Probabilistic Multi-Dimensional Classification (Supplementary Material)

Vu-Linh Nguyen[*1] **Yang Yang**[*2] **Cassio de Campos**[3]

[1]Heudiasyc Laboratory, University of Technology of Compiègne, France
[2]Department of Computer Science, KU Leuven, Belgium
[3]Eindhoven University of Technology, The Netherlands

A NOTATION AND ACRONYMS

Table 5: Notation and acronyms

symbol/acronym	meaning
$\mathcal{X}$, $\boldsymbol{x}$	instance space, instance
$\mathcal{Y}$, $\boldsymbol{y}$	output space, outcome
X^p, Y^k	feature, class variable
K, Q	number of class variables, number of features
$[\![\cdot]\!]$	indicator function
$[n]$	set $\{1,\dots,n\}$ of natural numbers
$\boldsymbol{p}(\boldsymbol{y}\,\vert\,\boldsymbol{x})$	probability of outcome $\boldsymbol{y}$ given $\boldsymbol{x}$
$\boldsymbol{p}(y^k\,\vert\,\boldsymbol{x})$	marginal probability of relevance for outcome $Y^k = y^k$ given $\boldsymbol{x}$
$\mathcal{D}$	training data
ℓ, ℓ_S, ℓ_L	MDC loss function, Subset 0/1 loss, Hamming loss
G, θ	Structure (i.e., a DAG) of a BN, Parameter set of a BN
Δ_G^Y	Parent set of Y in G
Π_d^Y	Set with all possible configurations of the discrete parents of Y
π	Configuration of the parents of a variable, stored as pairs (variable, value)
$\mathcal{G}$	Set of the $R(K+Q)$ possible DAGs
$\mathcal{G}^1$	Set of the $R(K)2^{KQ}R(Q)$ DAGs which contain no edge of the form $Y \longrightarrow X$
$\mathcal{G}^2$	Set of the $R(K)2^{KQ}$ DAGs, whose elements contain no edge between features
$\mathcal{G}^3$	Set of the $R(K)2^{K\vert\mathbf{X}_d\vert})$ DAGs such that, $\forall G \in \mathcal{G}^3$ and $\forall Y \in \mathbf{Y}$, we have $\mathbf{X}_c \subset \Delta^Y$
DAG	Directed acyclic graph
MDC	Multi-dimensional classification
BN	Bayesian network
BOP	Bayes-optimal prediction
CP	class powerset
CCs	classifier chains
BR	binary relevance

[*]These authors contributed equally to this work.

B PROOFS OF PROPOSITIONS

This section presents proofs for the propositions stated in the main paper. When it is necessary, we recall related notions and results in the literature before presenting proofs.

B.1 PROPOSITION 3.1

We first present Lemmas which are necessary to complete the proof of Proposition 3.1.

Lemma B.1. *Assume there is a $G' \in \mathcal{G}$ such that G' is an I-map for $\boldsymbol{p} \in \mathcal{P}^0$. All the I-maps G of $\boldsymbol{p}$ induce the same CLL score* (4).

Proof. Reminding that conditional joint probability distribution is

$$\boldsymbol{p}(\boldsymbol{y} \,|\, \boldsymbol{x}) := \frac{\boldsymbol{p}(\boldsymbol{x}, \boldsymbol{y})}{\sum_{\boldsymbol{y} \in \mathcal{Y}} \boldsymbol{p}(\boldsymbol{x}, \boldsymbol{y})}\,, \forall (\boldsymbol{x}, \boldsymbol{y}) \in \mathcal{X} \times \mathcal{Y}\,. \tag{29}$$

Assume $G \in \mathcal{G}$ is an I-map of $\boldsymbol{p}$. Because of that, we have $\boldsymbol{p}(\boldsymbol{x}, \boldsymbol{y}) = \boldsymbol{p}_\theta^G(\boldsymbol{x}, \boldsymbol{y})$ for a certain θ, with factorization respecting G. This implies

$$\boldsymbol{p}(\boldsymbol{y} \,|\, \boldsymbol{x}) = \frac{\boldsymbol{p}(\boldsymbol{x}, \boldsymbol{y})}{\sum_{\boldsymbol{y}' \in \mathcal{Y}} \boldsymbol{p}(\boldsymbol{x}, \boldsymbol{y}')} = \frac{\boldsymbol{p}_\theta^G(\boldsymbol{x}, \boldsymbol{y})}{\sum_{\boldsymbol{y}' \in \mathcal{Y}} \boldsymbol{p}_\theta^G(\boldsymbol{x}, \boldsymbol{y}')} = \boldsymbol{p}_\theta^G(\boldsymbol{y} \,|\, \boldsymbol{x})\,, \forall (\boldsymbol{x}, \boldsymbol{y}) \in \mathcal{X} \times \mathcal{Y}\,. \tag{30}$$

Then it is clear that

$$C(\boldsymbol{p} \,|\, \mathcal{D}) = \log \prod_{n=1}^{N} \boldsymbol{p}(\boldsymbol{y}_n \,|\, \boldsymbol{x}_n) = \log \prod_{n=1}^{N} \boldsymbol{p}_\theta^G(\boldsymbol{y}_n \,|\, \boldsymbol{x}_n) = C(\boldsymbol{p}_\theta^G \,|\, \mathcal{D})\,.$$

Thus, all the I-maps G of $\boldsymbol{p}$ have the same CLL score on $\mathcal{D}$. □

Lemma B.2. *Assume elements of* $\mathbf{X}$ *are always made available. Assume there is a $G \in \mathcal{G}$ such that G is an I-map for $\boldsymbol{p} \in \mathcal{P}^0$. Then, there is at least one $G' \in \mathcal{G}^1$ which is an I-map for $\boldsymbol{p}$.*

Proof. As long as the chain rule of probability is valid, we can lazily pick up any topological ordering $t'(1), \ldots, t'(K+Q)$ on $\mathbf{Z} = \mathbf{X} \cup \mathbf{Y}$ in which the Q features occupy the first Q places (and the K class variables occupy the next K places) and add arcs from each feature/variable to the ones succeed it until having a fully connected DAG G'. It is clear that $G' \in \mathcal{G}^1$ because we never add any arc of the form $Y \longrightarrow X$. Moreover, G' is an I-map[1] of G (and $\boldsymbol{p}$) because $\mathcal{I}(G') = \emptyset$.

□

In the following, we present a proof of Proposition 3.1.

Proof. There is always an I-map $G \in \mathcal{G}$ of $\boldsymbol{p} \in \mathcal{P}^0$ which maximizes the CLL function (4) on $\mathcal{D}$ (if chain rule of probability applies, as we can use a full graph). Lemma B.1 tells us that all the I-maps G of $\boldsymbol{p}$ maximize the CLL function. Lemma B.2 tells us that at least one of the I-maps belongs to $\mathcal{G}^1$. Hence, there is at least one I-map $G' \in \mathcal{G}^1$ which maximizes the CLL function (4) on $\mathcal{D}$. Or, equivalently, the relation (8) holds. □

[1]While this is already enough to complete the proof, fully connected DAGs are not really the goal of learning BNs. Sparser I-maps can be easily constructed by only adding arcs which preserve conditional dependencies when following the ordering. Also, during the execution of the main algorithms that we use, we naturally find small graphs because of the penalizations that are used (similar guarantees as those that exist for learning BNs).

B.2 PROPOSITION 3.2

Proof. The fact that, for any $G \in \mathcal{G}^1$, the joint conditional distribution (1) can be factorized as

$$\boldsymbol{p}_\theta^G(\boldsymbol{y} \,|\, \boldsymbol{x}) = \prod_{Y \in \mathbf{Y}} \boldsymbol{p}_\theta\left(y \,|\, \pi_y\right) \,, \forall (\boldsymbol{x}, \boldsymbol{y}) \in \mathcal{X} \times \mathcal{Y} \,.$$

can be checked easily. Since $Y \notin \Delta^X$, for any $Y \in \mathbf{Y}$ and for any $X \in \mathbf{X}$, we have

$$\begin{aligned}
\boldsymbol{p}_\theta^G(\boldsymbol{y} \,|\, \boldsymbol{x}) &= \frac{\boldsymbol{p}_\theta^G(\boldsymbol{x}, \boldsymbol{y})}{\sum_{\boldsymbol{y}' \in \mathcal{Y}} \boldsymbol{p}_\theta^G(\boldsymbol{x}, \boldsymbol{y}')} = \frac{\prod_{X \in \mathbf{X}} \boldsymbol{p}_\theta(x \,|\, \pi_x) \prod_{Y \in \mathbf{Y}} \boldsymbol{p}_\theta(y \,|\, \pi_y)}{\sum_{\boldsymbol{y}'} \prod_{X \in \mathbf{X}} \boldsymbol{p}_\theta(x \,|\, \pi_x) \prod_{Y \in \mathbf{Y}} \boldsymbol{p}_\theta(y' \,|\, \pi_{y'})} \\
&= \frac{\prod_{X \in \mathbf{X}} \boldsymbol{p}_\theta(x \,|\, \pi_x) \prod_{Y \in \mathbf{Y}} \boldsymbol{p}_\theta(y \,|\, \pi_y)}{\prod_{X \in \mathbf{X}} \boldsymbol{p}_\theta(x \,|\, \pi_x) \sum_{\boldsymbol{y}'} \prod_{Y \in \mathbf{Y}} \boldsymbol{p}_\theta(y' \,|\, \pi_{y'})} = \frac{\prod_{Y \in \mathbf{Y}} \boldsymbol{p}_\theta(y \,|\, \pi_y)}{\sum_{\boldsymbol{y}'} \prod_{Y \in \mathbf{Y}} \boldsymbol{p}_\theta(y' \,|\, \pi_{y'})} \qquad (31) \\
&= \prod_{Y \in \mathbf{Y}} \boldsymbol{p}_\theta(y \,|\, \pi_y) \,. \qquad (32)
\end{aligned}$$

