Parsimony Search in Small High-dimensional Datasets with HYB-PARSIMONY

Jose Divasón[a], Alpha Pernia-Espinoza[b], Ana Romero[a], Francisco Javier Martinez-de-Pison[b,*]

[a] *Department of Mathematics and Computer Science, University of La Rioja, Logroño, Spain*
[b] *Scientific Computation and Technological Innovation Center (SCoTIC), University of La Rioja, Logroño, Spain*

Abstract

Achieving effective generalization in machine learning models is particularly challenging with small datasets that have high dimensionality. The combination of numerous features and few training instances often results in overfitting and poor performance on unseen data. This study conducts an in-depth analysis of HYB-PARSIMONY's performance on high-dimensional datasets and introduces a novel methodology that integrates HYB-PARSIMONY with Bayesian Optimization to address this issue through iterative feature and hyperparameter selection. The methodology employs HYB-PARSIMONY with multiple random seeds to identify features with the highest mean probability, followed by hyperparameter tuning using Bayesian Optimization to further enhance model performance. The experimental results demonstrate that this combined approach leads to models that are not only parsimonious but also capable of generalizing better compared to previous methods. By iteratively refining feature selection and hyperparameters, the proposed approach provides a more robust framework for building accurate machine learning models, even in challenging problems with a large number of features.

In conclusion, the integration of HYB-PARSIMONY and Bayesian Optimization significantly improves model generalization and reduces feature complexity, making it a promising methodology for small and high-dimensional datasets.

Keywords: HYB-PARSIMONY, small high-dimensional datasets, parsimonious modeling, auto machine learning, PSO-PARSIMONY, GA-PARSIMONY

1. Introduction

In many machine learning problems involving small and high-dimensionality datasets (SHDD), achieving models that can generalize effectively is a significant challenge. The combination of a high number of features with limited training instances often leads to overfitting, which hinders the model's ability to perform well on unseen data. Therefore, obtaining models that generalize well with SHDD is not an easy task. The curse of dimensionality coupled with the low number of instances cause many machine learning algorithms to have trouble describing the underlying structure of the data.

*Corresponding author
Email address: fjmartin@unirioja.es (Francisco Javier Martinez-de-Pison)

In this context, AutoML (Automated Machine Learning) frameworks have emerged as a promising solution to streamline the process of model selection, feature selection, and hyperparameter optimization. AutoML aims to automate these processes, allowing practitioners with limited machine learning expertise to develop competitive models. One such library, `AutoGluon` [1], has demonstrated success in building high-performing ensembles using robust validation methods and algorithms that perform well with high-dimensional datasets, such as tree-based models and neural networks (e.g., `XGBoost` [2], `LightGBM` [3], `CatBoost` [4], and deep neural networks). These models are capable of capturing complex relationships in data and mitigating the risks associated with overfitting to some extent. However, these approaches often lead to complex ensemble models, which can be challenging to interpret. Nowadays, explainability is crucial, particularly in fields such as healthcare, finance, and legal systems, where understanding the reasoning behind a model's predictions is as important as the predictions themselves [5]. This has led to an increasing demand from industries and decision-makers for simpler, more explainable models, even at the cost of a slight reduction in accuracy. For example, linear models or decision trees with a few well-defined rules are often preferred as they can provide more transparent insights into the decision-making process [6, 7].

The motivation behind our work is to address these challenges by developing a methodology that focuses on reducing model complexity while preserving, or even enhancing, predictive accuracy. Our proposed hybrid approach combining HYB-PARSIMONY with Bayesian Optimization (BO) aims to find a balance between parsimony and performance. HYB-PARSIMONY [8], a hybrid algorithm that integrates Particle Swarm Optimization (PSO) and Genetic Algorithms (GA), is particularly well-suited for this purpose, as it combines the strengths of both optimization methods to effectively navigate the search space for feature selection and hyperparameter tuning. By iteratively refining the set of features and optimizing hyperparameters through Bayesian Optimization, our approach aims to produce simpler, more interpretable models that retain competitive performance.

This paper is organized as follows: Section 2 reviews related work in feature selection and hyperparameter optimization, highlighting recent advances and their limitations. Section 3 analyzes the performance of HYB-PARSIMONY on high-dimensional datasets to assess its ability to balance parsimony and accuracy. The performance is evaluated using synthetic datasets with an intrinsic dimension lower than the real dimension of the dataset, and involves analyzing optimal hyperparameters based on both real and intrinsic dimensions. Additionally, HYB-PARSIMONY's performance is compared against a classical method based on Genetic Algorithms (GA) and another using Bayesian Optimization (BO) with all features, using 100 datasets. Section 4 outlines the strategy for working with SHDD, detailing the experimental setup, including dataset descriptions, evaluation metrics, and the results of the experiments. Finally, Section 5 concludes the paper with a summary of key findings and potential directions for future work.

2. Related works

Hyperparameter optimization (HO) and feature selection (FS) are important techniques in machine learning, because they can improve the accuracy of predictive models. However, determining the right hyperparameters and the most relevant subset of features can be a complex problem, especially when dealing with high-dimensional datasets.

Current approaches to solving combinatorial problems in machine learning often draw inspiration from nature, particularly from biological systems such as animal herding, bac-

terial growth, and other natural phenomena. These methods usually involve a population of simple individuals that interact both locally and globally with each other according to simple rules. For example, one such meta-heuristic approach is the Grey Wolf Optimizer (GWO), which was proposed by Mirjalili *et al.* [9] and was inspired by the behavior of grey wolves. The Salp Swarm Algorithm, also proposed by Mirjalili *et al.* in [10], was inspired by the swarming behavior of salps when navigating and foraging in oceans. Other techniques inspired by animals include bats [11], glowworm [12], and bee colony [13] optimization.

Particle Swarm Optimization (PSO) is one of the most commonly used optimization technique. Originally proposed by Kennedy and Eberhart [14], PSO has been the subject of much research, with numerous improvements proposed in terms of topology, parameter selection, and other technical modifications. For example, there are hybridizations of PSO with other meta-heuristic methods, such as the improved binary particle swarm optimization proposed by Chuang *et al.* [15], which uses the 'catfish effect' to introduce new particles into the search space if the best solution does not improve in a certain number of consecutive iterations.

Despite the success of these approaches, there are challenges associated with using meta-heuristic methods to solve combinatorial problems in machine learning. For example, GA-PARSIMONY was proposed in [16, 17] to search for parsimonious solutions with genetic algorithms (GA) by performing HO and FS, and was successfully applied in many fields [18, 19, 20]. However, in this kind of problems where each solution has a high computational cost, it is not possible to evaluate a large number of individuals in each iteration. This makes GA not as efficient as other optimization techniques where hundreds or thousands of individuals are evaluated. As a continuation of this methodology, the authors used PSO combined with a parsimony criterion to find parsimonious and accurate machine learning models. The main novelty in the PSO-PARSIMONY methodology [21] was that it included a strategy in which the best position of each particle was computed considering not only the goodness-of-fit but also the principle of parsimony. The comparison between both methods was performed on 13 public datasets, and the results showed that PSO always improved accuracy over GA, but GA found solutions approximately 10% less complex on datasets with a low number of features.

