**	– 0.833 (2)	*0.116 (6)*	– 0.877 (1)	– 0.913 (1)	– 0.891 (1)
13	– 0.923 (1)	**– 0.948 (1)**	– 0.947 (1)	*0.186 (5)*	– 0.926 (1)	– 0.937 (1)	– 0.921 (1)
14	– 0.997 (1)	**– 0.999 (1)**	– 0.981 (1)	*0.220 (6)*	– 0.977 (1)	– 0.982 (1)	– 0.974 (1)
15	– 0.903 (1)	**– 0.907 (1)**	– 0.795 (3)	*0.474 (6)*	– 0.888 (1)	– 0.861 (2)	– 0.883 (1)
16	– 0.859 (1)	**– 0.864 (1)**	– 0.831 (1)	*0.356 (5)*	– 0.834 (1)	– 0.852 (1)	– 0.840 (1)
17	– 0.708 (1)	**– 0.717 (1)**	– 0.635 (2)	*0.672 (3)*	– 0.660 (2)	– 0.691 (1)	– 0.690 (1)
18	– 0.621 (2)	**– 0.652 (1)**	– 0.516 (3)	*0.651 (2)*	– 0.365 (3)	– 0.543 (2)	– 0.576 (2)
19	– 0.563 (1)	**– 0.583 (1)**	– 0.579 (1)	*0.650 (0)*	– 0.527 (2)	– 0.577 (1)	– 0.557 (1)
20	– 0.433 (2)	**– 0.480 (1)**	– 0.242 (3)	*0.561 (0)*	– 0.415 (2)	– 0.399 (2)	– 0.413 (2)
21	– 0.074 (3)	– 0.031 (3)	**– 0.141 (2)**	*0.383 (0)*	– 0.068 (2)	0.093 (3)	0.156 (3)
22	0.143 (2)	0.166 (2)	**0.042 (2)**	*0.331 (0)*	0.081 (2)	0.263 (2)	0.280 (1)
23	0.175 (1)	0.190 (0)	**0.133 (1)**	*0.195 (0)*	0.135 (1)	*0.195 (0)*	*0.195 (0)*
Avg (std)	– 0.660 (0)	**– 0.680 (0)**	– 0.626 (0)	*0.348 (0)*	– 0.643 (0)	– 0.617 (0)	– 0.637 (0)

values of alphas). IM_α has the best average IM score. The differences between the measures are similar to those achieved for ACC and IM_α as was seen in Sect. 6.1.

Run time measured by number of neighbors for artificial BNs

In Table 43, we analyze the time complexity of each algorithm by counting the number of neighbor graphs examined during the learning phase. The poor results of CEN with respect to the seven measures are compensated by a short run time. This makes sense due to the small number of iterations of the algorithm. On the other hand, the IM_α-based BNC suffers from the worse time complexity since it is a wrapper algorithm.

Table 43 Mean $\times 10^2$ (std $\times 10^2$) run time (measured by number of neighbors) of BNCs learned using seven measures and the RMCV algorithm that is initialized by the empty graph for 23 artificial databases