The transition from (31) to (32) is straightforward because, by the definition of BNs, we have

$$\sum_{\boldsymbol{y}'} \prod_{Y \in \mathbf{Y}} \boldsymbol{p}_\theta(y' \,|\, \pi_{y'}) = 1 \,. \qquad (33)$$

We now prove that following relation holds:

$$\max_{(G,\theta) \in \mathcal{P}^1} C(\boldsymbol{p}_\theta^G \,|\, \mathcal{D}) = \max_{(G,\theta) \in \mathcal{P}^2} C(\boldsymbol{p}_\theta^G \,|\, \mathcal{D}) \,,$$

We first partition $\mathcal{G}^1$ into $R(K) 2^{KQ}$ groups where each group consists of $R(Q)$ DAGs whose edges among $\mathbf{Y}$ and edges from features to class variables are the same. The relation

$$\boldsymbol{p}_\theta^G(\boldsymbol{y} \,|\, \boldsymbol{x}) = \prod_{Y \in \mathbf{Y}} \boldsymbol{p}_\theta\left(y \,|\, \pi_y\right) \,, \forall (\boldsymbol{x}, \boldsymbol{y}) \in \mathcal{X} \times \mathcal{Y}$$

ensures that all the members of each group have the same CLL score. Moreover, each group contains exactly one member of $\mathcal{G}^2$, i.e., the DAG with no edge among features. Therefore, the maximal CLL score attained over $\mathcal{G}^1$ equals the maximal score attained over $\mathcal{G}^2$. □

B.3 PROPOSITION 3.3

Proof of Proposition 3.3 is trivial and is written down for completeness.

Proof. For any $G \in \mathcal{P}^2$, there is at least one I-map $G' \in \mathcal{P}^3$ (to see that, simply add the extra arcs to G to complete the parent sets of any class variable with all the continuous feature variables, leading to a graph $G' \in \mathcal{P}^3$ – adding arcs will keep the I-map property). Thus, Proposition 3.3 comes as a consequence of Proposition 3.2. □

B.4 COROLLARY 3.4

We would like to re-emphasize that our assumptions of having the optimality of learned parameters in the local models are not too strong. These are much weaker assumptions than those one finds in the literature when investigating the optimality of PGM learning frameworks: that is typically the assumption that the hypothesis space contains the possible distributions from some given family and the best estimate(s) converge to the optimal distribution(s) asymptotically.

We do not require any asymptotic results, and the requirement of optimally learned parameters (given data) can be met by many standard estimation methods. Yet, this cannot be always guaranteed in practice, in particular if someone decides to use complicated models connecting the input feature variables and class variables, so our assumption is necessary for the proof of global optimality of the framework (which is a strong result and obviously cannot be achieved if base local models are not optimal themselves).

With this in mind, the following (short) proof would satisfactorily inform readers of the significance of the proposed framework regarding the optimality.

Proof. Assume the chain rule of probability holds (which is arguably a mild assumption) and the parameter learning problem is optimally solved. As a combination of Proposition 3.1, Proposition 3.2 and Proposition 3.3, we have

$$\max_{\boldsymbol{p}\in\mathcal{P}^0} C(\boldsymbol{p}\,|\,\mathcal{D}) = \max_{(G,\theta)\in\mathcal{P}} C(\boldsymbol{p}_\theta^G\,|\,\mathcal{D}) = \max_{(G,\theta)\in\mathcal{P}^1} C(\boldsymbol{p}_\theta^G\,|\,\mathcal{D}) = \max_{(G,\theta)\in\mathcal{P}^2} C(\boldsymbol{p}_\theta^G\,|\,\mathcal{D}) = \max_{(G,\theta)\in\mathcal{P}^3} C(\boldsymbol{p}_\theta^G\,|\,\mathcal{D})\,.$$

Thus, algorithm 1 should return an I-map of the optimal distribution in $\mathcal{P}^0$. In other words, the learning procedure is universal, as (G^*, θ^*) is optimal with respect to $\mathcal{P}$, and with enough data would match the true conditional $\boldsymbol{p}(\mathbf{Y}|\mathbf{X})$ for $\boldsymbol{p}$ in $\mathcal{P}^0$. □

B.5 PROPOSITION 3.5

Enlarging parent sets (with discrete features) in our setting is analogous to further partitioning the input space in local supervised learning parts [Wang and Saligrama, 2012]. A representative of such approaches is the top-down construction of decision trees [Landwehr et al., 2005, Rokach and Maimon, 2005]. In such approaches, it is well-known that further partitioning the input space leads to higher predictive performance on the training data sets [Landwehr et al., 2005, Rokach and Maimon, 2005], as long as they are optimally learned. We can expect a similar phenomenon in our setting because CLL (4) is indeed a performance measure for our probabilistic classifiers, and the way we encode each local distribution using $|\Pi_d^Y|$ distributions $\boldsymbol{p}_\theta\,(Y\,|\,\pi, \mathbf{X}_c)$, $\forall \pi \in \Pi_d^Y$, makes our approach an input space partitioning approach, where $\pi \in \Pi_d^Y$ are used to partition the space formed by $\mathbf{X}_c$.

We now present a proof for Proposition 3.5.

Proof. Let $\boldsymbol{p}_{Y,\pi}$ be the local models used for Y with parent set Δ, for $\pi \in \Pi_d^Y$, and $\boldsymbol{p}'_{Y,\pi'}$ be the local models for parent set $\Delta' \supset \Delta$ such that $\pi \subset \pi'$. Let θ be the optimal parameters used by $\boldsymbol{p}_{Y,\pi}$. Because the local models remain as models from continuous features $\mathbf{X}_c$ to the class variable Y, θ is still a valid solution (albeit non optimal) of the parameter learning of $\boldsymbol{p}'_{Y,\pi'}$, for each π' extending π. If we use such a θ and sum together the CLL of all $\boldsymbol{p}'_{Y,\pi'}$ with $\pi' \supset \pi$ (that is, the extended parent set configurations that are compatible with π over the variables they have in common), then we achieve the very same score (4). Repeating this for all extension of all $\pi \in \Pi_d^Y$, the same overall CLL score is reached. This means that the CLL obtained after the added parents in $\Delta' \setminus \Delta$ has to be equal or larger (as it is assumed to be optimally learned) than before adding the parents. This proves Inequality (21). Now, (21) guarantees that enlarging parent sets cannot decrease the CLL score (4). This ensures that at least one solution of the Algorithm 1 is a fully connected BN in the sense that the DAG over the class variables induced by its structure G is fully connected. Such a solution can be found by finding a topological order of a solution G from Algorithm 1 and then adding arcs to G (respecting that topological order) until the DAG over the class variables induced by the structure G is fully connected (side comment: obviously it is not our goal to have fully connected networks, this is just to proof the theoretical results). □

C DETAILED ALGORITHMS

C.1 ALGORITHM 1

In this section, we show that Algorithm 1 can be revised to find a GBNC $(G^*, \theta^*) \in \mathcal{P}^3$ of any regularized variant (22) of the CLL function. We first compute $C(\boldsymbol{p}_{\theta^*_{Y,\pi}}\,|\,Y, \pi, \mathcal{D})$ from $\mathcal{D}$ by solving (17) in line 5 of Algorithm 1, which can be done by extracting the data set

$$\mathcal{D}_\pi := \{(\boldsymbol{x}_n, \boldsymbol{y}_n) \in \mathcal{D}\,|\,\pi_{\boldsymbol{y}_n} = \pi\} \tag{34}$$

and calling any base learner to learn the optimal parameter set of $\boldsymbol{p}_\theta\,(Y\,|\,\pi, \mathbf{X}_c)$ on $\mathcal{D}_\pi$ with respect to (17).