3. The HYB-PARSIMONY method with high-dimensional datasets

The HYB-PARSIMONY approach was proposed in [22, 8] by Divasón *et al.* and is a hybrid methodology that combines Particle Swarm Optimization (PSO) with Genetic Algorithm (GA) operations to produce parsimonious models in high-dimensional datasets.[1] By integrating selection, crossover, and mutation mechanisms from GA in the early stages, HYB-PARSIMONY accelerates the search of parsimony in models. As the iterations progress, PSO takes precedence, refining the model's accuracy. This synergy allows HYB-PARSIMONY to achieve a balance between accuracy and model simplicity, which is particularly advantageous for computationally intensive contexts, where both precision and interpretability are essential.

Pseudocode in Algorithm 1 details the iterative selection and crossover process, thus improving the integration between PSO and GA phases. This ensures that each iteration

[1]HYB-PARSIMONY is available for Python at https://github.com/jodivaso/HYBparsimony

Algorithm 1 Pseudo-code of the HYB-PARSIMONY algorithm [22]

Set initial positions $\mathbf{X^0}$ using a random and uniformly distributed Latin hypercube within the ranges of feasible values for each input parameter
Initialization of velocities according to $\mathbf{V^0} = \frac{random_{LHS}(s,\ D) - \mathbf{X^0}}{2}$
for $t = 1$ to T **do**
    Train each particle $\mathbf{X}_i^t$ and validate with cross-validation (CV)
    Evaluate fitness (J) and complexity (M_c) for each particle
    Update individual best $\hat{\mathbf{X}}_i$, individual parsimonious best $\hat{\mathbf{X}}_i^p$ and global best $\hat{\hat{\mathbf{X}}}$
    if early stopping criterion is met **then**
        return $\hat{\hat{\mathbf{X}}}$
    end if
    Generate new neighborhoods if $\hat{\hat{\mathbf{X}}}$ has not improved
    Update each neighborhood's best $\hat{\mathbf{L}}_i$
    Select elitist population P_e for reproduction
    Obtain a `pcrossover` % of worst individuals P_w to be substituted with crossover
    Crossover P_e to substitute P_w with new individuals
    Update positions and velocities of P_e following the PSO formulas
    Mutation of % of hyperparameters
    Mutation of % of features
    Limitation of velocities and out-of-range positions
end for
return global best $\hat{\hat{\mathbf{X}}}$

advances both the parsimony and accuracy of the model, providing a robust framework for balancing complexity and predictive performance in high-impact applications.

Although HYB-PARSIMONY demonstrated strong performance in previous experiments, a deeper investigation into its effectiveness on high-dimensional datasets with a low number of instances was essential. This section delves into this by analyzing HYB-PARSIMONY's capability to balance parsimony and accuracy under these challenging conditions. The evaluation employs synthetic datasets with an intrinsic dimension lower than the actual dimension, allowing for an assessment of optimal hyperparameters based on both real and intrinsic dimensions. Moreover, HYB-PARSIMONY's performance is rigorously compared with a traditional Genetic Algorithm (GA) approach and a Bayesian Optimization (BO) method utilizing all features, across 100 datasets. This study provides a comprehensive view of HYB-PARSIMONY's adaptability and robustness in handling the complexities of high-dimensional, low-instance data.

3.1. Performance of feature selection in high-dimensional datasets

In HYB-PARSIMONY, the following equation was proposed to calculate the percentage of particles to be substituted by GA crossover in each iteration t:

$$pcrossover = max(0.80 \cdot e^{(-\Gamma \cdot t)}, 0.10) \tag{1}$$

Figure 1 shows thirteen curves obtained with different Γ values. In the first iterations, the hybrid method performs the substitution by crossing a high percentage of particles. As the optimization process progresses, the number of substituted particles is reduced exponentially

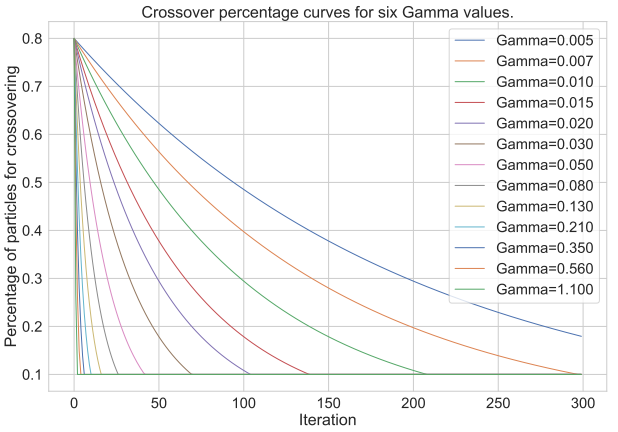

Figure 1: Example of thirteen curves created with different Γ values to establish the percentage of individuals to be replaced by crossover in each iteration.

until it ends up fixed at a percentage of 10%. Thus, the hybrid method begins by facilitating the search for parsimonious models using GA-based mechanisms and ends up using more PSO optimization.

To analyze the behavior of the hybrid method in high-dimensional datasets as a function of Γ, and with different dimensions and populations, a methodology was implemented with the following experiment's parameters:

- *method*: HYB vs previous methods (PSO or GA).
- *nruns*: number of runs with different random seeds. Value: 10.
- Γ (only for the hybrid method). Values: 0.005, 0.007, 0.010, 0.015, 0.020, 0.030, 0.050, 0.080, 0.130, 0.210, 0.350, 0.560, 1.100, 1.170.
- P: population size. Values: $[5,\ 5+1\cdot 5,\ 5+2\cdot 5,\ ...,\ 40]$.
- $\#feats$: dimension of the synthetic dataset. Values: 50, 150, 250, 350.
- i_{dim}: intrinsic dimension that refers to the features, F_{selec}, with relevant information present in a dataset. That is, the number of input features of the hypothetical model that explains an hypothetical target. Values: $5,\ 5+1\cdot 20,\ 5+2\cdot 20,\ ...,\ \lfloor 0.90\cdot \#feats \rfloor$.
- β: value which balances the weight between recall and precision in the F_{beta} score used to evaluate each individual (see below). Values: $[0.20,\ 0.20+1\cdot 0.06,\ 0.20+2\cdot 0.06,\ ...,\ 1.68]$.

For each combination of experiment's parameters, F_{selec} were randomly selected according to i_{dim}. In particular, F_{selec} corresponded to i_{dim} random feature positions selected within the range $[0, \#feats-1]$.

