# Ensemble pruning via an integer programming approach with diversity constraints

Marcelo A. Mendes Bastos
Universidade Federal de Minas Gerais
Belo Horizonte, MG, Brazil
marcelo.bastos@dcc.ufmg.br

Humberto Brandão
Universidade Federal de Alfenas
Alfenas, MG, Brazil
humberto.brandao@gmail.com

Cristiano Arbex Valle
Universidade Federal de Minas Gerais
Belo Horizonte, MG, Brazil
arbex@dcc.ufmg.br

## ABSTRACT

Ensemble learning combines multiple classifiers in the hope of obtaining better predictive performance. Empirical studies have shown that ensemble pruning, that is, choosing an appropriate subset of the available classifiers, can lead to comparable or better predictions than using all classifiers. In this paper, we consider a binary classification problem and propose an integer programming (IP) approach for selecting optimal classifier subsets. We propose a flexible objective function to adapt to different datasets as well as constraints to ensure minimum diversity levels in the ensemble. We are able to quickly obtain good solutions for datasets with up to 60,000 data points. Our approach yields competitive results when compared to some of the most used pruning methods in literature.

## 1 INTRODUCTION

Ensemble learning is a popular technique in the domain of machine learning. An ensemble is defined as the aggregation of multiple classifications into a single final decision. It is generally accepted in literature that the precision of an ensemble tends to improve when compared to the behaviour of individual classifiers [27].

Well-known approaches for efficiently generating ensembles include Bagging (bootstrap aggregating) [3] and Boosting [13], in which all classifiers are considered in the aggregation. There are, however, theoretical and empirical studies which have shown that pruning an ensemble by selecting a subset of the classifiers can lead to comparable or better predictions [19, 27].

In this work we tackle the ensemble pruning problem by introducing an integer programming (IP) approach for choosing an optimal subset of binary classifiers. Our formulation optimises a weighted function of the patterns in the binary confusion matrix. This flexible approach allows us to customise the objective function according to the properties of the underlying dataset. As our objective is based on performance we also introduce linear constraints that ensure minimum diversity levels in the ensemble.

Despite the existence of consolidated techniques for ensemble pruning, we believe that our approach contributes to the current knowledge in the field due to the flexibility of the IP paradigm, adaptable to particularities of different problems. One of its advantages is being able to combine performance and diversity criteria. Furthermore, our method is exact, while most algorithms in literature are suboptimal.

In this paper we show that with current solver technology we can find good solutions to relatively large problems in reasonable computational times. We compare our formulation to a full ensemble and six other well-known methods in literature. We report competitive results for publicly available datasets ranging from 195 to 60,000 data points.

The remainder of this paper is organised as follows. In Section 2 we give a brief overview of existing methods in ensemble learning. In Section 3 we present our optimisation approach and in Section 4 we amend it to enforce minimum diversity levels. Our computational experiments are shown in Section 5 and in Section 6 we present our concluding remarks.

## 2 LITERATURE REVIEW

The first step in an ensemble process is to generate a set of distinct classifiers that is precise and diverse. Highly correlated classifiers may hinder the potential benefit of using an ensemble. Several techniques for ensuring diversity in classifiers have been proposed [8, 10], such as randomisation, distinct tuning of hyperparameters and different classifiers. Other diversification techniques include training classifiers with different distributions of the training set and with distinct subsets of features.

The next step is selecting an appropriate classifier subset. This selection can be dynamic [8], where different subsets are chosen for different data points, or static, where a single subset is chosen. Static selection policies are based on ranking, clusters and optimisation.

Ranking methods sort classifiers according to a fitness function. In general they greedily increase the subset size. In Kappa pruning [20], every pair of classifiers is sorted according to a statistical measure of agreement. Reordering techniques [22] are used to build sub-ensembles of increasing size. In [26] classifiers are ranked according to a significance index.

Cluster methods first apply a clustering algorithm to separate classifiers according to some similarity measure and then prune each cluster separately to increase general diversity. Known clustering algorithms include $k$-means [17], where similarity is based on Euclidean distance, and hierarchical agglomerative clustering [15], which employs probabilities.

Several optimisation methods for ensemble pruning have also been proposed, with most offering approximate solutions. The most popular method is hill climbing, which has been applied with several different fitness functions. Some are based on performance [11] (e.g. accuracy), others on diversity [20, 24]. Three examples of diversity-based fitness functions are Complementariness [21], Concurrency [1] and Uncertainty Weighted Accuracy [25]. In [23], reinforcement learning was employed for a greedy method based on diversity. In

[27] a semi-definite programming approach was proposed which considers trade-offs between accuracy and diversity.

The last step in the procedure is combining classifiers into a single prediction, which is usually done through majority voting. For further details we refer the reader to [16].

# 3 FORMULATION

Consider a binary classification problem where data points belong to classes 1 (positive) or 0 (negative). Let $\mathcal{K} = \{1, \dots, K\}$ be the set of classifiers. Let $\mathcal{N}_0 = \{1, \dots, N_0\}$ and $\mathcal{N}_1 = \{1, \dots, N_1\}$ be the sets of negative and positive data points respectively, with $N = N_0 + N_1$ being the total number of data points. Consider a $N_1 \times K$ matrix $B$ where $\beta_{ik} = 1$ if classifier $k \in \mathcal{K}$ correctly classified data point $i \in \mathcal{N}_1$ as positive, $\beta_{ik} = 0$ if it mistakenly classified $i$ as negative. Accordingly consider a $N_0 \times K$ matrix $A$ where $\alpha_{jk} = 0$ if classifier $k \in \mathcal{K}$ correctly classified data point $j \in \mathcal{N}_0$ as negative, $\alpha_{jk} = 1$ if $j$ was mistakenly classified as positive.