DB	IM	IM_α	MI	CEN	MCC	MAE	ACC
1	1.7 (2)	*10.3 (12)*	1.7 (2)	1.7 (2)	1.7 (2)	**1.6 (2)**	**1.6 (2)**
2	12.3 (4)	*69.6 (18)*	12.5 (5)	**4.9 (5)**	12.7 (4)	10.5 (3)	10.7 (3)
3	11.7 (5)	*69.2 (31)*	11.0 (5)	**2.4 (4)**	11.0 (5)	10.9 (3)	10.0 (4)
4	11.5 (5)	*59.8 (18)*	9.9 (4)	**2.1 (3)**	10.1 (3)	11.4 (4)	10.0 (3)
5	13.3 (6)	*78.1 (34)*	12.3 (6)	**1.0 (1)**	12.6 (4)	11.2 (5)	11.2 (4)
6	13.0 (4)	*73.0 (26)*	12.6 (4)	**1.2 (1)**	12.4 (4)	11.2 (4)	11.2 (3)
7	11.5 (4)	*63.5 (20)*	13.0 (6)	**1.0 (0)**	11.2 (4)	10.3 (5)	11.2 (4)
8	10.5 (4)	*69.8 (24)*	10.2 (5)	**1.0 (1)**	12.5 (4)	8.3 (6)	12.0 (3)
9	7.0 (3)	*43.9 (19)*	6.7 (3)	**1.4 (1)**	8.2 (4)	6.9 (2)	6.5 (3)
10	9.3 (4)	*60.2 (21)*	9.2 (5)	**1.5 (1)**	10.8 (5)	8.9 (3)	9.5 (3)
11	12.8 (7)	*71.2 (21)*	12.3 (5)	**1.5 (2)**	10.8 (4)	9.9 (3)	10.7 (4)
12	11.7 (5)	*69.2 (31)*	11.0 (5)	**2.4 (4)**	11.0 (5)	10.9 (3)	10.0 (4)
13	13.2 (5)	*83.9 (31)*	15.5 (6)	**1.8 (3)**	13.9 (5)	12.3 (4)	12.2 (4)
14	15.4 (5)	*90.3 (45)*	16.6 (7)	**2.2 (4)**	14.2 (5)	12.9 (4)	11.6 (4)
15	13.6 (5)	*82.3 (36)*	12.8 (7)	**1.0 (1)**	13.6 (5)	11.3 (4)	12.7 (4)
16	13.0 (4)	*87.2 (28)*	12.8 (4)	**1.0 (1)**	13.0 (4)	12.5 (3)	12.5 (4)
17	11.7 (6)	*72.0 (28)*	12.7 (6)	**0.7 (0)**	13.0 (6)	10.8 (4)	10.9 (4)
18	15.9 (5)	*97.5 (48)*	13.9 (7)	**0.4 (0)**	9.6 (7)	12.3 (5)	13.9 (6)
19	13.3 (5)	*83.3 (25)*	13.3 (4)	**0.4 (0)**	12.5 (5)	14.7 (5)	12.2 (4)
20	15.8 (8)	*92.0 (31)*	9.9 (8)	**0.4 (0)**	14.7 (8)	13.4 (5)	12.3 (5)
21	9.6 (7)	*45.4 (41)*	12.4 (7)	**0.4 (0)**	8.9 (6)	4.6 (5)	4.5 (6)
22	4.5 (6)	*13.1 (21)*	6.4 (6)	**0.4 (0)**	5.5 (5)	1.9 (4)	1.2 (2)
23	0.9 (2)	*2.3 (0)*	1.5 (2)	**0.4 (0)**	1.9 (4)	**0.4 (0)**	**0.4 (0)**
Avg (std)	11.0 (4)	*66.3 (28)*	10.9 (4)	**1.4 (1)**	10.7 (4)	9.5 (4)	9.5 (4)

IM scores for UCI databases

Table 44 shows the average IM scores (again, not normalized) achieved by the seven algorithms initialized by the NBC graph.

Run Time measured by number of neighbors for UCI BNs

In Table 45, we analyze the time complexity of each algorithm by counting the number of neighbor graphs examined during the learning phase. This table is consistent with Table 43.

Table 44 Mean (std $\times 10^{-1}$) IM values of BNCs learned using seven measures and the RMCV algorithm that is initialized by the NBC graph for 17 UCI and real-world databases