To evaluate each solution, F_{beta} score was used. Based on the $F1$ score, F_{beta} is the weighted harmonic mean of precision and recall where β determines the weight between

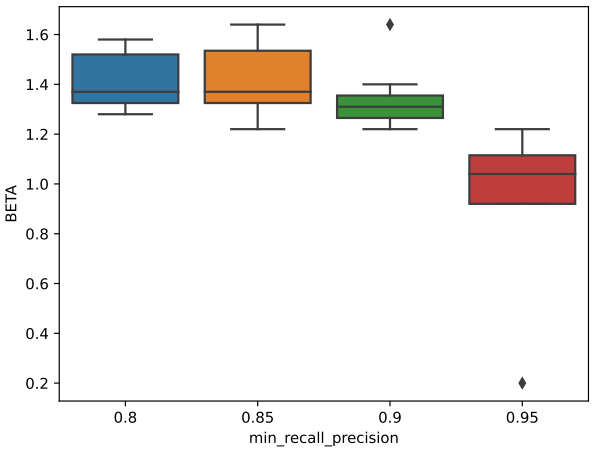

Figure 2: Distribution of the best β that successfully met the objectives of overcoming a minimum precision and recall defined by thr_{pr} (values = 0.80, 0.85, 0.90, 0.95).

recall and precision in the combined score. $\beta < 1$ gives more weight to precision, while $\beta > 1$ favors recall. F_{beta} is equal to $F1$ score with $\beta = 1.0$ and to precision with $\beta = 0.0$.

It is defined as:

$$F_{beta} = \frac{(1 + \beta^2)(precision \cdot recall)}{\beta^2 \cdot precision + recall} \tag{2}$$

and:

$$precision = \frac{TP}{TP + FP} \tag{3}$$

$$recall = \frac{TP}{TP + FN} \tag{4}$$

where TP are the correctly chosen features belonging to F_{selec}, TN the features not chosen and not belonging to F_{selec}, FP the features chosen but not belonging to F_{selec}, and FN the features not chosen but belonging to F_{selec}.

Each combination of [$method$ with Γ, P, $\#feats$, i_{dim} and β] was run 10 times with different randoms seeds, a maximum number of iterations of $T = 300$, $tol = 10^{-9}$, and an early stopping of 35.

All experiments[2] were implemented in 2 separately 40-core composed, respectively, of Intel(R) Xeon(R) CPU E5-2650 v3 @ 2.30GHz with 128 GB of RAM memory, and Intel(R) Xeon(R) CPU E5-2630 v4 @ 2.20GHz with 192 GB of RAM memory.

Figure 2 shows the number of experiments that successfully met the objectives of overcoming a minimum precision and recall defined by a threshold, thr_{pr}, and by each β and P values. The distribution of the best β is presented for each experiment and thr_{pr}. At low thr_{pr} values, the median of the best β for each combination of [$method$, Γ, P, $\#feats$, i_{dim}] is about 1.3. This indicates that precision tends to be prioritized over recall. Only at $thr_{pr} = 0.95$ it is observed that the median of the best β is close to 1.0, so the relationship between precision and recall is balanced when the level of demand is very high.

[2]The total number of experiments was 115170, resulting from all combinations.

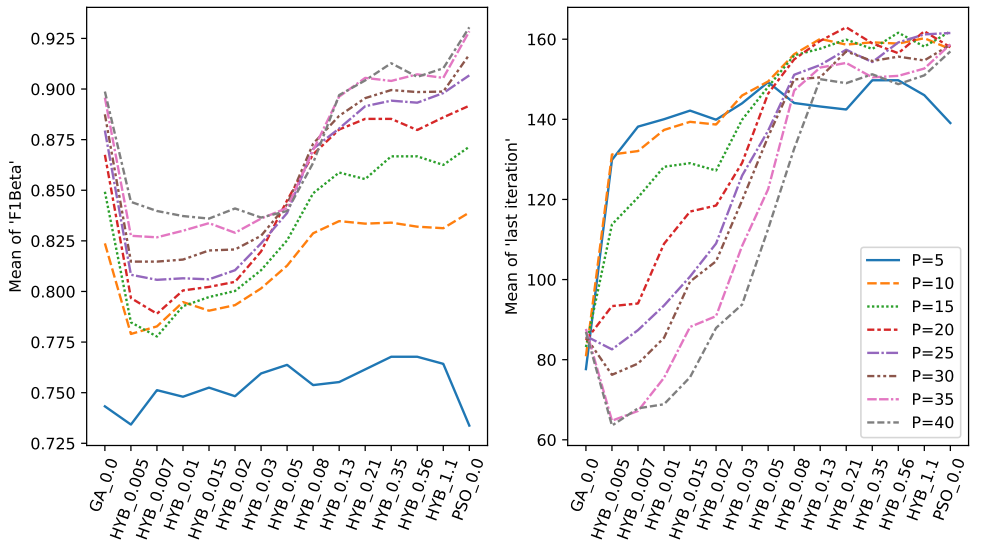

Figure 3: Mean of the F_{beta} (left) and last iteration ($last_{iter}$) (right) achieved with $\beta = 1.34$ and for each method and P.

Figure 3 shows respectively the mean of the last iteration ($last_{iter}$) and the average of F_{beta} with $\beta = 1.34$ and for each method and P. GA and HYB with low Γ converge, on average, faster than PSO and HYB with high Γ values, as they reach twice the number of final iterations. With respect to F_{beta}, the highest averages are obtained with PSO and HYB with Γ values greater than 0.08, but GA has similar performance to HYB_0.08.

However, these results are average values that may be different for each method, depending on $\#feats$ and i_{dim}. Figure 4 shows the distribution of F_{beta} for each method with $\#feats = 150$, $\beta = 1.34$ and four different i_{dim} values: 5, 45, 85 and 125. At very low values of i_{dim}, GA is competitive with the hybrid and PSO methods. However, since the intrinsic dimension is closer to the real dimension of the dataset, PSO and hybrid models with high Γ achieve better accuracy. The problem is that for a particular dataset the intrinsic dimension of the data will be unknown, so it will be necessary to realize an estimation of i_{dim} in order to select an appropriate method.

In order to have a quick estimate of the hybrid model for F_{beta} and $last_{iter}$, linear *Ridge* models were trained with the previously obtained dataset but eliminating instances corresponding to GA and PSO, and selecting only those cases with a β within the range $[0.92, 1.64]$ where the methodology was most successful. Equations 5 and 6 correspond to the best Ridge models selected with a 10-fold cross-validation RMSE error of 0.0815 and 57.36 with values of the *alpha* Ridge's hyperparameter equal to 4.0 and 2.0, respectively.

$$\hat{F_{beta}} = -0.0462 \cdot \Gamma - 0.0027 \cdot P + 0.0012 \cdot \#feats - 0.0011 \cdot i_{dim} - 0.0108 \cdot \beta - 0.88 \quad (5)$$

$$\hat{last_{iter}} = 28.391 \cdot \Gamma - 0.8883 \cdot P + 0.2963 \cdot \#feats - 0.38 \cdot i_{dim} + 36.517 \cdot \beta + 72.71 \quad (6)$$