Suppose $\mathcal{S} \subseteq \mathcal{K}$ is a set of $S$ classifiers selected to compose a given pruned ensemble. For any data point $i \in \mathcal{N}_1$, $\sum_{s \in \mathcal{S}} \beta_{is}$ is the number of correct positive classifications within $\mathcal{S}$. Accordingly, for any data point $j \in \mathcal{N}_0$, $\sum_{s \in \mathcal{S}} \alpha_{js}$ represents the number of (wrong) positive classifications within $\mathcal{S}$.

We define a threshold $0 \leq L \leq S$ such that for a given data point $i \in \mathcal{N}_1$, $\sum_{s \in \mathcal{S}} \beta_{is} > L$ implies that the ensemble classifies $i$ as positive. If $\sum_{s \in \mathcal{S}} \beta_{is} \leq L$, then $i$ is classified by the ensemble as negative. Similarly for $j \in \mathcal{N}_0$, $\sum_{s \in \mathcal{S}} \alpha_{js} > L$ implies a positive ensemble classification and $\sum_{s \in \mathcal{S}} \alpha_{js} \leq L$ implies a negative ensemble classification. For instance, if $S = 10$ and $L = 5$, then the ensemble classifies a data point as positive if at least 6 individual classifications are positive. If 5 or less are positive, then the ensemble classifies that data point as negative.

In our formulation we let the optimisation define both $\mathcal{S}$ and $L$. Hence we include $L$ as a general integer variable representing the classification threshold and binary variables $x_k = 1$ if classifier $k \in \mathcal{K}$ is chosen to compose the ensemble ($x_k = 0$ otherwise).

**Table 1: Binary classification confusion matrix**

| | | Predicted | |
|---|---|---|---|
| | | 1 | 0 |
| Actual | 1 | $T^+$ | $F^-$ |
| | 0 | $F^+$ | $T^-$ |

Consider the binary confusion matrix shown in Table 1, where $T^+, F^-, T^-$ and $F^+$ are the total number of classifications of each possible pattern. For each patterns we assign weights $W_T^+, W_T^-, W_F^+, W_F^- \in \mathbb{R}$, and the objective function is defined by the weighted sum $W_T^+ T^+ + W_F^- F^- + W_T^- T^- + W_F^+ F^+$.

For modelling this function we define binary variables $t_i^+, f_i^-$ if the ensemble classification of $i \in \mathcal{N}_1$ is respectively a true positive or false negative. Similarly we define binary variables $t_j^-, f_j^+$ if the ensemble classification of $j \in \mathcal{N}_0$ is a true negative or false positive. The IP formulation is given by:

$$\max \sum_{i=1}^{N_1} (W_T^+ t_i^+ + W_F^- f_i^-) + \sum_{j=1}^{N_0} (W_T^- t_j^- + W_F^+ f_j^+) \tag{1}$$

subject to

$$(L+1) - \sum_{k=1}^{K} x_k \, \beta_{ik} \leq (K+1)(1 - t_i^+), \qquad \forall i \in \mathcal{N}_1 \tag{2}$$

$$\sum_{k=1}^{K} x_k \, \beta_{ik} - L \leq (K+1) t_i^+, \qquad \forall i \in \mathcal{N}_1 \tag{3}$$

$$t_i^+ + f_i^- = 1, \qquad \forall i \in \mathcal{N}_1 \tag{4}$$

$$\sum_{k=1}^{K} x_k \, \alpha_{jk} - L \leq K(1 - t_j^-), \qquad \forall j \in \mathcal{N}_0 \tag{5}$$

$$(L+1) - \sum_{k=1}^{K} x_k \, \alpha_{jk} \leq K t_j^-, \qquad \forall j \in \mathcal{N}_0 \tag{6}$$

$$f_j^+ + t_j^- = 1, \qquad \forall j \in \mathcal{N}_0 \tag{7}$$

$$x_k \in \mathbb{B} \qquad \forall k \in \mathcal{K} \tag{8}$$

$$t_i^+, f_i^- \in \mathbb{B} \qquad \forall i \in \mathcal{N}_1 \tag{9}$$

$$t_j^-, f_j^+ \in \mathbb{B} \qquad \forall j \in \mathcal{N}_0 \tag{10}$$

$$0 \leq L \leq K, \tag{11}$$

$$L \in \mathbb{Z} \tag{12}$$

Constraints (2) ensure that a positive data point $i \in \mathcal{N}_1$ has $t_i^+ = 1$ if the number of individual positive classifications exceeds $L$. Conversely, constraints (3) ensure that $t_i^+ = 0$ if the number of individual positive classifications is no more than $L$. Constraints (4) ensure that either $t_i^+ = 1$ or $f_i^- = 1$. Constraints (5) guarantee that a negative data point $j \in \mathcal{N}_0$ has $t_j^- = 0$ if the number of positive classifications exceeds $L$. Otherwise, constraints (6) make sure that $t_j^- = 1$. Constraints (7) ensure that either $f_j^+ = 1$ or $t_j^- = 1$. Constraints (8)-(12) define variables bounds.