DB	IM	IM_α	MI	CEN	MCC	MAE	ACC
1	0.030 (1)	**0.028 (1)**	**0.028 (0)**	*0.059 (3)*	0.030 (0)	**0.028 (1)**	0.031 (0)
2	**− 0.274 (1)**	**− 0.274 (1)**	**− 0.274 (1)**	*− 0.263 (1)*	**− 0.274 (1)**	**− 0.274 (1)**	**− 0.274 (1)**
3	− 0.820 (2)	− 0.829 (2)	− 0.795 (2)	*− 0.794 (2)*	− 0.824 (2)	− 0.824 (2)	**− 0.836 (2)**
4	**− 0.614 (2)**	− 0.609 (2)	− 0.580 (2)	− 0.590 (2)	*− 0.578 (2)*	− 0.596 (2)	− 0.603 (2)
5	− 0.889 (1)	**− 0.891 (1)**	*− 0.861 (1)*	− 0.890 (1)	− 0.888 (1)	− 0.877 (1)	− 0.867 (1)
6	− 0.135 (1)	− 0.135 (1)	− 0.135 (1)	*− 0.132 (1)*	− 0.135 (1)	**− 0.140 (1)**	**− 0.140 (1)**
7	**− 0.954 (1)**	**− 0.954 (1)**	**− 0.954 (1)**	**− 0.954 (1)**	*− 0.928 (2)*	**− 0.954 (1)**	**− 0.954 (1)**
8	**0.026 (2)**	**0.026 (2)**	**0.026 (2)**	*0.219 (3)*	**0.026 (2)**	0.027 (2)	0.027 (2)
9	− 0.043 (3)	*− 0.035 (2)*	− 0.068 (3)	− 0.036 (2)	**− 0.096 (3)**	− 0.065 (2)	− 0.065 (2)
10	− 0.602 (2)	− 0.609 (2)	**− 0.613 (2)**	*− 0.580 (2)*	− 0.588 (2)	− 0.597 (2)	− 0.606 (2)
11	− 0.202 (0)	**− 0.205 (0)**	− 0.202 (0)	− 0.203 (0)	*0.043 (1)*	− 0.201 (0)	− 0.186 (0)
12	− 0.457 (3)	− 0.461 (3)	− 0.457 (3)	*− 0.455 (3)*	**− 0.474 (3)**	− 0.461 (3)	− 0.461 (3)
13	0.130 (0)	**0.127 (0)**	0.128 (0)	*0.138 (0)*	0.131 (0)	0.130 (0)	0.130 (0)
14	**− 0.998 (0)**	**− 0.998 (0)**	**− 0.998 (0)**	**− 0.998 (0)**	*− 0.996 (0)*	**− 0.998 (0)**	**− 0.998 (0)**
15	− 0.908 (0)	− 0.908 (0)	**− 0.910 (0)**	*− 0.905 (0)*	− 0.907 (0)	*− 0.905 (0)*	− 0.907 (0)
16	− 1.393 (1)	− 1.400 (1)	− 1.393 (1)	− 1.389 (1)	**− 1.418 (1)**	− 1.393 (1)	*− 1.388 (1)*
17	**− 0.695 (2)**	**− 0.695 (2)**	**− 0.695 (2)**	**− 0.695 (2)**	− 0.682 (2)	*− 0.680 (2)*	*− 0.680 (2)*
Avg (std)	− 0.517 (0)	**− 0.519 (0)**	− 0.515 (0)	*− 0.498 (0)*	− 0.503 (0)	− 0.516 (0)	− 0.516 (0)

Table 45 Mean $\times 10^2$ (std $\times 10^2$) run time (measured by number of neighbors) of BNCs learned using seven measures and the RMCV algorithm that is initialized by the NBC graph for 17 real-world and UCI databases