For any given regularized variant (22) of the CLL function, we denote by

$$S(Y, \Delta_d^Y) = C(Y, \Delta_d^Y) - \text{pen}(\Delta_d^Y, |\mathcal{D}|)\,. \tag{35}$$

Clearly, the problem of learning a best $G \in \mathcal{G}^3$ can be re-expressed as an Integer Programming (IP):

$$\begin{aligned}
\text{Maximize} & \sum_{Y \in \mathbf{Y}} \sum_{\Delta_d^Y \in \mathcal{F}^Y} \gamma(\Delta_d^Y) \cdot S(Y, \Delta_d^Y)\,, \qquad (36)\\
\text{Subject to} & \sum_{\Delta_d^Y \in \mathcal{F}^Y} \gamma(\Delta_d^Y) = 1\,, \forall Y \in \mathbf{Y}\,,\\
& \sum_{Y \in \mathbf{Y}'} \sum_{\substack{\Delta_d^Y \in \mathcal{F}^Y \\ \Delta_d^Y \cap \mathbf{Y}' = \emptyset}} \gamma(\Delta_d^Y) > 1\,, \forall \mathbf{Y}' \subseteq \mathbf{Y}\,, |\mathbf{Y}'| > 1\,,\\
& \gamma(\Delta_d^Y) \in \{0,1\}\,, \forall Y \in \mathbf{Y}, \forall, \Delta_d^Y \in \mathcal{F}^Y\,.
\end{aligned}$$

Altogether, we end up with the implementation given in Algorithm 2, which returns a GBNC $(G^*, \theta^*) \in \mathcal{P}^3$ of (12).

Algorithm 2 Learning a GBNC of (12) under the presence of regularization

```
 1: Input: Data $\mathcal{D}$, Probabilistic hypothesis spaces encoding $\boldsymbol{p}_\theta(Y \mid \pi, \mathbf{X}_c)$, $\forall \pi \in \Pi_d^Y$, $\forall \Delta_d^Y \in \mathcal{F}^Y$, $\forall Y \in \mathbf{Y}$
 2: for $Y \in \mathbf{Y}$ do
 3:   for $\Delta_d^Y \in \mathcal{F}^Y$ do
 4:     for $\pi \in \Pi_d^Y$ do
 5:       Solve (17) and store it in a proper data structure
 6:     end for
 7:     Compute $S(Y, \Delta_d^Y)$ by (35) using stored values
 8:   end for
 9: end for
10: Find a best collection $\{\Delta_d^Y \mid Y \in \mathbf{Y}\}$ which optimizes (36) using GOBNILP
11: Output: A GBNC $(G^*, \theta^*) \in \mathcal{P}^3$ of (12)
```

C.2 A REFINEMENT OF ALGORITHM 1

In this section, we show how pruning rules [de Campos et al., 2018] can be employed to find GBNCs which optimize regularized variants (22) without losing any optimality.

We first generalize the pruning rule [de Campos et al., 2018][Lemma 3] for any regularized variant of the form (22).

Lemma C.1. *Let $Y \in \mathbf{Y}$ be a node in $G \in \mathcal{G}^3$ with $\Delta \subset \Delta' \in \mathcal{F}^Y$, such that*

$$S(Y, \Delta) \geq -pen(\Delta', |\mathcal{D}|)\,. \qquad (37)$$

Then Δ' and all its supersets can be safely discarded from $\mathcal{F}^Y$ without decreasing the maximum score of (12).

Proof. Using the shorthand notation (35), a regularized variant of the CLL function can be rewritten as

$$S(\boldsymbol{p}_{\theta^*}^G \mid \mathcal{D}) = \sum_{Y \in \mathbf{Y}} S(Y, \Delta_d^Y) = \sum_{Y \in \mathbf{Y}} \left(C(Y, \Delta_d^Y) - \text{pen}(\Delta_d^Y, |\mathcal{D}|) \right)\,.$$

For any $G, G' \in \mathcal{G}^3$ such that Δ and Δ' are respectively the parent set of Y in G and G', and the parent sets of all $Y' \neq Y$ are the same, we have the relation

$$\begin{aligned}
S(\boldsymbol{p}_{\theta^*}^G \mid \mathcal{D}) - S(\boldsymbol{p}_{\theta^*}^{G'} \mid \mathcal{D}) &= S(Y, \Delta) - S(Y, \Delta') = S(Y, \Delta) - C(Y, \Delta) + \text{pen}(\Delta', |\mathcal{D}|)\\
&\geq -\text{pen}(\Delta', |\mathcal{D}|) - C(Y, \Delta) + \text{pen}(\Delta', |\mathcal{D}|) = -C(Y, \Delta) \geq 0\,.
\end{aligned}$$

Thus, we can safely discard Δ' from $\mathcal{F}^Y$ without decreasing the maximum score of (12).

Because for any $\Delta' \subset \Delta'' \in \mathcal{F}^Y$, we have $-\text{pen}(\Delta', |\mathcal{D}|) \geq -\text{pen}(\Delta'', |\mathcal{D}|)$ and assumption (21) ensures that $C(Y, \Delta) \leq C(Y, \Delta'')$. Thus, we have the relation

$$S(Y, \Delta) \geq -\text{pen}(\Delta', |\mathcal{D}|) \geq -\text{pen}(\Delta'', |\mathcal{D}|)\,.$$

which ensures that we can also safely discard Δ'' from $\mathcal{F}^Y$. $\square$

Intuitively, Lemma C.1 provides us a "stopping criterion" for enlarging parent sets by exploiting the fact that regularized variants (22) of the CLL (4) seek for a trade-off between the predictive performance provided by more complex classifiers and the simplicity of classifiers. More precisely, condition (37) allows one to safely discard (some/many large) possible parent set Δ' and its supersets Δ'' without the need of learning local probabilistic classifiers (15) for these parent sets. It is very beneficial, because if we do not have such a stopping criterion, we will need to evaluate all the possible parent sets and evaluating each $\Delta^Y \in \mathcal{F}^Y$ requires one to the learning a possibly large number of local probabilistic classifiers (15) which is exponential in the cardinality of Δ^Y.

Ideally, we would expect that enlarging the parent sets (or increasing the model complexity) gives us a better score $S(Y, \Delta^Y)$, i.e., assumption (21) should hold. However, in practice, it may happen that the learning algorithm fails to converge and returns unreliable (and inaccurate) local probabilistic classifiers (15). In such a case, we would keep adding redundant parents and end up with unreliable local probabilistic classifiers (15) in the final GBNC. In other words, we pick up an unnecessary complex GBNC which contains unreliable local probabilistic classifiers (15). To avoid this unexpected behavior, we propose a variant of the pruning rule (37).

Definition C.2. Let $Y \in \mathbf{Y}$ be a node in $G \in \mathcal{G}^3$ with $\Delta \subset \Delta' \in \mathcal{F}^Y$, such that

$$\max_{\Delta'' \subset \Delta} S(Y, \Delta'') \geq -\text{pen}(\Delta', |\mathcal{D}|)\,. \tag{38}$$

Then all the $\Delta \supset \Delta^*$, where

$$\Delta^* = \arg\max_{\Delta'' \subset \Delta} S(Y, \Delta'')\,, \tag{39}$$

will be discarded from $\mathcal{F}^Y$.

Intuitively, the pruning rule (38) –(39) allows us to prune all the supersets of Δ^*. For example, if $\Delta^* \subsetneq \Delta$, we discard all of its supersets, such as Δ, Δ' and their supersets. Adopting the pruning rule (38) –(39), we propose a refinement of Algorithm 1 which is summarized in Algorithm 3. To simplify Pseudocode, for any $Y \in \mathbf{Y}$, we denote by

$$\mathcal{F}^Y_k = \left\{\Delta \in \mathcal{F}^Y \,|\, |\Delta| = k\right\}\ , \forall k = |\mathbf{X}_c|, \ldots, Q + K\,. \tag{40}$$

Algorithm 3 only learns a local classifier which estimates the local distributions $\boldsymbol{p}_\theta\left(Y \,|\, \pi, \mathbf{X}_c\right)$, $\pi \in \Pi$, if Δ is still included in $\mathcal{F}^Y$ and its complexity is not so high according to (38). In practice, we observed that large $\Delta \in \mathcal{F}^Y$ are usually discarded.

Algorithm 3 Learning a GBNC of (12) under the presence of regularization

```
1: Input: Data $\mathcal{D}$, Probabilistic hypothesis spaces encoding $\boldsymbol{p}_\theta\left(Y \,|\, \pi, \mathbf{X}_c\right), \forall \pi \in \Pi^Y_d, \forall \Delta^Y_d \in \mathcal{F}^Y, \forall Y \in \mathbf{Y}$
2: for $Y \in \mathbf{Y}$ do
3:   for $k = |\mathbf{Y}_c|, \ldots, Q + K$ do
4:     for $\Delta^Y_d \in \mathcal{F}^Y_k$ do
5:       if Condition (38) holds then
6:         Determine $\Delta^*$ using (39); Discard all the $\Delta^Y_d \supset \Delta^*$ from $\mathcal{F}^Y$
7:       else
8:         Compute $S(Y, \Delta^Y_d)$ defined in (35); Store $\boldsymbol{p}_{\theta^*}\left(Y \,|\, \pi, \mathbf{X}_c\right), \forall \pi \in \Pi^Y_d$ in a proper data structure
9:       end if
10:     end for
11:   end for
12: end for
13: Find a best collection $\{\Delta^Y_d \,|\, Y \in \mathbf{Y}\}$ which optimizes (36) using GOBNILP
14: Output: A GBNC $(G, \theta) \in \mathcal{P}^3$ of (12)
```

C.3 INFERENCE ALGORITHMS

Practical procedures for finding BOPs of ℓ_S (27) and ℓ_H (26) are presented in Algorithm 4 and Algorithm 5, respectively.