## 3.1 Objective function

For some classification problems, it may be desirable to optimise some patterns instead of others. For instance, in an investment decision, investing in the wrong project may cause bankruptcy while not investing in a promising project may be seen as a regretful but acceptable lost opportunity. In this case prioritising the minimisation of $F^+$ is desirable. The weights in Equation (1) provide flexibility for defining optimisation criteria depending on the characteristics of the dataset at hand (such as being highly imbalanced). A few examples are outlined below.

Accuracy is defined as $\frac{T^+ + T^-}{N}$. As $N$ is constant we can maximise accuracy by defining weights $W_T^+ = W_T^- = 1$ and $W_F^+ = W_F^- = 0$. Notice that if we choose this objective then constraints (3) and (6) are redundant as maximising positive weights $W_T^+$ and $W_T^-$ ensure that $t_i^+ = 1$ and $t_i^- = 1$ if allowed by constraints (2) and (5). Similarly, Recall is defined as $\frac{T^+}{T^+ + F^-} = \frac{T^+}{N_1}$ and can be maximised by setting $W_T^+ = 1$ and $W_T^- = W_F^+ = W_F^- = 0$ (with constraints (3) being redundant).

Accuracy may not be the most appropriate metric for the selected datasets since several are imbalanced. Let $\theta = \frac{N_1}{N}$ be the dataset imbalance level. If, for instance, $\theta \geq 1 - \epsilon$ for small $\epsilon$, a high accuracy can be achieved by simply classifying every data point as positive. For imbalanced datasets a possibly useful configuration is setting weights $W_T^+ = (1-\theta)$, $W_T^- = \theta$ and $W_F^+ = W_F^- = 0$. We refer to this function as $\theta$-weighted).

Balanced Accuracy (BA) is an alternative metric which weighs equally the accuracy of positive data points and the accuracy of negative data points. BA is a more appropriate measure for imbalanced datasets [4] and is given by:

$$\text{BA} = \frac{\frac{T^+}{T^+ + F^-} + \frac{T^-}{T^- + F^+}}{2} = \frac{\frac{T^+}{N_1} + \frac{T^-}{N_0}}{2} \tag{13}$$

Theorem 1 shows that maximising BA is equivalent to maximising the $\theta$-weighted function.

Theorem 1. *Maximising the $\theta$-weighted configuration is equivalent to maximising balanced accuracy.*

Proof. Following the definition of the $\theta$-weighted function in Section 3.1, objective function $z$ can be written as:

$$\max z = \left(1 - \frac{N_1}{N}\right)T^+ + \frac{N_1}{N}T^-$$

where $T^+ = \sum_{i=1}^{N_1} t_i^+$, $T^- = \sum_{j=1}^{N_0} t_j^-$ and $\theta = \frac{N_1}{N}$. As $N = N_0 + N_1$ it follows that:

$$\begin{aligned} \max \quad z &= \frac{N_0}{N}T^+ + \frac{N_1}{N}T^- \\ \max Nz &= N_0 T^+ + N_1 T^- \\ \max \quad cz &= \frac{T^+}{N_1} + \frac{T^-}{N_0} \end{aligned}$$

where $c = \frac{N}{N_1 N_0} > 0$ is a scaling factor, and thus maximising the $\theta$-weighted function is equivalent to maximising balanced accuracy. □

## 4 DIVERSITY

As mentioned before many ensemble pruning algorithms employ diversity criteria. Our proposed formulation optimises a performance measure, and in this section we introduce a way to control diversity with linear constraints. We consider a diversity measure called Pairwise Failure Crediting (PFC), proposed originally by [5], chosen due to well-known good performance in imbalanced datasets [2, 12]. PFC measures how diverse an individual classifier is from the remaining classifiers in the ensemble.

PFC is calculated as follows. For each classifier $k$, we compute a *failure pattern* (FP). A FP is a string of 0's and 1's with length $N$. A '0' in the string means that the classifier failed to correctly predict the corresponding data point and a '1' means that it predicted the data point correctly (irrespective of its real value). Once we have all failure patterns we take any two classifiers $k$ and $l$ and calculate their Hamming distance. The Hamming distance between same-length strings is the number of different characters in the same positions. For example, if $\text{FP}_k = \{0011011101\}$ and $\text{FP}_l = \{0110001110\}$, the Hamming distance between $k$ and $l$ is 5 (characters 2, 4, 6, 9 and 10 differ). Next, we sum all failures by both classifiers - that is, we sum the number of zeros in both strings which, in the example, is 9. The *failure credit* (FC) between $k$ and $l$ is obtained by dividing the Hamming distance by the sum of failures. In the example, $\text{FC}_{kl} = 5/9$. For every pair $k, l \in \mathcal{K}$ we compute $\text{FC}_{kl}$.

Consider again $\mathcal{S}$ as a set of $S \leq K$ classifiers selected to compose an ensemble. We assume without loss of generality that classifiers in $\mathcal{S}$ are indexed by $k = 1, \ldots, S$. PFC is defined as:

$$\text{PFC}_k = \frac{\sum_{l=1, l\neq k}^{S} \text{FC}_{kl}}{S-1} \qquad k \in \mathcal{S}$$

A (maximum) value of 1 in $\text{PFC}_k$ means that $k$ classifies all data points differently from every other classifier in the ensemble, and a (minimum) value of 0 means that $k$ is identical to all other classifiers. Both extreme cases imply that all other classifiers are identical among themselves.