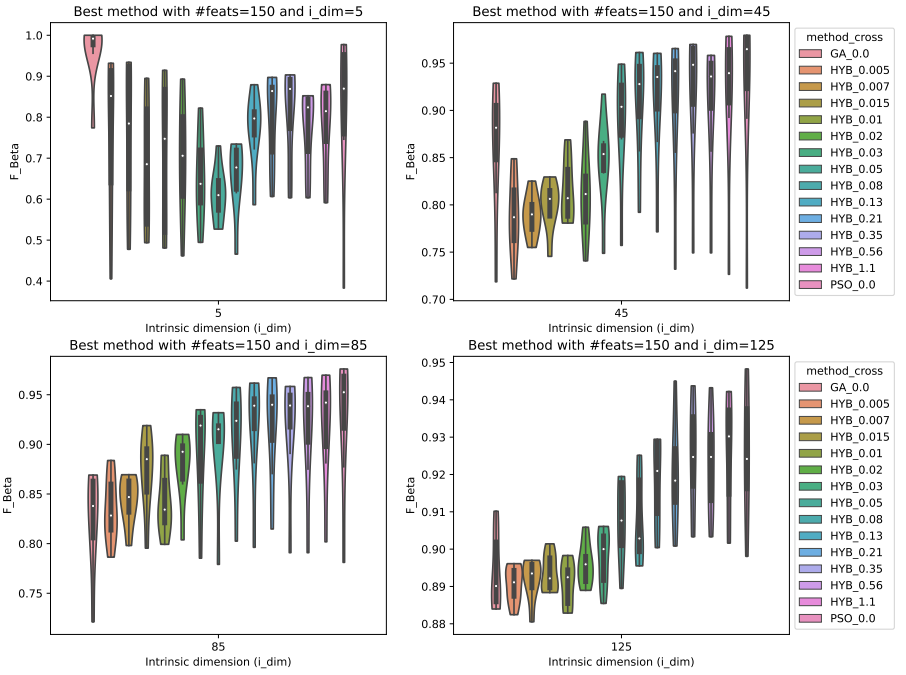

Figure 4: Violin plot of F_{beta} with $\beta = 1.34$, $\#feats = 150$ and for each method and four different i_{dim} values. The order of the violin plots in the legend (from top to bottom) corresponds to their position when viewed from left to right.

Using these two models, it was possible to approximately predict the values of F_{beta} and $last_{iter}$ for a data set dimension, $\#feats$, and a fixed value of P. Figure 5 shows the box plots for F_{beta} and $last_{iter}$ obtained from a simulation performed with multiple combinations of the input values for four high-dimensional data sets: *ailerons* ($\#feats = 40$), *crime* ($\#feats = 127$), *blog* ($\#feats = 276$) and *slice* ($\#feats = 378$). Each simulation was performed by fixing $P = 15$ and for each $\#ncols$, with $\Gamma \in [0.0, 0.0 + 0.1, ..., 1.1]$, $i_{dim} \in [0.10 \cdot \#feats, 0.10 \cdot \#feats + 5, ..., \#feats]$, and $\beta \in [0.92, 0.92 + 0.1, ..., 1.64]$. The graphs show the expected reduction that can be obtained in $last_{iter}$ *vs.* F_{beta} depending on the Γ used. However, estimates will be approximate and may vary greatly depending on the dataset (type and size[3]) and the machine learning algorithm used for modeling the problem.

3.2. Comparative of HYB-PARSIMONY vs Bayesian Optimization and a three-step method based on Genetic Algorithms

Figures 6 to 9 and Table 1 present a comparison of HYB-PARSIMONY with other two methods:

- Bayesian Optimization with all features (BO): $bo_{FINAL\_SCORE\_TST}$ represents the testing fitness value (J_{tst}) obtained using all input features ($bo_{NFS} = num\_cols$) and 250 iterations.

[3]As can be seen in Figure 5.

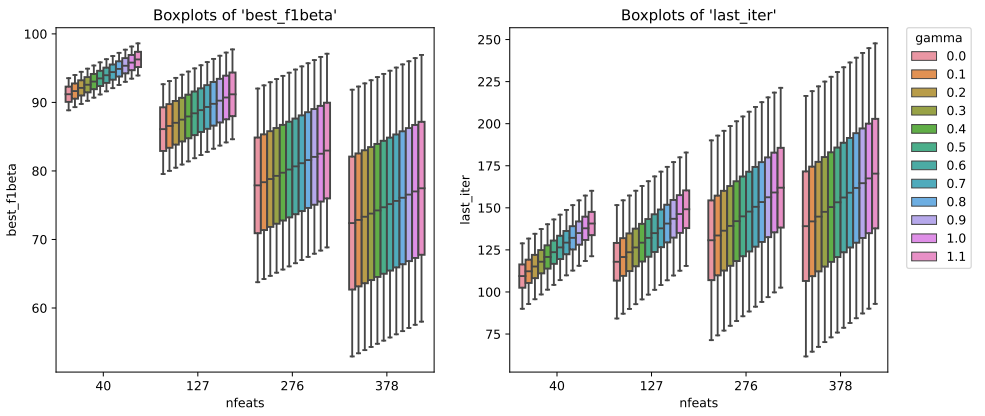

Figure 5: Box plots of F_{beta} (left) and $last_{iter}$ (right) for $\Gamma \in [0.0, 0.0 + 0.1, ..., 1.1]$ and four $\#feats$ values obtained by simulation with $P = 15$, and different i_{dim} and β values. The order of the boxplots in the legend (from top to bottom) corresponds to their position when viewed from left to right for each $\#feats$.

- HYB-PARSIMONY: hyb_{NFS} and $hyb_{FINAL\_SCORE\_TST}$ correspond to the number of features (NFS) and the fitness value (J_{tst}) obtained with each test dataset using HYB-PARSIMONY. HYB-PARSIMONY was executed with the following parameters: $\Gamma = 0.50$, $nruns = 250$, $time\_limit = 120min$, $P = 15$ and $early\_stopping = 35$. Additionally, *ReRank* was set to 0.001, representing the maximum allowable difference between the J values of two models to be considered equal.

- SKLEARN-GENETIC-OPT with a three-step methodology: first, performing hyperparameter optimization with BO using all features ($nruns = 250$); second, employing Genetic Algorithms (GA) from the *sklearn-genetic-opt* package for feature selection with the hyperparameters obtained in the first step and the following GA hyperparameters: $nruns = 250$ and $P = 15$; and finally, repeating the hyperparameter tuning with BO but using only the selected variables ($nruns = 250$). The columns ga_{NFS} and $ga_{FINAL\_SCORE\_TST}$ display the results obtained using $sklearn - genetic - opt$ with the three steps described above.

Results were obtained of 100 datasets from *openml.org* composed of 44 binary, 22 regression and 34 multiclass problems. Base ML algorithms were scikit-learn's *KernelRidge* for regression datasets, and scikit-learn's *LogisticRegression* for binary and multiclass datasets.

Table 1 shows the results for datasets with more than 100 features. As can be seen in both figures and tables, HYB-PARSIMONY obtains, in most cases, models with similar scores but with a significant reduction in the number of the selected features. However, HYB-PARSIMONY results with these datasets could be improved by adjusting parameters such as *Gamma*, *npart*, and others.