Algorithm 4 Find a BOP of the Subset $0/1$ loss (27)

```
1: Input: A GBNC $(G^*, \theta^*) \in \mathcal{P}^3$ of (12): $\boldsymbol{p}_{\theta^*}(Y \mid \pi, \mathbf{X}_c), \forall \pi \in \Pi_d^Y, \forall \Delta_d^Y \in \mathcal{F}^Y, \forall Y \in \mathbf{Y}$, a test instance $\boldsymbol{x}$
2: Extract the sub-DAG $\mathcal{K}$ over $\mathbf{Y}$ from $G$
3: for $Y \in \mathbf{Y}$ do
4:     Extract parent set of $Y$ in $\mathcal{K}$: $\Delta_{\mathbf{Y}}^Y = \Delta_d^Y \cap \mathbf{Y}$; Form the set $\Pi_{\mathbf{Y}}^Y$ of the possible configurations of $\Delta_{\mathbf{Y}}^Y$
5:     for $\pi_{\mathbf{Y}}^Y \in \Pi_{\mathbf{Y}}^Y$ do
6:         Predict $\boldsymbol{p}_{\theta^*}^{\mathcal{K}}\left(Y \mid \pi_{\mathbf{Y}}^Y\right)$ using $\boldsymbol{p}_{\theta^*}(Y \mid \pi, \mathbf{X}_c)$ which are specified by $\boldsymbol{x}_d$
7:     end for
8: end for
9: Find a MPE $\hat{\boldsymbol{y}} \in \mathcal{Y}$ given $\mathcal{K}$ and $\boldsymbol{p}_{\theta^*}^{\mathcal{K}}\left(Y \mid \pi_{\mathbf{Y}}^Y\right), \forall \pi_{\mathbf{Y}}^Y \in \Pi_{\mathbf{Y}}^Y, \forall Y \in \mathbf{Y}$
10: Output: A BOP $\hat{\boldsymbol{y}}$ of the Subset $0/1$ loss (27)
```

Algorithm 5 Find a BOP of the Hamming loss (26)

```
1: Input: A GBNC $(G^*, \theta^*) \in \mathcal{P}^3$ of (12): $\boldsymbol{p}_{\theta^*}(Y \mid \pi, \mathbf{X}_c), \forall \pi \in \Pi_d^Y, \forall \Delta_d^Y \in \mathcal{F}^Y, \forall Y \in \mathbf{Y}$, a test instance $\boldsymbol{x}$
2: Extract the sub-DAG $\mathcal{K}$ over $\mathbf{Y}$ from $G$
3: for $Y \in \mathbf{Y}$ do
4:     Extract parent set of $Y$ in $\mathcal{K}$: $\Delta_{\mathbf{Y}}^Y = \Delta_d^Y \cap \mathbf{Y}$; Form the set $\Pi_{\mathbf{Y}}^Y$ of the possible configurations of $\Delta_{\mathbf{Y}}^Y$
5:     for $\pi_{\mathbf{Y}}^Y \in \Pi_{\mathbf{Y}}^Y$ do
6:         Predict $\boldsymbol{p}_{\theta^*}^{\mathcal{K}}\left(Y \mid \pi_{\mathbf{Y}}^Y\right)$ using $\boldsymbol{p}_{\theta^*}(Y \mid \pi, \mathbf{X}_c)$ which are specified by $\boldsymbol{x}_d$
7:     end for
8: end for
9: Find $K$ marginals $\hat{y}^1, \ldots, \hat{y}^K$ given $\mathcal{K}$ and $\boldsymbol{p}_{\theta^*}^{\mathcal{K}}\left(Y \mid \pi_{\mathbf{Y}}^Y\right), \forall \pi_{\mathbf{Y}}^Y \in \Pi_{\mathbf{Y}}^Y, \forall Y \in \mathbf{Y}$
10: Output: A BOP $\hat{\boldsymbol{y}} := (\hat{y}^1, \ldots, \hat{y}^K)$ of the Hamming loss (26)
```

D THE CASE OF PARTIAL/MISSING DATA

The structural Expectation-Maximization (structural EM) approach has been used in different works in BN learning from missing data [Adel and de Campos, 2017, Rancoita et al., 2016, Friedman, 1998]. Reminding that, in BN learning with an incomplete training data, the structural EM approach [Friedman, 1998] can be employed to find a pair of a possible precise/complete data set and a possible BN, which optimizes some given target function.

The structural EM approach can be implemented as a two-step algorithm, which should be iterated until either the algorithm converges or some stopping criterion is met.

- Expectation step (E): we complete the data by imputing partial/missing data from a fitted BN;
- Maximization step (M): we learn a BN by optimizing given target function over the completed data.

Yet, we can in principle adapt the M step of the structural EM approach to the setting of probabilistic MDC straightforwardly.

However, depending on the concrete type of missing data we are dealing with, handling the E step may require more attention. In the case of partially specified class variables and precise features [Wang et al., 2021], GBNCs given by Algorithm 1 and 2 are estimates of $\boldsymbol{p}(\mathcal{Y}|\mathcal{X})$ and can be used to impute partial/missing data during the E step. In the general case where both the features and class variables can be partially specified [Hüllermeier, 2014, Nguyen et al., 2021], estimates of $\boldsymbol{p}(\mathcal{Y}|\mathcal{X})$ itself seems to be inadequate, because estimates of $\boldsymbol{p}(\mathcal{X}, \mathcal{Y})$ may be needed for doing imputation if one wishes to use exact/approximate inference. We however leave this problem as a future work because it is beyond the scope of this paper.

E EXPERIMENTS

E.1 EXPERIMENTAL SETTING

We evaluate our approaches on both tabular and image data sets. Table 6 summarizes the detailed statistics of all tabular data sets, which are originally collected by [Jia and Zhang, 2021]. From left to right, the meaning of each column is the number of class variables (#CV), the number of samples (#Samples), and the number of states of each class variable. (#States/CV) and the number of features (#Features), respectively. Among the 20 tabular data sets, there are three data sets (Adult, Default and Thyroid) which contain mixed features. If all class variables contain the same number of states, only this number is reported. For example, the Flickr data set has five class variables, which have 3, 4, 3, 4, and 4 states, respectively.

Table 6: Statistics of the tabular benchmark data sets.

Data Set	#CV	#States/CV	#Samples	#Features
Edm	2	3	154	$16n$
Jura	2	4,5	359	$9n$
Enb	2	2,4	768	$6n$
Voice	2	4,2	3136	$19n$
Song	3	3	785	$98n$
Adult	4	$7, 7, 5, 2$	18419	$5n, 5x$
Default	4	$2, 7, 4, 2$	28779	$14n, 6x$
Flickr	5	$3, 4, 3, 4, 4$	12198	$1536n$
Fera	5	6	14052	$136n$
WQplants	7	4	1060	$16n$
WQanimals	7	4	1060	$16n$
Thyroid	7	$5, 5, 3, 2, 4, 4, 3$	9172	$7n, 22x$
Rf1	8	$4, 4, 3, 4, 4, 3, 4, 3$	8987	$64n$
Pain	10	$2, 5, 4, 2, 2, 5, 2, 5, 2, 2$	9734	$136n$
Disfa	12	$5, 5, 6, 3, 4, 4, 5, 4, 4, 4, 6, 4$	13095	$136n$
WaterQuality	14	4	1060	$16n$
Oes97	16	3	334	$263n$
Oes10	16	3	403	$298n$
Scm20d	16	4	8966	$61n$
Scm1d	16	4	9803	$280n$

We compare two instantiations of GBNCs (GBNC-S which optimizes (22) and produces BOP (27) of ℓ_S, and GBNC-H which optimizes (22) and produces BOP (26) of ℓ_H) with BR, CP, and CCs [Jia and Zhang, 2021][Section II–III]. Reminding that pen$(\Delta_d^Y, |\mathcal{D}|)$ is the penalty term of the Bayesian Information Criterion (BIC) [Schwarz, 1978] in our experiments.

It is known that the chain order of CCs can (significantly) affect its performance and choosing the best order is one of the toughest problems in learning CCs [Read et al., 2021]. Although choosing good orders of CCs is not a focus of our work, randomly choosing orders would make CCs too weak. We thus sample 11 orders (which are the original order of the class variables from the data source and 10 other orders generated randomly) and pick the best chain order in terms of validation performance, i.e., we use 80% of the training data to learn CCs and pick up the most promising order with the highest performance on the validation set consists of 20% of training data, and report its test performance. When running the experiments, we observed that this often improves the performance of CCs, compared to randomly choosing one chain order. We follow the suggestion of [Jia and Zhang, 2021] and convert discrete features/variables into continuous variables using one-hot encoding whenever they appear as parts of input of local classifiers of BR, CP and CCs. While there are other multi-dimensional classifiers with promising predictive performances, such as [Jia and Zhang, 2021] and [Jia and Zhang, 2023], we find it hard to interpret such classifiers as probabilistic classifiers. Thus, we do not include them in our experimental comparison, which specifically focuses on probabilistic classifiers.