For ensuring minimum desired diversity levels, we propose two approaches: (i) the minimum PFC of any individual classifier is at least a certain threshold $0 \leq \tau \leq 1$ in order to prevent very similar pairs of classifiers and (ii) the average PFC of the ensemble must be at least a certain threshold $0 \leq \gamma \leq 1$ to ensure an overall good level of diversity. Clearly we must have $\gamma \geq \tau$.

We add the following new decision variables. Let $y_{kl} = 1$ if both classifiers $k$ and $l$ have been selected to be part of the ensemble, and $y_{kl} = 0$ if at most one of $k$ and $l$ is chosen to compose the ensemble. This adds $\binom{K}{2}$ extra variables (for every possible pair $k, l$). For simplicity, both $y_{kl}$ and $y_{lk}$ denote the exact same variable. The following constraints ensure that $y_{kl}$ takes the correct values:

$$y_{kl} \geq x_k + x_l - 1 \qquad \forall k, l \in \mathcal{K}, k < l \tag{14}$$
$$y_{kl} \leq x_k \qquad \forall k, l \in \mathcal{K}, k < l \tag{15}$$
$$y_{kl} \leq x_l \qquad \forall k, l \in \mathcal{K}, k < l \tag{16}$$
$$y_{kl} \geq 0 \qquad \forall k, l \in \mathcal{K}, k < l \tag{17}$$

Notice that there is no need for the $y_{kl}$ variables to be binary. Both $x_k$ and $x_l$ being binary ensure $y_{kl}$ to be 0-1 in any integer solution.

We then rewrite the PFC equation using variables $x_k$ and $y_{kl}$:

$$\text{PFC}_k = \frac{\sum_{l=1, l\neq k}^{K} \text{FC}_{kl}\; y_{kl}}{\sum_{m=1}^{K} x_m - 1} \qquad \forall k \in \mathcal{K}$$

The term $\sum_{m=1}^{K} x_m$ is the cardinality of the ensemble and any non-selected classifier $k$ (with $x_k = 0$) has a PFC equal to zero (as all $y_{kl} = 0, l \neq k$).

The following linear constraints enforce that every classifier has a minimum PFC of $\tau$:

$$\sum_{\substack{l=1\\ l\neq k}}^{K} \text{FC}_{kl}\; y_{kl} \geq \tau\left(\sum_{m=1}^{K} x_m - 1\right) - K\tau(1 - x_k) \qquad \forall k \in \mathcal{K} \tag{18}$$

The term $K\tau(1 - x_k)$ ensures that the constraints above are only enforced if classifier $k$ is chosen to compose the ensemble.

The following nonlinear constraint ensures that the average PFC of the ensemble is at least $\gamma$:

$$\frac{1}{\sum_{m=1}^{K} x_m} \frac{\sum_{k=1}^{K} \sum_{l=1, l\neq k}^{K} \text{FC}_{kl}\; y_{kl}}{\sum_{m=1}^{K} x_m - 1} \geq \gamma \tag{19}$$

Observe that in Equation (19) the FCs of every pair are added twice. We use this fact to linearise this expression. For a given subset $\mathcal{S}$, the average PFC $\mu_{\text{PFC}}$ is given by:

$$\begin{aligned}\mu_{\text{PFC}} &= \frac{1}{S}\sum_{k=1}^{S}\frac{\sum_{l=1,l\neq k}^{S}\text{FC}_{kl}}{S-1}\\ &= \frac{1}{S(S-1)}\sum_{k=1}^{S}\sum_{\substack{l=1\\ l\neq k}}^{S}\text{FC}_{kl}\\ &= \frac{2}{S(S-1)}\sum_{k=1}^{S-1}\sum_{l=k+1}^{S}\text{FC}_{kl}\\ &= \frac{1}{\binom{S}{2}}\sum_{k=1}^{S-1}\sum_{l=k+1}^{S}\text{FC}_{kl} = \mu_{\text{FC}}\end{aligned}$$

where $\mu_{\text{FC}}$ denotes the average FC of all pairs in the ensemble. We have that the average PFC among all classifiers in the ensemble is equal to the average FC among all pairs.

If $S$ classifiers are selected in the ensemble, then the number of $y_{kl}$ variables that take value 1 is exactly $\binom{S}{2}$. Therefore we can ensure that the average PFC is at least $\gamma$ with the following linear constraint:

$$\sum_{k=1}^{K-1}\sum_{l=k+1}^{K}\text{FC}_{kl}\ y_{kl} \geq \gamma\sum_{k=1}^{K-1}\sum_{l=k+1}^{K} y_{kl} \tag{20}$$

The expanded formulation with minimum diversity levels is given by maximising (1) subject to (2)-(18) and (20). It requires $\binom{K}{2}$ extra variables and a similar number of extra constraints. Even so, we observed empirically in Section 5.3 that the inclusion of such constraints causes a negligible decrease in solution quality.

## 5 COMPUTATIONAL EXPERIMENTS

In this section we outline the computational experiments used to evaluate the proposed formulation. We used 9 publicly available datasets, outlined in Table 2[1], ranging from $N = 195$ to $N = 60{,}000$. Imbalance parameter $\theta$ is shown in the table.