In these experiments[4], half of the instances from each dataset were used for training/validation, and the remaining half constituted the test dataset to assess model generalization. The results represent averages from five runs (with different random seeds) of

[4]All results and the code to replicate the experiments can be found at: https://github.com/jodivaso/hybparsimony/tree/master/examples/analysis

Table 1: Results with datasets with more than 100 columns.

$name\_db$	$trn-val_{size}$	$score$	bo_{NFS}	hyb_{NFS}	ga_{NFS}	bo_{tst}	hyb_{tst}	ga_{tst}
dna	1593	f1_macro	360	**156.2**	181.4	.942	**.944**	.943
ames_housing	1465	RMSE	354	**178.8**	187	**.442**	.520	.525
philippine	2916	logloss	308	**62.6**	151.6	.557	**.542**	.552
scene	1203	logloss	304	**49.2**	142.6	.200	**.061**	.108
splice	1595	f1_macro	287	**146.6**	157.4	.941	**.943**	.933
jasmine	1492	logloss	280	**71.4**	141	**.445**	.448	.446
topo_2_1	4442	RMSE	266	**68.8**	138.6	.925	.924	**.923**
madeline	1570	logloss	259	**54**	126.8	**.670**	.680	.672
USPS	4649	f1_macro	256	**135.8**	136.6	**.940**	.938	.937
yprop_4_1	4442	RMSE	251	**84.6**	121.4	**.916**	.925	.918
autoUniv-au4-2500	1250	f1_macro	226	**93**	109.8	.397	**.402**	.397
Indian_pines	4572	f1_macro	220	**105.2**	108.4	**.838**	.832	.826
mtp	2225	RMSE	202	**70.2**	110.4	.694	.688	**.687**
clean2	3299	logloss	168	**36**	73.8	**.000**	**.000**	**.000**
COIL2000-train	2911	logloss	150	**35.6**	77.2	**.210**	.213	.211
ECG5000	2499	logloss	140	**29.4**	68.6	**.044**	.050	**.044**
kings_county	10806	RMSE	132	97	**79.8**	**.357**	.374	.370
SAT11-HAN-rr	2220	RMSE	130	**48.4**	77	.518	.514	**.511**
yeast_ml8	1208	logloss	129	**38**	66.5	**.067**	.114	.072
gas-drift	6955	f1_macro	128	**56.8**	67.4	.988	**.989**	**.989**
mushroom	4062	logloss	125	**59.4**	70.4	**.000**	**.000**	**.000**
sylva_prior	7197	logloss	108	**55.8**	61.2	**.018**	.020	.019
BachChoralH.	2832	f1_macro	104	81.4	**61.8**	**.405**	.388	.371
ada_prior	2281	logloss	102	**43.4**	56.6	**.353**	.356	**.353**

each methodology and dataset. All methods utilized 5-fold cross-validation.

The left sides of figs. 6 to 8 show, for the three groups of problems: binary (Figure 6), regression (Figure 7) and multiclass (Figure 8), the number of features selected by HYB-PARSIMONY (hyb_NFS) and the classical method (ga_NFS) versus the total number of variables used by BO (bo_NFS). The right sides show the score with the test dataset (*logloss*, *RMSE* and *F1-Macro*) of the best model obtained with each method and dataset.

Finally, the Figure 9 below shows that the computation time with HYB-PARSIMONY was considerably reduced compared to the feature selection method with GA and close to BO (although, to be fair, in the GA method no early stopping was implemented).

4. Strategy for working with SHDD

Creating accurate models with SHDD is a current challenge. If the dataset has hundreds or a few thousand instances, and the dimension is high (several tens or hundreds of features), the search for models that correctly generalize the problem will face two fundamental problems: the curse of dimensionality and an excessive over-fitting in the optimization process.

Although there are algorithms, such as trees and neural networks, that may be less affected by the curse of dimensionality, in these cases it is highly recommended to use feature selection or dimensional reduction. In addition, the regularization included in machine

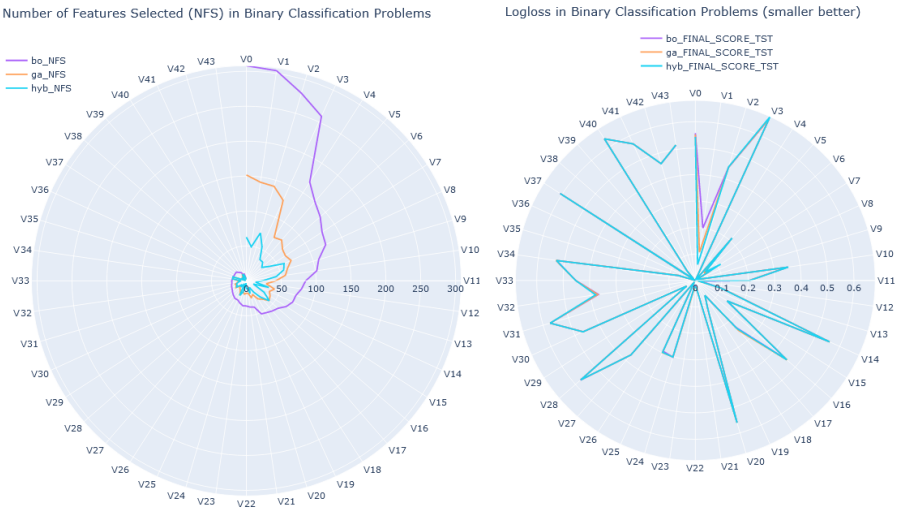

Figure 6: Selected Features (NFS) and *LogLoss* with 44 Binary Datasets.

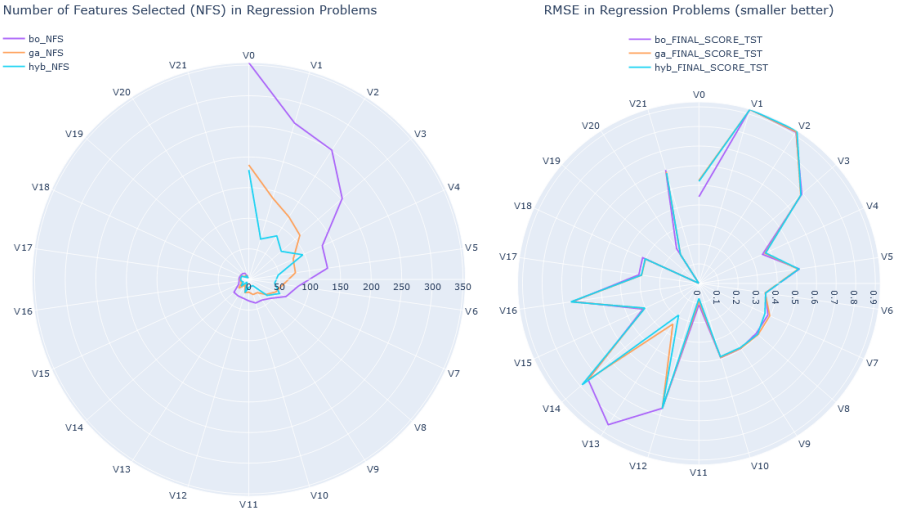

Figure 7: Selected Features (NFS) and *RMSE* with 22 Regression Datasets.