DB	IM	IM_α	MI	CEN	MCC	MAE	ACC
1	126.1 (48)	*634.4 (37)*	**120.3 (24)**	146.4 (10)	166.1 (46)	134.4 (23)	139.2 (31)
2	14.1 (4)	*70.5 (6)*	14.1 (4)	13.2 (4)	14.1 (4)	**11.9 (2)**	**11.9 (2)**
3	2.6 (1)	*16.9 (1)*	2.6 (1)	3.0 (1)	3.1 (1)	**2.4 (1)**	2.5 (1)
4	21.4 (5)	*141.3 (7)*	20.2 (6)	20.2 (4)	22.5 (5)	**19.0 (3)**	19.7 (4)
5	2.6 (1)	*21.5 (2)*	**2.5 (1)**	2.8 (1)	2.7 (1)	2.8 (1)	2.6 (1)
6	6.4 (3)	*57.8 (4)*	6.4 (3)	6.8 (2)	6.4 (3)	**5.2 (2)**	**5.2 (2)**
7	1.4 (0)	*14.3 (0)*	1.4 (0)	1.4 (0)	**1.4 (0)**	1.4 (0)	1.4 (0)
8	2.0 (1)	*22.5 (1)*	2.0 (1)	2.5 (1)	2.0 (1)	**2.0 (1)**	**2.0 (1)**
9	17.0 (4)	*100.8 (6)*	17.6 (5)	17.2 (4)	16.6 (4)	**15.9 (3)**	**15.9 (3)**
10	1.5 (0)	*10.0 (1)*	1.5 (0)	1.9 (1)	1.6 (0)	1.6 (0)	**1.4 (0)**
11	67.6 (14)	*410.5 (20)*	67.8 (15)	**58.3 (13)**	66.3 (27)	72.3 (15)	72.5 (12)
12	7.4 (5)	*44.9 (7)*	7.4 (5)	**7.4 (5)**	7.7 (5)	7.5 (5)	7.5 (5)
13	29.4 (8)	*185.4 (12)*	30.9 (12)	**26.3 (2)**	30.9 (12)	28.7 (7)	27.2 (6)
14	12.9 (2)	*77.5 (2)*	12.9 (2)	12.9 (2)	**11.9 (2)**	12.9 (2)	12.9 (2)
15	3.5 (1)	*20.4 (1)*	3.8 (1)	3.2 (1)	3.3 (1)	3.1 (1)	**2.9 (1)**
16	5.2 (2)	*36.8 (2)*	5.0 (2)	**4.7 (2)**	6.4 (2)	5.3 (2)	5.7 (2)
17	13.8 (3)	*82.5 (5)*	13.8 (3)	14.1 (3)	13.6 (3)	**13.2 (3)**	**13.2 (3)**
Avg (std)	19.7 (32)	*114.6 (166)*	**19.4 (31)**	20.1 (35)	22.2 (40)	20.0 (34)	20.2 (35)

References

Agresti, A. (2011). *An introduction to categorical data analysis*. Berlin: Springer.

Baccianella, S., Esuli, A., & Sebastiani, F. (2009). Evaluation measures for ordinal regression. In *Proceedings of the ninth international conference on intelligent systems design and applications* (pp. 283–287). IEEE.

Baldi, P., Brunak, S., Chauvin, Y., Andersen, C. A., & Nielsen, H. (2000). Assessing the accuracy of prediction algorithms for classification: An overview. *Bioinformatics*, *16*(5), 412–424.

Breiman, L., Friedman, J., Stone, C. J., & Olshen, R. A. (1984). *Classification and regression trees*. Boca Raton: CRC Press.

Brodersen, K. H., Ong, C. S., Stephan, K. E., & Buhmann, J. M. (2010). The balanced accuracy and its posterior distribution. In *Proceedings of the 20th international conference on pattern recognition* (pp. 3121–3124). IEEE.

Brooks, B. R., Sanjack, M., Ringel, S., England, J., Brinkmann, J., Pestronk, A., et al. (1996). The amyotrophic lateral sclerosis functional rating scale-assessment of activities of daily living in patients with amyotrophic lateral sclerosis. *Archives of Neurology*, *53*(2), 141–147.

Caballero, J. C. F., Martínez, F. J., Hervás, C., & Gutiérrez, P. A. (2010). Sensitivity versus accuracy in multiclass problems using memetic pareto evolutionary neural networks. *IEEE Transactions on Neural Networks*, *21*(5), 750–770.

Chang, C. C., & Lin, C. J. (2011). LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology,2*(3):27, http://www.csie.ntu.edu.tw/~cjlin/libsvm

Chawla, N., Bowyer, K., Hall, L., & Kegelmeyer, W. (2002). SMOTE: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, *16*, 321–357.

Chawla, N. V. (2005). Data mining for imbalanced datasets: An overview. In *Data mining and knowledge discovery handbook* (pp. 853–867).

Cooper, G. F., & Herskovits, E. (1992). A Bayesian method for the induction of probabilistic networks from data. *Machine Learning*, *9*(4), 309–347.

Cover, T. M., & Thomas, J. A. (2012). *Elements of information theory*. New York: Wiley.

Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *The Journal of Machine Learning Research*, *7*, 1–30.