### 5.1 Description of the experiments

We prepared 10 different heterogeneous classifier models. Each model was instantiated a number of times with different random seeds and parameters. We set $K$ as multiples of 10 in order to have an equal number of instantiations of each classifier. For instance, if $K = 60$, we have 6 classifiers of each model. In our experiments, reported below, we used $K = \{40, 60, 80, 100\}$. Each classifier produces, as output, a probability of a data point being positive. This probability is rounded to define matrices $A$ and $B$.

For evaluating performance we used a stratified 10-fold cross-validation procedure. The $N$ data points are initially shuffled randomly and the dataset is split into 10 folds. At each iteration, one of the folds is left out as an independent set. The results presented below are based solely on this set. The other 9 folds, comprising 90% of the original dataset, are joined and split into two sets: a training set, containing 63% of the data points, is used to optimise the individual classifiers. A validation set, comprising the remaining 27% data points, is used to optimise the ensemble algorithms.

The procedure above is repeated 10 times: in each we vary the random seeds required to both shuffle the dataset and initialise the individual classifiers. For each value of $K$ and for each instance shown in Table 2, we run 100 experiments: 10 random initialisations × 10 folds. For ensuring reproducibility of our results, we have made all necessary data publicly available. A link and a description of the classifiers can be found in the supplementary material[2].

**Table 2: Selected datasets from the UCI Machine Learning Repository [18]**

| Identifier | Dataset | Features | $N$ | $N_0$ | $N_1$ | $\theta$ |
|---|---|---|---|---|---|---|
| PRK | Parkinsons | 23 | 195 | 48 | 147 | 0.77 |
| MSK | Musk (Version 1) | 168 | 476 | 269 | 207 | 0.44 |
| BCW | Breast Cancer Wisconsin | 32 | 569 | 357 | 212 | 0.37 |
| QSR | QSAR biodegradation | 41 | 1055 | 356 | 699 | 0.66 |
| DRD | Diabetic Retinopathy Debrecen | 20 | 1151 | 540 | 611 | 0.53 |
| SPA | Spambase | 57 | 4601 | 2788 | 1813 | 0.39 |
| DEF | Default of credit card clients | 24 | 30000 | 23364 | 6636 | 0.22 |
| BMK | Bank Marketing | 21 | 41188 | 36548 | 4640 | 0.11 |
| APS | APS Failure at Scania Trucks | 171 | 60000 | 59000 | 1000 | 0.02 |

### 5.2 Benchmarks

We compare our formulation to seven other approaches: Full (non-pruned) Ensemble (FE), Reduced-Error Pruning with Backfitting [14] (hereby Backfitting or BFT), Kappa pruning [20] and four different hill climbing based methods. Here we report here results for the four approaches with the best overall out-of-sample performance. The full results are available in the supplementary material accompanying this paper. All benchmarks classify data points based on majority voting and are allowed to run for a maximum of 5 minutes.

Backfitting follows a greedy approach with revision. From an empty subset $S$, BFT iteratively adds to $S$ a classifier $s$ such that the accuracy of $S \cup s$ is maximised. This process is repeated until $M$ classifiers are added to $S$, with ties broken arbitrarily. Whenever a classifier is added, the greedy choice is revised through a local search procedure. Each classifier in the ensemble is iteratively replaced by another previously left out. If the overall accuracy is improved, the method starts again with the new subset $S$. Kappa pruning is similar, but does not revise the greedy choice and optimises the $\kappa$-statistic [6]. Both methods require $M$ to be fixed. For a fairer comparison, we varied $M$ within 20% and 80% of $K$. The best in-sample results are used to evaluate the independent set.

The other benchmarks use the forward version of the hill climbing search algorithm, differing in the selected fitness function. In all four methods, the first iteration selects the individual classifier with maximum accuracy. Then classifiers are greedily added so as to maximise the selected fitness. This process is repeated until all classifiers are added to $S$. The chosen ensemble is the one with best fitness over all the ensembles iteratively created. As opposed to the other benchmarks, direct hill climbing does not define the ensemble size *a priori*. The fitness functions chosen are the same as tested by [25]: Accuracy, Complementariness [21], Concurrency [1] (HC-CON) and Uncertainty Weighted Accuracy [25] (HC-UWA).

We compare our method to BFT, HC-CON, HC-UWA and FE.

### 5.3 Solving the formulation

Due to limited space, in this paper we refrain from evaluating our proposed formulation with regards to the computational effort

[1] All datasets can be found at the UCI Machine Learning Repository [18]

[2] The supplementary material is available here.

required to solve it. We leave that for future work. We however observed in practice that, with a 5-minutes time limit, we were able to either optimally solve or terminate the algorithm with small optimality gaps for all instances.

The average gaps for the results reported in Section 5.4 for $K = 100$ are summarised in Table 3. The "No diversity" column corresponds to **F1** in that section, and only constraints (2)-(12) are used. The "With diversity" column corresponds to **F3**, which uses constraints (2)-(18) and (20). The largest instance, APS, had average gaps of only 0.1% in both cases. The hardest instance was DEF (6.7% and 6.9%). The only case where a difference was notable was for the DRD instance (4.3% and 6.1%).

In our view, even the hardest instances were still relatively close to optimality considering the short computational time. We used CPLEX 12.8 [7] with default parameters as the IP solver and we ran all experiments in an Intel Core(TM) I7-7700 @ 3.60GHz with 32GB of RAM, using 8 cores and having Linux as the operating system.