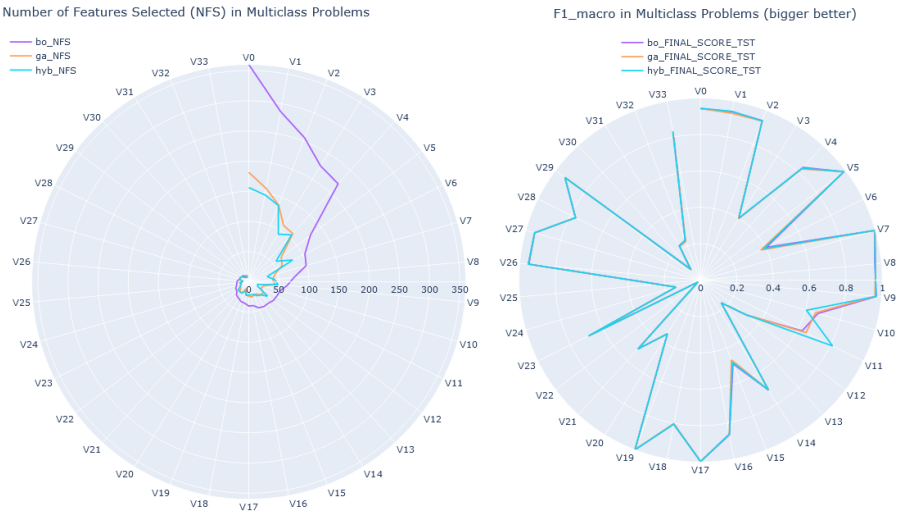

Figure 8: Selected Features (NFS) and *F1-Macro* with 34 Multiclass Datasets

learning algorithms helps penalize models that are too complex and with high probability of overfitting. The proposed hybrid methodology greatly facilitates both aspects since it seeks to reduce as much as possible the number of features of the selected model, as well as its internal complexity.

However, HYB-PARSIMONY is such an intensive search method that when working with SHDD the method may find parsimonious solutions that are too specific to that set of instances. Thus, the chosen hyperparameters and the selected features may be the most appropriate for that sample but not be sufficient to create a model that will generalize correctly in the future. To reduce this over-fitting and to find a feature selection that can be used to create a robust model that generalizes correctly in this kind of problems, we propose the following methodology:

1. Repeat n runs with different random seeds the search for the best model with HYB-PARSIMONY and hold-out validation. In each repetition, extract the feature probability vector of the best individual. The use of hold-out validation increases the diversity of validation samples employed in feature selection. By repeating the process multiple times with different random seeds, we ensure that the features selected most frequently are those that consistently contribute valuable information across a broader and more varied set of samples. This approach enhances the robustness of the selected features, as they are more likely to generalize well when faced with diverse data.
2. Average the probabilities for each feature and select those that have a value greater than a given threshold, thr_{fs}.
3. Use BO to perform hyperparameter tuning with the features selected in the previous point.
4. Repeat points 2 and 3 with different thr_{fs}.
5. Select the model that obtains the best error J with another test dataset.

Table 2 shows the results with 13 high-dimensional datasets of using the described methodology versus using BO with all features ($\#feats$). In these experiments, 2000 rows

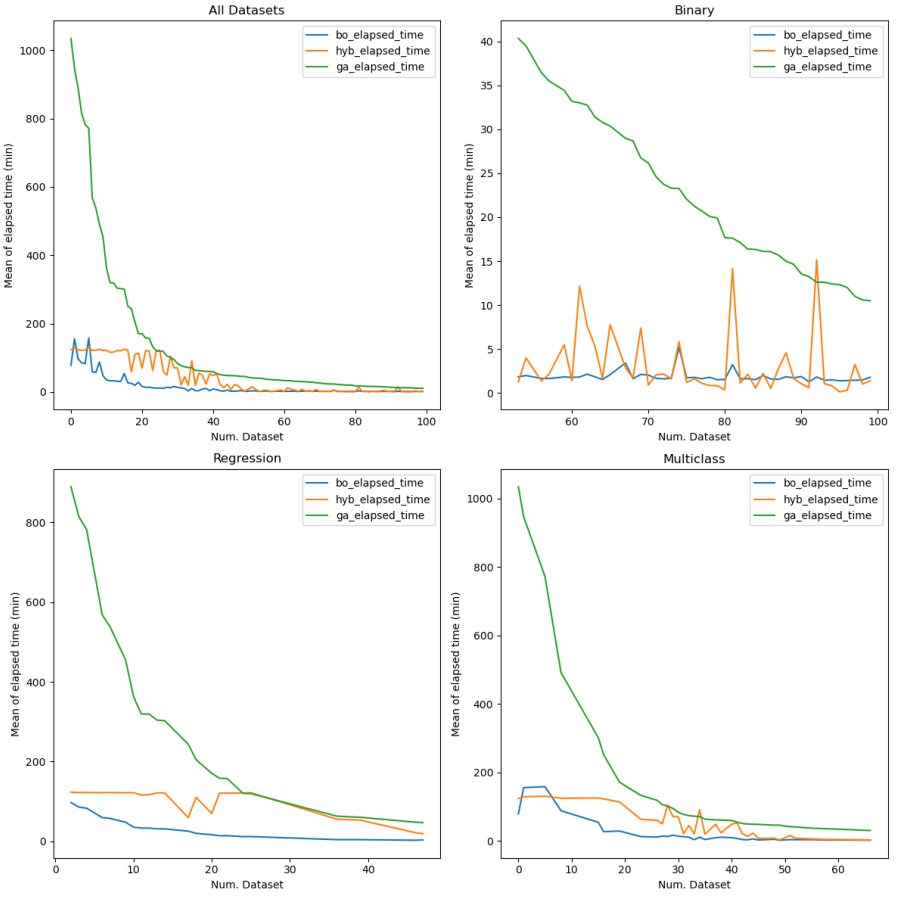

Figure 9: Elapsed times with all datasets and for each kind of problem (binary, regressión and multiclass).

Table 2: Results for 13 datasets with BO (J) versus HYB-PARSIMONY with $thr_{fs} = 0.50$ followed by BO.