For each tabular data set, we do a 10-fold cross-validation, and report the mean and standard deviation of the performance of the classifiers. For the image data set, we do a 3-fold cross-validation, and report the mean and standard deviation of the performance of the classifiers.

We implement all approaches in Python and use the pgmpy framework [Ankan and Panda, 2015]. We use the PyTorch framework [Paszke et al., 2019] to implement neural networks. The source code to replicate experiments is provided as supplementary materials and has been made made public at `https://github.com/yangyang-pro/probabilistic-mdc`.

E.2 RESULTS

This appendix provides detailed experimental results which are summarized in section 5 of the main text.

Hamming losses and their ranks provided by the classifiers are given in table 7 and 8. Subset 0/1 losses and their ranks provided by the classifiers are given in table 9 and 10. Scatter plots for the losses provided by pairs of classifiers are given in Figure 2–5. Each black point illustrates losses provided by the classifiers labeled on the horizontal axis and the vertical axis on one data set. The differences provided by pairs of classifiers are illustrated by the horizontal distances between (black) points and the blue line $y = x$. Points lie on the left side of $y = x$ indicate that classifiers labeled on the horizontal axis are better than ones labeled on the vertical axis, and points lie on the right side of $y = x$ indicate that classifiers labeled on the vertical axis are better than ones labeled in the horizontal axis. Points lie far away from $y = x$ suggest visible differences.

Table 7: Hamming loss (mean ± std.) of each MDC approach (**base learner: *logisitic regression***).

Data Set	Hamming loss (in %)			
	BNCH	BR	CC	CP
Edm	**26.54 ± 9.57 (1.0)**	27.65 ± 7.95 (3.0)	27.23 ± 6.95 (2.0)	28.23 ± 8.10 (4.0)
Jura	37.32 ± 4.48 (2.0)	**35.51 ± 5.06 (1.0)**	39.81 ± 8.10 (3.0)	67.84 ± 7.23 (4.0)
Enb	**22.20 ± 4.59 (1.5)**	24.16 ± 4.44 (3.0)	**22.20 ± 5.29 (1.5)**	31.77 ± 1.89 (4.0)
Voice	8.21 ± 1.18 (3.0)	**8.08 ± 1.05 (1.5)**	8.08 ± 0.67 (1.5)	41.69 ± 1.30 (4.0)
Song	**24.33 ± 2.35 (1.0)**	25.74 ± 2.86 (2.0)	26.46 ± 5.49 (3.0)	49.30 ± 3.32 (4.0)
Adult	32.46 ± 0.77 (3.0)	**28.26 ± 0.47 (1.0)**	28.46 ± 0.42 (2.0)	41.40 ± 0.61 (4.0)
Default	**33.29 ± 0.42 (1.0)**	33.48 ± 0.59 (3.0)	33.39 ± 0.42 (2.0)	43.94 ± 0.31 (4.0)
Flickr	21.74 ± 0.57 (3.0)	**20.22 ± 0.69 (1.0)**	20.46 ± 0.47 (2.0)	49.21 ± 0.72 (4.0)
Fera	**37.76 ± 0.64 (1.0)**	38.82 ± 0.96 (2.0)	39.23 ± 0.70 (3.00)	52.97 ± 2.41 (4.0)
WQplants	**34.61 ± 1.67 (1.0)**	34.65 ± 2.18 (2.0)	35.42 ± 2.02 (3.0)	38.06 ± 3.05 (4.0)
WQanimals	**36.85 ± 1.35 (1.0)**	36.98 ± 1.97 (2.0)	38.23 ± 0.87 (3.0)	43.07 ± 2.68 (4.0)
Thyroid	**3.38 ± 0.14 (1.0)**	3.52 ± 0.19 (3.0)	3.44 ± 0.17 (2.0)	3.89 ± 0.16 (4.0)
Rf1	**9.53 ± 0.65 (1.0)**	16.10 ± 0.67 (2.0)	16.33 ± 0.42 (3.0)	36.49 ± 0.79 (4.0)
Pain	**4.70 ± 0.33 (1.0)**	4.74 ± 0.32 (2.0)	4.91 ± 0.37 (3.0)	5.24 ± 0.46 (4.0)
Disfa	**10.30 ± 0.36 (1.0)**	10.58 ± 0.37 (2.0)	10.63 ± 0.30 (3.0)	13.08 ± 0.60 (4.0)
WaterQuality	**35.53 ± 1.24 (1.0)**	36.10 ± 1.12 (2.0)	36.50 ± 0.95 (3.0)	40.92 ± 1.58 (4.0)
Oes97	**27.61 ± 1.41 (1.0)**	28.24 ± 1.68 (2.0)	29.35 ± 1.78 (3.0)	45.59 ± 3.36 (4.0)
Oes10	19.55 ± 1.80 (2.0)	**19.21 ± 1.98 (1.0)**	20.80 ± 1.73 (3.0)	38.40 ± 3.30 (4.0)
Scm20d	**31.45 ± 0.84 (1.0)**	36.04 ± 0.71 (2.0)	38.15 ± 0.96 (3.0)	57.87 ± 0.65 (4.0)
Scm1d	**18.07 ± 0.38 (1.0)**	23.42 ± 0.79 (2.0)	25.63 ± 0.95 (3.0)	55.59 ± 0.80 (4.0)
Ave. rank	**1.43**	1.98	2.60	4.00

References

Tameem Adel and Cassio P de Campos. Learning Bayesian networks with incomplete data by augmentation. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI)*, pages 1684–1690, 2017.

Ankur Ankan and Abinash Panda. pgmpy: Probabilistic graphical models using python. In *Proceedings of the 14th Python in Science Conference (SciPy)*, volume 10. Citeseer, 2015.

Cassio P de Campos, Mauro Scanagatta, Giorgio Corani, and Marco Zaffalon. Entropy-based pruning for learning Bayesian networks using BIC. *Artificial Intelligence*, 260:42–50, 2018.

Nir Friedman. The Bayesian structural EM algorithm. In *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 129–138, 1998.

Eyke Hüllermeier. Learning from imprecise and fuzzy observations: Data disambiguation through generalized loss minimization. *International Journal of Approximate Reasoning*, 55(7):1519–1534, 2014.

Bin-Bin Jia and Min-Ling Zhang. Decomposition-based classifier chains for multi-dimensional classification. *IEEE Transactions on Artificial Intelligence*, 3(2):176–191, 2021.

Bin-Bin Jia and Min-Ling Zhang. Multi-dimensional classification via decomposed label encoding. *IEEE Transactions on Knowledge and Data Engineering*, 35(2):1844–1856, 2023. doi: 10.1109/TKDE.2021.3100436.

Table 8: Hamming loss (mean ± std.) of each MDC approach (**base learner: *Naive Bayes***).

Data Set	Hamming loss (in %)			
	BNCH	BR	CC	CP
Edm	32.17 ± 6.55 (2.0)	33.08 ± 3.67 (3.0)	34.19 ± 7.64 (4.0)	**27.88 ± 7.41 (1.0)**
Jura	**43.01 ± 6.83 (1.0)**	45.40 ± 4.45 (3.0)	43.45 ± 4.45 (2.0)	69.06 ± 4.25 (4.0)
Enb	**22.60 ± 2.21 (1.0)**	29.62 ± 2.81 (3.0)	29.49 ± 3.18 (2.0)	31.25 ± 2.91 (4.0)
Voice	**9.34 ± 0.85 (1.0)**	11.62 ± 1.50 (2.0)	12.42 ± 1.75 (3.0)	45.54 ± 1.52 (4.0)
Song	**34.24 ± 3.17 (1.0)**	38.22 ± 3.14 (3.0)	38.01 ± 2.36 (2.0)	56.91 ± 5.67 (4.0)
Adult	44.38 ± 1.03 (2.0)	76.69 ± 3.14 (4.0)	**32.60 ± 0.79 (1.0)**	60.63 ± 1.19 (3.0)
Default	**41.62 ± 2.01 (1.0)**	48.17 ± 2.74 (3.0)	48.04 ± 2.49 (2.0)	63.32 ± 1.08 (4.0)
Flickr	**31.04 ± 0.65 (1.0)**	35.18 ± 0.69 (3.0)	35.15 ± 0.73 (2.0)	51.83 ± 0.56 (4.0)
Fera	**55.50 ± 0.76 (1.0)**	57.54 ± 0.38 (2.0)	57.82 ± 0.38 (3.0)	59.11 ± 0.55 (4.0)
WQplants	52.94 ± 4.31 (2.0)	60.46 ± 3.35 (3.0)	62.75 ± 3.86 (4.0)	**49.72 ± 2.39 (1.0)**
WQanimals	52.37 ± 1.43 (2.0)	61.77 ± 0.89 (3.0)	61.90 ± 1.04 (4.0)	**48.48 ± 1.79 (1.0)**
Thyroid	**3.50 ± 0.35 (1.0)**	24.01 ± 1.68 (4.0)	20.03 ± 0.68 (3.0)	7.82 ± 0.34 (2.0)
Rf1	**17.10 ± 0.60 (1.0)**	23.39 ± 0.40 (2.0)	23.51 ± 0.39 (3.0)	37.69 ± 0.90 (4.0)
Pain	22.53 ± 0.49 (2.0)	34.52 ± 1.03 (3.0)	39.71 ± 2.07 (4.0)	**18.61 ± 0.54 (1.0)**
Disfa	21.19 ± 1.03 (2.0)	33.44 ± 0.57 (3.0)	36.89 ± 0.68 (4.0)	**20.09 ± 0.22 (1.0)**
WaterQuality	46.91 ± 1.88 (2.0)	61.26 ± 1.20 (3.0)	64.45 ± 1.25 (4.0)	**40.65 ± 1.57 (1.0)**
Oes97	**29.47 ± 1.95 (1.0)**	33.26 ± 1.60 (2.0)	33.29 ± 1.62 (3.0)	70.07 ± 4.16 (4.0)
Oes10	**22.71 ± 1.55 (1.0)**	25.65 ± 2.15 (2.0)	25.87 ± 2.08 (3.0)	57.32 ± 5.16 (4.0)
Scm20d	**48.72 ± 0.75 (1.0)**	53.36 ± 0.68 (2.0)	57.20 ± 0.76 (3.0)	59.21 ± 0.98 (4.0)
Scm1d	33.94 ± 0.67 (2.0)	**33.29 ± 0.64 (1.0)**	34.10 ± 0.68 (3.0)	57.52 ± 1.13 (4.0)
Ave. rank	**1.40**	2.70	2.95	2.95