**Table 3: Avg. optimality gaps and standard deviations (in %).**

| Instance | 27% of $N$ | No diversity | | With diversity | |
|---|---|---|---|---|---|
| | | Avg. | Std. | Avg. | Std. |
| PRK | 26 | 0.0 | 0.0 | 0.0 | 0.0 |
| BCW | 129 | 0.0 | 0.0 | 0.0 | 0.0 |
| MSK | 154 | 0.0 | 0.0 | 0.0 | 0.0 |
| QSR | 285 | 0.0 | 0.0 | 0.0 | 0.1 |
| DRD | 311 | 4.3 | 2.0 | 6.1 | 1.8 |
| SPA | 1242 | 0.0 | 0.0 | 0.2 | 0.2 |
| DEF | 8100 | 6.7 | 0.4 | 6.9 | 0.4 |
| BMK | 11121 | 5.4 | 0.3 | 5.5 | 0.2 |
| APS | 16200 | 0.1 | 0.0 | 0.1 | 0.0 |

## 5.4 Accuracy

In the results reported in this section, we seek to maximise accuracy regardless of $\theta$, by setting $W_T^+ = W_T^- = 1$ and $W_F^+ = W_F^- = 0$. We evaluate three different configurations.

In the first, **F1**, we employ only constraints (2)-(12), without enforcing diversity. The other two configurations, **F2** and **F3**, enforce minimum diversity levels in the hope of preventing possible overfitting. In **F2** we only constrain the overall average PFC by setting $\tau = 0$ and $\gamma = \frac{\text{PFC}_{\min}+\text{PFC}_{\text{avg}}}{2}$, where $\text{PFC}_{\min}$ and $\text{PFC}_{\text{avg}}$ are the minimum individual PFCs among all classifiers and the average PFC of the full ensemble. In **F3** we also set $\tau = \text{PFC}_{\min}$.

Table 4 summarises the results with an average rank per value of $K$ across all datasets. We use the ranking procedure of [9]. The full results are shown in the supplementary material.

**Table 4: Average ranks of accuracies**

| $K$ | F1 | F2 | F3 | BFT | HC-CON | HC-UWA | FE |
|---|---|---|---|---|---|---|---|
| 40 | 3.82 | 3.74 | 3.79 | 4.04 | 3.71 | 3.66 | 5.25 |
| 60 | 3.82 | 3.70 | 3.76 | 4.03 | 3.66 | 3.66 | 5.39 |
| 80 | 3.82 | 3.80 | 3.75 | 4.09 | 3.54 | 3.58 | 5.42 |
| 100 | 3.91 | 3.72 | 3.72 | 4.07 | 3.55 | 3.56 | 5.48 |
| **Avg:** | **3.84** | **3.74** | **3.75** | **4.06** | **3.61** | **3.61** | **5.38** |

The results suggest that while our proposed formulation is overall competitive, it was slightly outperformed by HC-CON and HC-UWA - both in terms of average accuracy (from the table in the supplementary material) and average rank. Still, with the exception of FE, the difference between BFT (worst performing) and HC-CON (best performing) was 0.33% in terms of average overall accuracy and 0.45 in terms of average rank. Adding diversity constraints to our formulation also had a small beneficial impact in improving average accuracy and reducing the average ranking. In 11 out of the 36 cases, **F2** outperformed all benchmarks.

Both HC benchmarks had a higher dispersion of accuracies than our methods. Also, adding diversity in **F2** and **F3** helped reduce dispersion. Further studies on either better enforcing these constraints or proposing new constraints based on alternative diversity measures remain as future work. Since our proposed method is exact in nature (although limited to 5 minutes), in the supplementary material we discuss in more detail the effects of overfitting.

## 5.5 Balanced accuracy

In this section, we evaluate the out-of-sample performance according to Balanced Accuracy (BA). We employ **F1** as defined earlier and a modified **F1** where we maximise the $\theta$-weighted configuration suggested in Section 3.1. Tables 5 and 6 show the results. In Table 5, we show results for only $K = \{80, 100\}$ and for the five largest datasets, but the full results are available in the supplementary material. A bold value in the **Avg.** columns means that our formulation obtained a higher average than all benchmarks. The averages in the last row are for all results, not only those displayed in the table. We did not rerun the experiments for the accuracy version of **F1** nor for the benchmarks, rather we used the same ensemble subsets to calculate the corresponding balanced accuracies.

Here both configurations of our formulation outperformed the benchmarks. **F1** ($\theta$-weighted) consistently outperformed **F1** and all benchmarks, with better ranks, overall average accuracies and lower dispersion, especially for the larger (and more imbalanced) datasets. **F1** ($\theta$-weighted) outperformed all benchmarks in 22 out of 36 cases. It had worse performance than the benchmarks in MSK and DRD, which are the most balanced datasets. These results suggest that being able to configure the objective function according to the characteristics of the dataset at hand can be highly beneficial.

## 6 CONCLUSIONS AND FUTURE DIRECTIONS

In this work we proposed an IP approach for the problem of selecting a subset of classifiers in ensemble learning, with the goal of maximising a weighted function of the patterns in the confusion matrix. In order to combine performance and diversity criteria, we also proposed linear constraints to enforce minimum diversity levels. We observed that state-of-the-art solvers can find good solutions in reasonable computational times for relatively large datasets. The IP approach is, in our view, able to provide a flexible exact algorithm which can also be used as a heuristic if short computational time limits are required. This approach has the additional advantage of providing bounds on optimal values.