Dataset				BO	HYB-PARSIMONY and BO		
name	$train_{size}$	$test_{size}$	$\#feats$	J_{BO}	$last_{iter}$	$J0.5$	$Fs0.5$
slice	2000	23000	378	**.1414**	165.8 (42.4)	.1449 (.0022)	148.0 (3.74)
blog	2000	50397	276	.8216	178.6 (45.6)	**.8154 (.0170)**	67.0 (17.42)
crime	1107	1108	127	**.6373**	199.0 (0.0)	.6379 (.0076)	28.8 (4.32)
ailerons	2000	11750	40	.3984	131.2 (17.9)	**.3982 (.0016)**	13.2 (2.05)
bank	2000	6192	32	.6745	160.0 (32.1)	**.6726 (.0023)**	15.6 (1.52)
puma	2000	6192	32	.8762	106.2 (16.3)	**.2006 (.0250)**	3.6 (0.89)
pol	2000	13000	26	.3165	148.0 (32.2)	**.2413 (.0034)**	7.2 (0.45)
sat11	2000	2440	130	**.5329**	191.6 (10.9)	.5444 (.0070)	46.6 (3.44)
ames	1465	1465	354	**.3939**	192.2 (13.6)	.4997 (.0334)	174.2 (13.50)
kings	2000	19614	132	.3883	176.6 (37.7)	**.3874 (.0046)**	93.8 (8.53)
mtp	2000	2450	202	**.6874**	197.4 (3.6)	.7045 (.0059)	49 (10.17)
topo	2000	6885	266	**.9202**	192.6 (14.3)	.9265 (.0031)	42.6 (3.21)
yprop	2000	6885	251	**.9207**	199.0 (0.0)	.9231 (.0035)	62.6 (11.26)

were selected for training/validation (except `crime` where half of them were used) and the rest were utilized as a test dataset to verify the degree of generalization of the models. J_{BO} corresponds to the testing RMSE error obtained from a model that used all the input features and whose hyperparameters were adjusted by BO. The last three columns corresponds to the new proposal. First, 25 runs of HYB-PARSIMONY were performed with $\Gamma = 0.50$, $nruns = 200$, $P = 15$, $early\_stopping = 35$, hold-out validation with a 20%, and $KernelRidge$ as ML algorithm. Finally, $ReRank$ was set to 0.001 which corresponds to the maximum difference between the J of two models to be considered equal. A high value of this parameter facilitates the search for parsimony in HYB-PARSIMONY because between two models with a similar J the less complex model is selected. Next, hyperparameter tuning with BO was done of a $KernelRidge$ algorithm with the features whose probabilities were greater o equal than 0.50 ($thr_{fs} = 0.50$). Columns in table indicate: the last iteration ($last_{iter}$) of HYB-PARSIMONY, RMSE ($J0.5$) and the number of features used ($Fs0.5$) in the final model. The results correspond to the average values and the standard deviation (in parentheses) of 5 runs of the whole methodology with different random seeds.

As can be seen in Table 2, the proposed methodology obtained a significant reduction in the average number of final selected features, being the differences between J_{BO} and $J0.5$ not too high. The average reduction of features was considerable, reducing the number of selected features by more than 50% in 12 of the 13 datasets. In some cases, the reduction was over 75%. For example, in `blog` the average number of features was reduced to 24.3% (from 276 to 67, a 75.7% reduction), in `crime` to 22.6%, in `mtp` to 24.3% and in `yprop` to 25.0%. However, BO obtained more accurate models in 7 of the 13 datasets, although these results could be improved by using different thr_{fs} as shown in Table 3. By using other thresholds, we can seek a compromise between the complexity of the model (number of features) and the final accuracy of the model. As can be seen in Table, more parsimonious and accurate models were obtained in all datasets except `mtp`.

Table 4 shows a comparative analysis of the proposed methodology using HYB-PARSIMONY versus the previous method, PSO-PARSIMONY. The best results are shown in bold where

Table 3: Proposed methodology with different thr_{fs} *vs.* BO with all features (J_{BO}).

dataset	J_{BO}	J0.2	Fs0.2	J0.3	Fs0.3	J0.4	Fs0.4	J0.5	Fs0.5	J0.6	Fs0.6	J0.7	Fs0.7
slice	.1414	.1392	346.4	.1374	294.2	**.1370**	**228.2**	.1449	148.0	.1583	85.8	.2267	40.2
blog	.8216	.8265	247.0	.8288	196.8	.8234	128.8	**.8154**	**67.0**	1.014	29.4	.9462	10.8
crime	.6373	**.6367**	**108.8**	.6371	85.0	.6386	56.4	.6379	28.8	.6461	12.8	.7867	3.8
ailerons	.3984	.3986	33.2	.3994	25.4	.3993	17.8	**.3982**	**13.2**	.4338	9.8	.5043	5.6
bank	.6745	.6729	29.4	.6756	24.6	.6744	20.4	**.6726**	**15.6**	.6791	11.4	.6889	9.0
puma	.8762	.3030	12.2	.2308	7.6	.2096	4.6	**.2006**	**3.6**	.2007	3.6	.2230	2.8
pol	.3165	.2736	13.2	.2584	10.4	.2461	8.2	.2413	7.2	.2413	7.2	**.2387**	**6.4**
sat11	.5329	**.5328**	**127.2**	.5363	102	.5383	79.6	.5444	46.6	.5781	28.6	.6193	14.2
ames	.3939	**.3917**	**351.4**	.4177	308	.4483	252.8	.4997	174.2	.4982	103.8	.5601	53.2
kings	.3883	.3878	130.6	.3877	126.8	**.3856**	**114.4**	.3874	93.8	.3928	71	.4091	45.8
mtp	**.6874**	.6880	178	.6899	137.2	.6930	90.4	.7045	49	.7318	25	.8181	9.2
topo	.9202	**.9200**	**222.4**	.9208	165	.9226	95.8	.9265	42.6	.9286	18.4	.9350	6.2
yprop	.9207	.9198	217.8	**.9198**	**172.4**	.9214	115.4	.9231	62.6	.9283	28.2	.9346	8.8

Table 4: Comparison of HYB-PARSIMONY and PSO-PARSIMONY with the best thr_{fs}. Best results of J and F_s are highlighted in bold where statistically significant differences are observed at a 95% confidence level.

	HYB-PARSIMONY				PSO-PARSIMONY			
dataset	$last_{iter}$	thr_{fs}	J	Fs	$last_{iter}$	thr_{fs}	J	Fs
slice	**165.8**	.4	**.1370(.003)**	**228.2(6.0)**	182.8	.5	.1372(.002)	239.0(6.9)
blog	**178.6**	.5	**.8154(.017)**	**67.0(17.4)**	191.6	.0	.8215(.000)	276.0(0.0)
crime	199.0	.2	**.6367(.002)**	**108.8(1.3)**	**175.4**	.1	.6371(.000)	124.8(1.3)
ailerons	**131.2**	.5	**.3982(.002)**	**13.2(2.1)**	154.8	.5	.3984(.002)	16.0(3.5)
bank	160.0	.5	.6725(.002)	**15.6(1.5)**	**149.8**	.5	.6724(.002)	16.8(1.6)
puma	106.2	.5	.2006(.025)	**3.6(0.9)**	**104.4**	.4	**.1894(.000)**	4.0(0.0)
pol	148.0	.7	.2387(.004)	6.4(0.6)	**127.2**	.7	**.2374(.003)**	6.2(0.5)
sat11	**191.6**	.2	.5328(.001)	127.2(2.3)	198.0	.3	.5328(.003)	**103.0(4.7)**
ames	**192.2**	.2	**.3917(.005)**	**351.4(4.0)**	197.4	.2	.3920(.007)	352.2(2.7)
kings	**176.6**	.4	**.3856(.002)**	**114.4(6.2)**	188.8	.3	.3879(.003)	126.0(4.6)
mtp	197.4	.0	.6874(.000)	202.0(0.0)	**145.6**	.2	.6873(.002)	**187.8(6.5)**
topo	**192.6**	.2	.9200(.000)	222.4(7.8)	197.6	.2	.9199(.001)	**217.8(4.8)**
yprop	199.0	.3	**.9189(.001)**	**172.4(6.8)**	**190.4**	.3	.9195(.001)	184.4(8.0)

there are statistically significant differences with a confidence factor of 95%.