Niels Landwehr, Mark Hall, and Eibe Frank. Logistic model trees. *Machine Learning*, 59(1):161–205, 2005.

Vu-Linh Nguyen, Sébastien Destercke, Marie-Hélène Masson, and Rashad Ghassani. Racing trees to query partial data. *Soft Computing*, 25(14):9285–9305, 2021.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Proceedings of the Thirty-Third Conference on Neural Information Processing Systems (NeurIPS)*, 32, 2019.

Paola M.V. Rancoita, Marco Zaffalon, Emanuele Zucca, Francesco Bertoni, and Cassio P. de Campos. Bayesian network data imputation with application to survival tree analysis. *Computational Statistics & Data Analysis*, 93:373–387, 2016.

Jesse Read, Bernhard Pfahringer, Geoffrey Holmes, and Eibe Frank. Classifier chains: a review and perspectives. *Journal of Artificial Intelligence Research*, 70:683–718, 2021.

Lior Rokach and Oded Maimon. Top-down induction of decision trees classifiers-a survey. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 35(4):476–487, 2005.

Gideon Schwarz. Estimating the dimension of a model. *The Annals of Statistics*, pages 461–464, 1978.

Haobo Wang, Weiwei Liu, Yang Zhao, Tianlei Hu, Ke Chen, and Gang Chen. Learning from multi-dimensional partial labels. In *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence (IJCAI)*, pages 2943–2949, 2021.

Joseph Wang and Venkatesh Saligrama. Local supervised learning through space partitioning. In *Proceedings of the 25th International Conference on Neural Information Processing Systems (NIPS)*, pages 91–99, 2012.

Table 9: Subset 0/1 loss (mean ± std.) of each MDC approach (**base learner:** ***logisitic regression***).

Data Set	Subset 0/1 loss (in %)			
	BNCS	BR	CC	CP
Edm	**40.83 ± 11.73 (1.0)**	47.54 ± 12.49 (2.0)	50.54 ± 15.58 (3.0)	55.17 ± 14.99 (4.0)
Jura	60.71 ± 5.75 (2.0)	**59.05 ± 5.86 (1.0)**	63.45 ± 8.29 (3.0)	98.33 ± 4.16 (4.0)
Enb	44.27 ± 8.86 (2.0)	48.32 ± 8.88 (3.0)	**42.96 ± 6.33 (1.0)**	63.54 ± 3.78 (4.0)
Voice	16.07 ± 1.81 (3.0)	**15.69 ± 1.98 (1.0)**	15.85 ± 1.61 (2.0)	81.44 ± 2.32 (4.0)
Song	**57.57 ± 4.27 (1.0)**	60.38 ± 4.93 (3.0)	58.98 ± 5.67 (2.0)	94.26 ± 2.09 (4.0)
Adult	75.67 ± 0.99 (3.0)	72.91 ± 1.33 (2.0)	**71.76 ± 0.84 (1.0)**	86.40 ± 0.95 (4.0)
Default	**81.23 ± 0.61 (1.0)**	82.43 ± 0.79 (3.0)	82.31 ± 1.00 (2.0)	94.01 ± 0.33 (4.0)
Flickr	70.93 ± 1.03 (3.0)	67.74 ± 1.33 (2.0)	**67.40 ± 1.14 (1.0)**	95.97 ± 0.54 (4.0)
Fera	**80.15 ± 0.92 (1.0)**	80.72 ± 1.09 (2.0)	80.86 ± 1.03 (3.0)	83.56 ± 2.56 (4.0)
WQplants	**90.75 ± 2.86 (1.0)**	90.85 ± 2.27 (2.0)	91.79 ± 2.83 (3.0)	92.36 ± 3.88 (4.0)
WQanimals	95.19 ± 2.62 (2.0)	95.47 ± 2.22 (3.0)	**95.09 ± 3.01 (1.0)**	97.83 ± 2.15 (4.0)
Thyroid	**21.63 ± 0.94 (1.0)**	22.76 ± 1.32 (2.0)	23.15 ± 1.41 (3.0)	25.29 ± 1.16 (4.0)
Rf1	**47.51 ± 2.11 (1.0)**	70.88 ± 1.69 (2.0)	72.36 ± 1.92 (3.0)	93.58 ± 0.81 (4.0)
Pain	**24.07 ± 1.50 (1.0)**	24.74 ± 1.41 (2.0)	24.76 ± 1.40 (3.0)	24.86 ± 1.55 (4.0)
Disfa	**60.24 ± 1.42 (1.0)**	60.96 ± 1.15 (3.0)	60.51 ± 1.50 (2.0)	63.40 ± 1.62 (4.0)
WaterQuality	99.43 ± 0.46 (2.0)	**99.06 ± 0.60 (1.0)**	99.72 ± 0.43 (3.5)	99.72 ± 0.43 (3.5)
Oes97	**95.24 ± 4.02 (1.0)**	95.84 ± 3.53 (2.0)	97.03 ± 3.73 (3.0)	100 ± 0.00 (4.0)
Oes10	90.58 ± 3.45 (2.0)	**90.33 ± 4.62 (1.0)**	92.30 ± 3.26 (3.0)	100 ± 0.00 (4.0)
Scm20d	**87.79 ± 1.25 (1.0)**	95.46 ± 0.96 (4.0)	93.89 ± 1.29 (3.0)	92.58 ± 0.98 (2.0)
Scm1d	**80.75 ± 1.02 (1.0)**	89.86 ± 2.17 (3.0)	88.81 ± 0.88 (2.0)	90.81 ± 1.01 (4.0)
Ave. rank	**1.55**	2.20	2.38	3.88

Table 10: Subset 0/1 loss (mean ± std.) of each MDC approach (**base learner:** ***Naive Bayes***).