We compared our formulation to seven well-known benchmarks. We used a stratified 10-fold cross validation procedure and evaluated the effect of enforcing minimum diversity levels and varying

**Table 5: Balanced Accuracy averages and standard deviations**

| Dataset | $K$ | **F1** | | **F1** ($\theta$-weighted) | | BFT | | HC-CON | | HC-UWA | | FE | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Avg. | Std. | Avg. | Std. | Avg. | Std. | Avg. | Std. | Avg. | Std. | Avg. | Std. |
| DRD | 80 | 0.7457 | 0.0064 | 0.7468 | 0.0046 | 0.7503 | 0.0057 | 0.7525 | 0.0065 | 0.7525 | 0.0072 | 0.7126 | 0.0077 |
| | 100 | 0.7404 | 0.0088 | 0.7466 | 0.0093 | 0.7479 | 0.0063 | 0.7506 | 0.0075 | 0.7551 | 0.0057 | 0.7148 | 0.0075 |
| SPA | 80 | 0.9504 | 0.0021 | **0.9521** | 0.0018 | 0.9488 | 0.0017 | 0.9512 | 0.0017 | 0.9493 | 0.0020 | 0.9403 | 0.0009 |
| | 100 | 0.9500 | 0.0022 | **0.9517** | 0.0025 | 0.9485 | 0.0020 | 0.9515 | 0.0015 | 0.9493 | 0.0018 | 0.9402 | 0.0008 |
| DEF | 80 | **0.6662** | 0.0014 | **0.6985** | 0.0016 | 0.6484 | 0.0020 | 0.6553 | 0.0009 | 0.6557 | 0.0011 | 0.6484 | 0.0011 |
| | 100 | **0.6661** | 0.0019 | **0.6992** | 0.0014 | 0.6482 | 0.0018 | 0.6542 | 0.0020 | 0.6560 | 0.0007 | 0.6473 | 0.0012 |
| BMK | 80 | **0.7765** | 0.0055 | **0.8684** | 0.0015 | 0.7322 | 0.0046 | 0.7477 | 0.0028 | 0.7469 | 0.0018 | 0.6822 | 0.0036 |
| | 100 | **0.7794** | 0.0053 | **0.8694** | 0.0012 | 0.7363 | 0.0040 | 0.7478 | 0.0029 | 0.7479 | 0.0018 | 0.6762 | 0.0031 |
| APS | 80 | **0.8731** | 0.0039 | **0.9395** | 0.0038 | 0.8398 | 0.0045 | 0.8535 | 0.0033 | 0.8500 | 0.0040 | 0.8021 | 0.0037 |
| | 100 | **0.8735** | 0.0053 | **0.9416** | 0.0041 | 0.8447 | 0.0064 | 0.8562 | 0.0040 | 0.8513 | 0.0032 | 0.8006 | 0.0032 |
| | **Average:** | **0.8433** | **0.0080** | **0.8665** | **0.0063** | **0.8313** | **0.0069** | **0.8397** | **0.0075** | **0.8383** | **0.0076** | **0.8111** | **0.0057** |

**Table 6: Average ranks of balanced accuracies**

| $K$ | **F1** | **F1** | BFT | HC-CON | HC-UWA | FE |
|---|---|---|---|---|---|---|
| | | ($\theta$-weighted) | | | | |
| 40 | **2.98** | **2.37** | 3.94 | 3.33 | 3.43 | 4.95 |
| 60 | **2.88** | **2.45** | 3.97 | 3.29 | 3.40 | 5.02 |
| 80 | **2.92** | **2.39** | 3.97 | 3.25 | 3.38 | 5.09 |
| 100 | **2.97** | **2.41** | 4.00 | 3.20 | 3.36 | 5.06 |
| **Avg:** | **2.94** | **2.40** | **3.97** | **3.27** | **3.39** | **5.03** |

the weights assignments of the objective function. The results suggest that our approach is competitive and its flexibility can be beneficial when dealing with different datasets. All data required to reproduce our results is made available as supplementary material.

As future work we intend to experiment with different criteria and larger datasets. We also plan to study alternative diversity constraints and to research IP techniques/matheuristics for both finding good solutions quickly and solving the formulation faster.

## REFERENCES

[1] Robert E. Banfield, Lawrence O. Hall, Kevin W. Bowyer, and W.Philip Kegelmeyer. 2005. Ensemble diversity measures and their application to thinning. *Information Fusion* 6, 1 (3 2005), 49–62. https://doi.org/10.1016/j.inffus.2004.04.005

[2] Urvesh Bhowan, Mark Johnston, Mengjie Zhang, and Xin Yao. 2012. Evolving diverse ensembles using genetic programming for classification with unbalanced data. *IEEE Transactions on Evolutionary Computation* 17, 3 (2012), 368–386. https://doi.org/10.1109/TEVC.2012.2199119

[3] L. Breiman. 1996. Bagging predictors. *Machine Learning* 24, 2 (1996), 123–140. https://doi.org/10.1007/BF00058655

[4] Kay Henning Brodersen, Cheng Soon Ong, Klaas Enno Stephan, and Joachim M Buhmann. 2010. The balanced accuracy and its posterior distribution. In *2010 20th international conference on pattern recognition.* IEEE, 3121–3124. https://doi.org/10.1109/ICPR.2010.764

[5] A. Chandra and X. Yao. 2006. Ensemble learning using multi-objective evolutionary algorithms. *Journal of Mathematical Modelling and Algorithms* 5, 1 (2006), 417–445. https://doi.org/10.1007/s10852-005-9020-3

[6] J. Cohen. 1960. A coefficient of agreement for nominal scales. *Educational and psychological measurement* 20, 1 (1960), 37–46.