Domingos, P. (1999). Metacost: A general method for making classifiers cost-sensitive. In *Proceedings of the fifth international conference on knowledge discovery and data mining (KDD'99)* (pp. 155–164).

Duin, R., Juszczak, P., Paclik, P., Pekalska, E., Ridder, D. D., Tax, D. M. J., & Verzakov, S. (2000). PRTools: A Matlab toolbox for pattern recognition. version 3, http://www.prtools.org

Elkan, C. (2001). The foundations of cost-sensitive learning. *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence*, *17*, 973–978.

Fawcett, T. (2006). An introduction to ROC analysis. *Pattern Recognition Letters*, *27*(8), 861–874.

Ferri, C., Hernández-Orallo, H., & Modroiu, R. (2009). An experimental comparison of performance measures for classification. *Pattern Recognition Letters*, *30*(1), 27–38.

Frank, E., & Hall, M. (2001). A simple approach to ordinal classification. In *Proceedings of the 12th European conference on machine learning* (pp. 145–156). Springer.

Friedman, N., Geiger, D., & Goldszmidt, M. (1997). Bayesian network classifiers. *Machine Learning*, *29*(2–3), 131–163.

Galar, M., A Fernandez, E. B., Bustince, H., & Herrera, F. (2012). A review on ensembles for the class imbalance problem: Bagging-, boosting-, and hybrid-based approaches. *IEEE Transactions on Systems, Man, and Cybernetics: Part C—Applications and Reviews*, *42*(4), 463–484.

García, S., Fernández, A., Luengo, J., & Herrera, F. (2009). A study of statistical techniques and performance measures for genetics-based machine learning: Accuracy and interpretability. *Soft Computing*, *13*(10), 959–977.

García, V., Mollineda, R. A., & Sanchez, J. S. (2010). Theoretical analysis of a performance measure for imbalanced data. In *Proceedings of the 20th international conference on pattern recognition* (pp. 617–620). IEEE.

Geiger, D., & Heckerman, D. (1997). A characterization of the Dirichlet distribution through global and local parameter independence. *The Annals of Statistics*, *25*(3), 1344–1369.

Gordon, J., & Lerner, B. (2019). Insights into ALS from a machine learning perspective. *Journal of Clinical Medicine, 8*(10), 1578.

Gorodkin, J. (2004). Comparing two k-category assignments by a k-category correlation coefficient. *Computational Biology and Chemistry*, *28*(5), 367–374.