Data Set	Subset 0/1 loss (in %)			
	BNCS	BR	CC	CP
Edm	**48.79 ± 8.25 (1.0)**	57.12 ± 4.34 (4.0)	52.13 ± 9.22 (2.0)	55.08 ± 15.36 (3.0)
Jura	65.44 ± 7.07 (2.0)	69.33 ± 5.45 (3.0)	**64.05 ± 6.26 (1.0)**	98.89 ± 1.36 (4.0)
Enb	**45.20 ± 4.43 (1.0)**	59.25 ± 5.62 (3.0)	58.99 ± 6.36 (2.0)	62.51 ± 5.81 (4.0)
Voice	**17.92 ± 1.23 (1.0)**	21.62 ± 2.46 (2.0)	22.71 ± 2.94 (3.0)	84.82 ± 2.11 (4.0)
Song	**70.72 ± 4.57 (1.0)**	78.60 ± 4.13 (3.0)	77.71 ± 3.23 (2.0)	93.63 ± 3.26 (4.0)
Adult	92.75 ± 0.68 (3.0)	76.69 ± 3.14 (2.0)	**74.65 ± 2.80 (1.0)**	98.46 ± 0.35 (4.0)
Default	**92.11 ± 1.89 (1.0)**	96.10 ± 2.23 (2.0)	96.47 ± 1.91 (3.0)	99.94 ± 0.03 (4.0)
Flickr	**82.94 ± 1.33 (1.0)**	86.03 ± 1.11 (3.0)	85.89 ± 1.08 (2.0)	96.56 ± 0.55 (4.0)
Fera	**93.34 ± 0.37 (1.0)**	97.94 ± 0.35 (3.0)	98.04 ± 0.31 (4.0)	95.69 ± 0.53 (2.0)
WQplants	**99.15 ± 0.66 (1.5)**	100 ± 0.00 (4.0)	99.91 ± 0.28 (3.0)	**99.15 ± 0.89 (1.5)**
WQanimals	**99.15 ± 0.66 (1.0)**	99.62 ± 0.86 (3.5)	99.43 ± 0.96 (2.0)	99.62 ± 0.63 (3.5)
Thyroid	**20.39 ± 1.92 (1.0)**	90.96 ± 0.95 (4.0)	88.79 ± 1.15 (3.0)	47.24 ± 2.33 (2.0)
Rf1	**69.12 ± 1.09 (1.0)**	83.82 ± 0.97 (2.5)	83.82 ± 0.65 (2.5)	93.05 ± 0.60 (4.0)
Pain	89.14 ± 0.77 (2.0)	93.08 ± 0.80 (3.0)	**87.49 ± 1.38 (1.0)**	91.56 ± 1.10 (4.0)
Disfa	**88.62 ± 2.03 (1.0)**	99.66 ± 0.14 (4.0)	99.47 ± 0.17 (3.0)	94.58 ± 1.02 (2.0)
WaterQuality	100 ± 0.00 (3.0)	100 ± 0.00 (3.0)	100 ± 0.00 (3.0)	**99.72 ± 0.43 (1.0)**
Oes97	96.69 ± 3.12 (3.0)	**94.30 ± 4.50 (1.0)**	94.60 ± 4.36 (2.0)	100 ± 0.00 (4.0)
Oes10	91.79 ± 3.91 (2.0)	**91.54 ± 4.09 (1.0)**	92.53 ± 4.21 (3.0)	99.01 ± 1.22 (4.0)
Scm20d	98.63 ± 0.32 (4.0)	98.27 ± 0.30 (3.0)	97.60 ± 0.43 (2.0)	**96.07 ± 0.43 (1.0)**
Scm1d	95.54 ± 1.25 (3.0)	91.66 ± 0.70 (2.0)	**91.03 ± 0.91 (1.0)**	96.59 ± 0.58 (4.0)
Ave. rank	**1.73**	2.80	2.28	3.20

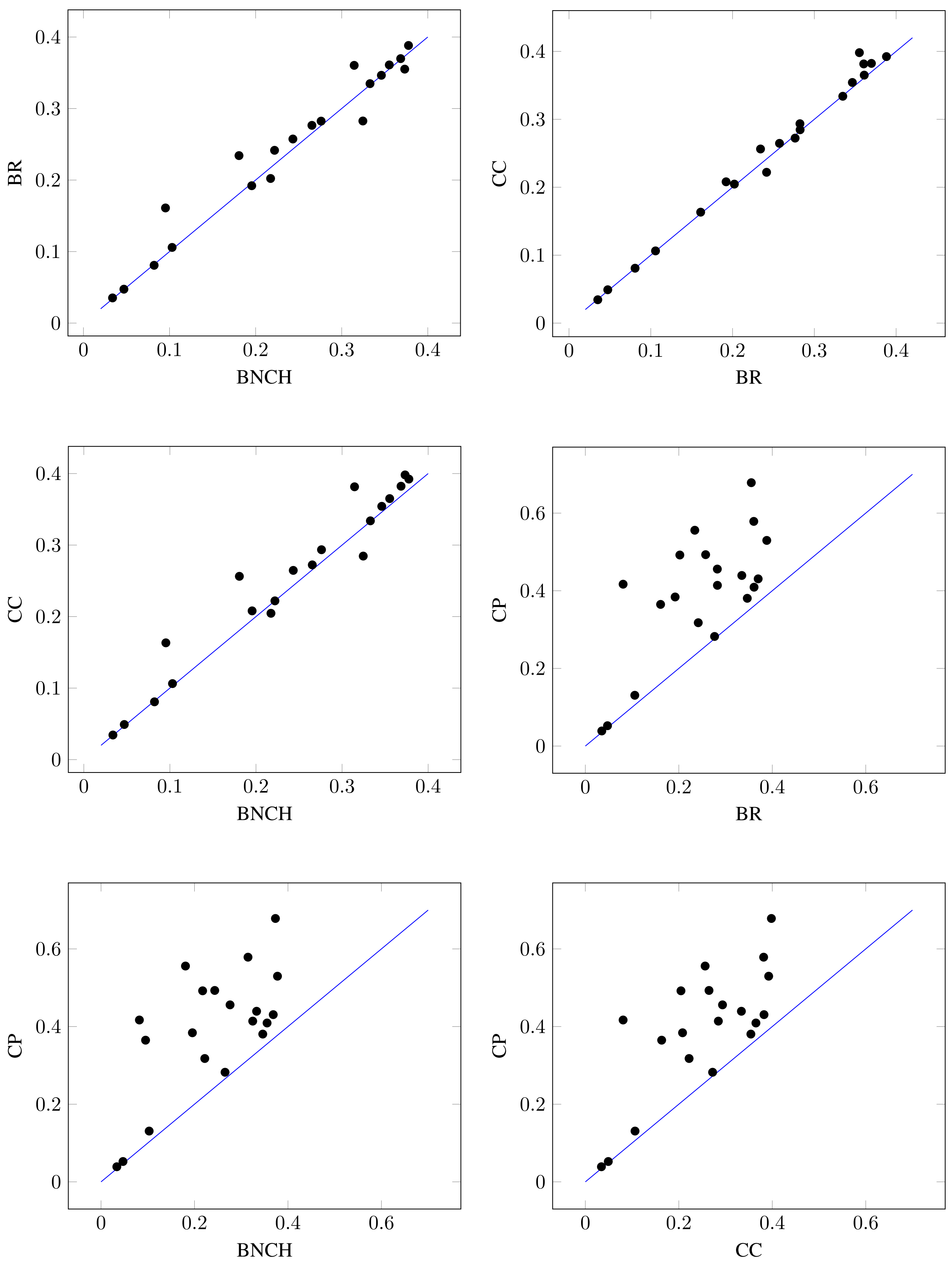

Figure 2: Hamming loss (**base learner: *Logistic regression***)

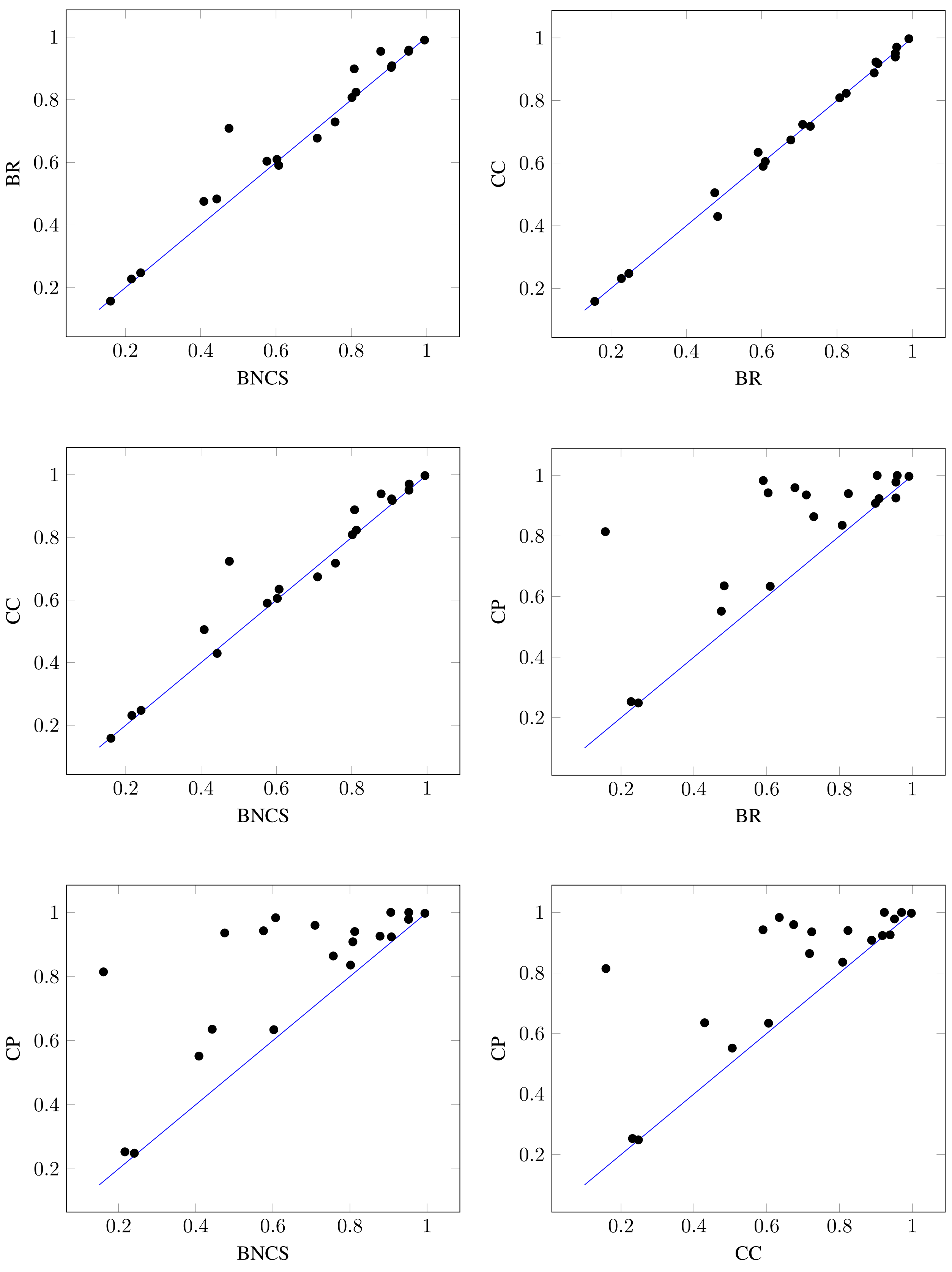

Figure 3: Subset 0/1 loss (**base learner: *Logistic regression***)