[7] CPLEX Optimizer. 2021. IBM. Available from https://www.cplex.com, last accessed April 1st.

[8] R. M. O. Cruz, R. Sabourin, and G. D. C. Cavalcanti. 2018. Dynamic classifier selection: Recent advances and perspectives. *Information Fusion* 41, May (2018), 195–216. https://doi.org/10.1016/j.inffus.2017.09.010

[9] Janez Demšar. 2006. Statistical comparisons of classifiers over multiple data sets. *The Journal of Machine Learning Research* 7 (2006), 1–30. https://doi.org/10.5555/1248547.1248548

[10] R. P. W. Duin. 2002. The combining classifier: to train or not to train? *Object recognition supported by user interaction for service robots* 2, c (2002), 765–770. https://doi.org/10.1109/ICPR.2002.1048415

[11] Wei Fan, Fang Chu, Haixun Wang, and Philip S. Yu. 2002. Pruning and dynamic scheduling of cost-sensitive ensembles. In *AAAI/IAAI*. 146–151.

[12] Everlandio RQ Fernandes, André CPLF de Carvalho, and André LV Coelho. 2015. An evolutionary sampling approach for classification with imbalanced data. In *IJCNN'15. International Joint Conference on Neural Networks.* IEEE, 1–7. https://doi.org/10.1109/IJCNN.2015.7280760

[13] Y. Freund and R. E. Schapire. 1997. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences* 55, 1 (1997), 119–139. https://doi.org/10.1006/jcss.1997.1504

[14] J. H. Friedman and W. Stuetzle. 1981. Projection pursuit regression. *Journal of the American statistical Association* 76, 376 (1981), 817–823. https://doi.org/10.1080/01621459.1981.10477729

[15] G. Giacinto, F. Roli, and G. Fumera. 2000. Design of effective multiple classifier systems by clustering of classifiers. In *Proceedings 15th International Conference on Pattern Recognition. ICPR-2000*, Vol. 2. IEEE, 160–163. https://doi.org/10.1109/ICPR.2000.906039

[16] J. Kittler, M. Hatef, R. P. W. Duin, and J. Matas. 1998. On combining classifiers. *IEEE transactions on pattern analysis and machine intelligence* 20, 3 (1998), 226–239. https://doi.org/10.1109/ICPR.1996.547205

[17] Aleksandar Lazarevic and Zoran Obradovic. 2001. Effective pruning of neural network classifier ensembles. In *IJCNN'01. International Joint Conference on Neural Networks.*, Vol. 2. IEEE, 796–801. https://doi.org/10.1109/IJCNN.2001.939461

[18] M. Lichman. 2021. UCI Machine Learning Repository. Available from http://archive.ics.uci.edu/ml, last accessed April 1st.

[19] Z. Lu, X. Wu, X. Zhu, and J. Bongard. 2010. Ensemble pruning via individual contribution ordering. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.* ACM, 871–880. https://doi.org/10.1145/1835804.1835914

[20] D. D. Margineantu and T. G. Dietterich. 1997. Pruning adaptive boosting. In *Proceedings of the 14th International Conference of Machine Learning*, Vol. 97. Citeseer, 211–218. https://doi.org/10.1007/11875581_39

[21] Gonzalo Martınez-Muñoz and Alberto Suárez. 2004. Aggregation ordering in bagging. In *Proc. of the IASTED International Conference on Artificial Intelligence and Applications.* Citeseer, 258–263. https://doi.org/10.1.1.146.3650

[22] G. Martínez-Muñoz and A. Suárez. 2006. Pruning in ordered bagging ensembles. In *Proceedings of the 23rd International Conference on Machine Learning.* ACM, 609–616. https://doi.org/10.1145/1143844.1143921

[23] Ioannis Partalas, Grigorios Tsoumakas, Ioannis Katakis, and Ioannis Vlahavas. 2006. Ensemble pruning using reinforcement learning. In *Hellenic Conference on Artificial Intelligence.* Springer, 301–310. https://doi.org/10.1007/11752912_31

[24] Ioannis Partalas, Grigorios Tsoumakas, and Ioannis Vlahavas. 2008. Focused ensemble selection: A diversity-based method for greedy ensemble selection. *Frontiers in Artificial Intelligence and Applications* 178 (2008), 117–121. https://doi.org/10.3233/978-1-58603-891-5-117

[25] Ioannis Partalas, Grigorios Tsoumakas, and Ioannis Vlahavas. 2010. An ensemble uncertainty aware measure for directed hill climbing ensemble pruning. *Machine Learning* 81, 3 (2010), 257–282. https://doi.org/10.1007/s10994-010-5172-0

[26] G. Tsoumakas, I. Katakis, and I. Vlahavas. 2004. Effective voting of heterogeneous classifiers. In *European Conference on Machine Learning.* Springer, 465–476. https://doi.org/10.2147/JPR.S129139

[27] Y. Zhang, S. Burer, and W. N. Street. 2006. Ensemble pruning via semi-definite programming. *Journal of Machine Learning Research* 7, Jul (2006), 1315–1338. https://doi.org/10.1016/j.jasms.2006.06.007