Grossman, D., & Domingos, P. (2004). Learning Bayesian network classifiers by maximizing conditional likelihood. In *Proceedings of the twenty-first international conference on machine learning* (pp 361–368). ACM.
Halbersberg, D., & Lerner, B. (2016). Learning a Bayesian network classifier by jointly maximizing accuracy and information. In *Proceedings of the 22nd European conference on artificial intelligence* (pp. 1638–1639). IOS Press.
Halbersberg, D., & Lerner, B. (2019). Young driver fatal motorcycle accident analysis by jointly maximizing accuracy and information. *Accident Analysis and Prevention*, *129*, 350–361.
Hand, D. J., & Till, R. J. (2001). A simple generalisation of the area under the ROC curve for multiple class classification problems. *Machine Learning*, *45*(2), 171–186.
Heckerman, D. (1998). A tutorial on learning with Bayesian networks. In *Learning in graphical models* (pp 301–354). Springer.
Heckerman, D., Geiger, D., & Chickering, D. (1995). Learning Bayesian networks: The combination of knowledge and statistical data. *Machine Learning*, *20*, 197–243.
Hyndman, R. J., & Koehler, A. B. (2006). Another look at measures of forecast accuracy. *International Journal of Forecasting*, *22*(4), 679–688.
Ide, J. S., & Cozman, F. G. (2002). Random generation of Bayesian networks. In *Advances in artificial intelligence* (pp. 366–376). Springer.
Jurman, G., Riccadonna, S., & Furlanello, C. (2012). A comparison of MCC and CEN error measures in multi-class prediction. *PLoS ONE*, *7*(8), e41882.
Kelner, R., & Lerner, B. (2012). Learning Bayesian network classifiers by risk minimization. *International Journal of Approximate Reasoning*, *53*(2), 248–272.
Kiernan, M., Vucic, S., Cheah, B., Turner, M., & Eisen, A. (2011). Amyotrophic lateral sclerosis. *Lancet*, *377*, 942–955.
Kontkanen, P., Myllymäki, P., Silander, T., & Tirri, H. (1999). On supervised selection of bayesian networks. *In Proceedings of the fifteenth conference on uncertainty in artificial intelligence* (pp. 334–342). Morgan Kaufmann Publishers Inc.
Labatut, V., & Cherifi, H. (2011). Accuracy measures for the comparison of classifiers. In *Proceedings of the fifth international conference on information technology*, ICIT.
Lam, W., & Bacchus, F. (1994). Learning Bayesian belief networks: An approach based on the MDL principle. *Computational Intelligence*, *10*(3), 269–293.
Leray, P., & Francois, O. (2004). *BNT structure learning package: Documentation and experiments*. Tech Rep: Laboratoire PSI.
Lerner, B., Yeshaya, J., & Koushnir, L. (2007). On the classification of a small imbalanced cytogenetic image database. *IEEE Transactions on Computational Biology and Bioinformatics*, *4*(2), 204–215.
Lichman, M. (2013). UCI machine learning repository. http://archive.ics.uci.edu/ml
Liu, X. Y., Wu, J., & Zhou, Z. H. (2009). Exploratory undersampling for class-imbalance learning. *IEEE Transactions on Systems, Man, and Cybernetics: Part B—Cybernetics*, *39*(2), 539–550.
Mitchell, D., & Borasio, G. (2007). Amyotrophic lateral sclerosis. *Lancet*, *33*, 51–59.
Murphy, K. (2001). The Bayes net toolbox for Matlab. *Computing Science and Statistics*, *33*(2), 1024–1034.
OECD. (2006). Young drivers: The road to safety. Organization for Economic Co-operation and Development.
Piccareta, R. (2008). Classification trees for ordinal variables. *Computational Statistics*, *23*(20), 407–427.
Provost, F. (2000). Machine learning from imbalanced data sets. In *Proceedings of the AAAI workshop on imbalanced data sets* (pp. 1–3).
Provost, F. J., Fawcett, T., & Kohavi, R. (1998). The case against accuracy estimation for comparing induction algorithms. *Proceedings of the Fifteenth International Conference on Machine Learning*, *98*, 445–453.
Ranawana, R., & Palade, V. (2006). Optimized precision—A new measure for classifier performance evaluation. In *IEEE Congress on evolutionary computation* (pp. 2254–2261). IEEE.
Sokolova, M., & Lapalme, G. (2009). A systematic analysis of performance measures for classification tasks. *Information Processing and Management*, *45*(4), 427–437.
Suzuki, M. (1990). Fractal decomposition of exponential operators with applications to many-body theories and Monte Carlo simulations. *Physics Letters A*, *146*(6), 319–323.
Toledo, T., Lotan, T., Taubman-Ben-Ari, O., & Grimberg, E. (2012). Evaluation of a program to enhance young drivers' safety in Israel. *Accident Analysis & Prevention*, *45*, 705–710.
Wallace, C. S., & Boulton, D. M. (1968). An information measure for classification. *The Computer Journal*, *11*(2), 185–194.

Wasikowski, M., & Chen, X. W. (2010). Combating the small sample class imbalance problem using feature selection. *IEEE Transactions on Knowledge and Data Engineering*, *22*(10), 1388–1400.
Wei, J. M., Yuan, X. J., Hu, Q. H., & Wang, S. Q. (2010). A novel measure for evaluating classifiers. *Expert Systems with Applications*, *37*(5), 3799–3809.

Affiliations

Dan Halbersberg[1] · Maydan Wienreb[1] · Boaz Lerner[1]

Maydan Wienreb
maydanw@gmail.com

Boaz Lerner
boaz@bgu.ac.il

[1] Ben-Gurion University of the Negev, 84105 Beer-Sheva, Israel