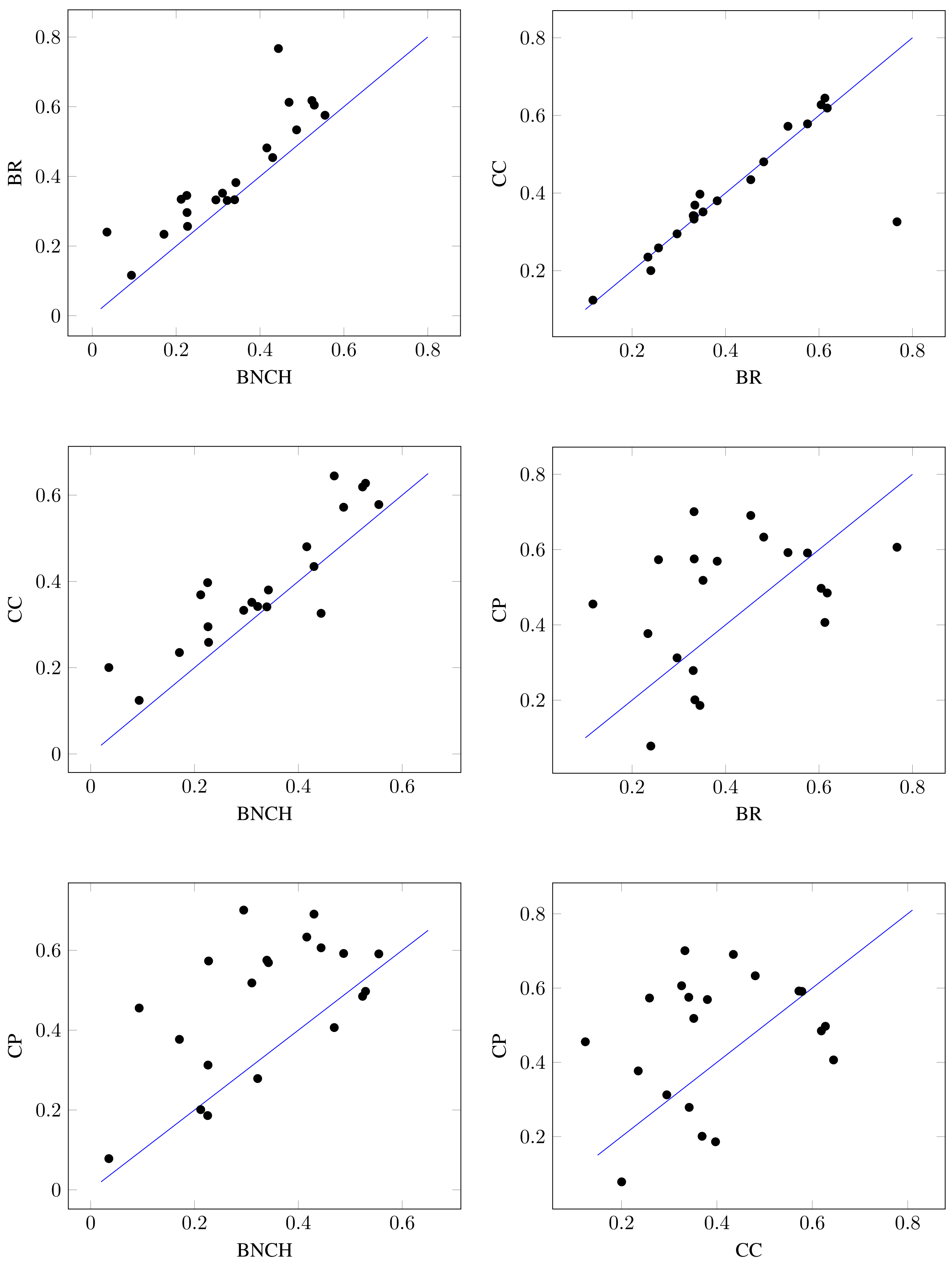

Figure 4: Hamming loss (**base learner:** ***Naive Bayes***)

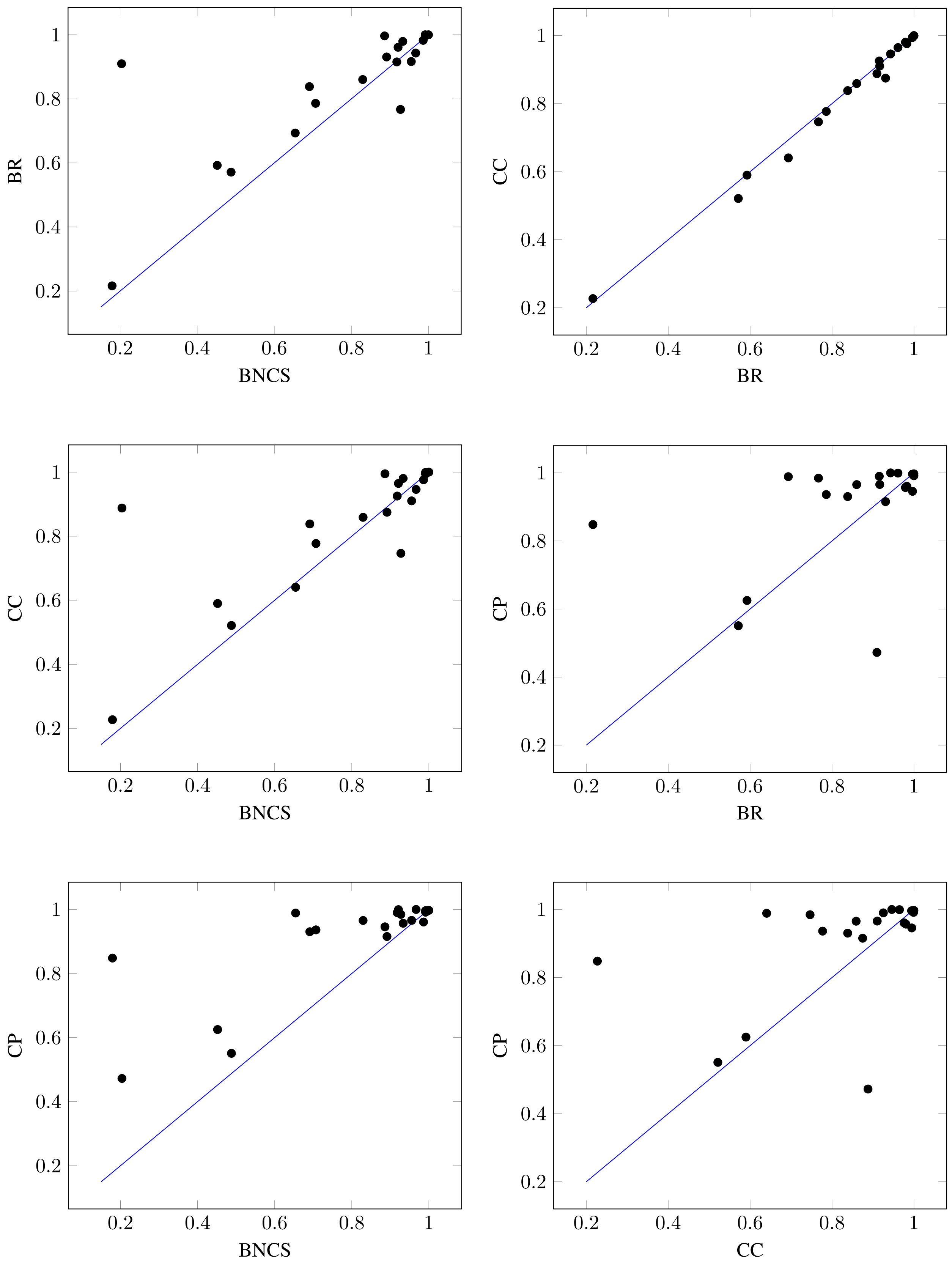

Figure 5: Subset 0/1 loss (**base learner: *Naive Bayes***)