The results show that the new methodology improved J in 7 high dimensionality datasets and in a statistically significant way. Likewise, it obtained an improvement in the reduction of the number of features in 9 of the 13 datasets. The most significant reduction was observed in *blog*, although there were substantial improvements in *slice*, *crime*, *ailerons*, *kings* and *yprop*. However, PSO-PARSIMONY obtained better J in *puma* and *pol*, but with worse parsimony in the first one. On the other hand, in models where J was similar, PSO-PARSIMONY obtained more parsimonious models, although in these three cases the number of final iterations ($last_{iter}$) was close to the 200 limit, which may indicate that with these datasets the optimization would have needed more iterations.

Finally, Table 5 shows a comparison of the proposed methodology versus the use of the `sklearn-genetic-opt` library for feature selection and hyperparameter tuning. The datasets have been sorted according to the number of features. To use `sklearn-genetic-opt` for feature selection and hyperparameter fitting (similar to that performed by HYB-PARSIMONY), three steps were performed: perform the hyperparameter fitting with GA, select with GA the best features and finally perform again the hyperparameter fitting with GA but using only the selected variables. The results correspond to the average of 10 runs of the proposed methodology with different random seeds and selecting the best thr_{fs}. Only those with statistical significance ($p < 0.05$ value in the Wilcoxon–Mann–Whitney test) are shown in bold.

HYB-PARSIMONY obtained J improvements in 6 of the 12 databases, mainly in the lower dimensionality datasets. When analyzing the last three columns related to the reduction of the number of features and, fundamentally in those datasets with $\#feats > 100$, it can be observed that the methodology for working with SHDD avoided a drastic reduction of the features that would lead to an overfitting of the models. For example, the complexity of the `ames`, `topo` or `kings` models was not reduced much. However, in other cases such as `slice`, `yprop`, `blog` and `crime`, the feature reduction was significant.

Finally, it is important to note that all HYB-PARSIMONY tests have been performed with $\Gamma = 0.50$, although other values of this hyperparameter can be used to find better results. Likewise, the results in the Table 5 correspond to the thr_{fs} value that obtained lower J with a new test database. However, the methodology allows the use of any other value of thr_{fs} that improves parsimony at the expense of slightly reducing the accuracy of the models.

5. Conclusions

GA-PARSIMONY, PSO-PARSIMONY, and HYB-PARSIMONY are methodologies that have been developed for the search of accurate yet low complexity machine learning models. However, an intensive search with these methods in Small High-Dimensional Datasets (SHDD) can lead to overfitted models. Specifically, HYB-PARSIMONY is such an intensive search method that, when working with SHDD, it may find parsimonious solutions that are overly specific to the training data. As a result, the selected hyperparameters and features may be optimal for that sample but not sufficient to create a model that generalizes well to new data. To reduce this overfitting and to find a feature selection that can be used to create a robust model that generalizes correctly in this kind of problem, we proposed a new methodology in this paper.

The proposed methodology is based on repeating HYB-PARSIMONY with different random seeds and using hold-out validation. In this way, the search for the best model is

Table 5: Comparison of HYB-PARSIMONY and `sklearn-genetic-opt` for feature selection and hyperparameter tuning across datasets ordered by feature count. Results show the average of 10 runs, with statistically significant differences ($p < 0.05$, Wilcoxon–Mann–Whitney test) highlighted in bold.

dataset	$\#feats$	J_{hyb}	J_{gen}	p-value	Fs_{hyb}	Fs_{gen}	p-value
slice	378	**0.131430**	0.142033	0.000330	**183.3**	202.4	0.000489
ames	354	0.444826	0.432153	1.000000	**328.2**	344.2	0.011667
blog	276	**0.823028**	0.841876	0.002827	**69.8**	143.6	0.000085
topo	266	0.918339	0.918139	0.690476	257.6	251.4	0.057008
yprop	251	0.917062	0.917360	0.420635	116.4	121.0	0.547619
mtp	202	0.687080	0.686763	0.690476	172.0	168.8	0.399075
kings	132	0.370370	0.369793	0.690476	129.2	**126.4**	0.033895
sat11	130	0.519502	0.520955	0.222222	**110.0**	124.2	0.011667
crime	127	**0.630145**	0.633698	0.011330	70.7	67.7	0.148308
ailerons	40	**0.398032**	0.398548	0.025748	40.0	**22.9**	0.000062
bank	32	**0.672656**	0.676080	0.021134	24.7	**19.8**	0.000143
puma	32	**0.189436**	0.258673	0.000183	**4.0**	11.0	0.000048
pol	26	**0.236990**	0.276652	0.000183	**6.0**	15.7	0.000058

validated with different parts of the dataset at each run. Averaging the feature probability vectors from multiple runs allows for a more robust selection of the final features, reducing the risk of overfitting while improving model generalization. Once these features are selected, using different thresholds, Bayesian Optimization is used to tune the hyperparameters of the model. Although the same methodology can be applied using other feature selection and/or hyperparameter optimization methods, the use of HYB-PARSIMONY significantly facilitates feature reduction (the search for parsimony), which is a critical factor in obtaining low-complexity, interpretable models.

Results have demonstrated that it is possible to obtain more accurate models with a significant reduction in the number of features compared to other methodologies. This allows the development of models that not only generalize well but also maintain lower complexity, making them suitable for practical, interpretable use in SHDD.

One limitation of the proposed methodology is that it must be tested with various thresholds (thr_{fs}), as results can vary considerably depending on the threshold chosen. Moreover, altering other hyperparameters of HYB-PARSIMONY increases the likelihood of finding suitable solutions, but at the cost of significantly increasing the computational time required for the search. Therefore, there is a trade-off between computational efficiency and model performance.

Future research will focus on developing methods to accurately estimate hyperparameters such as the threshold for feature selection (thr_{fs}), Γ (which affects the crossover rate), and other HYB-PARSIMONY hyperparameters according to the specific characteristics of each dataset. Additionally, further studies will investigate alternative validation processes to accelerate the search for the most robust features for SHDD. Such approaches aim to enhance computational efficiency while preserving the reliability of feature selection and the accuracy of the resulting models.

GA-PARSIMONY, PSO-PARSIMONY and HYB-PARSIMONY are methodologies that have been developed for the search of accurate but low complexity ML models. However, an intensive search with these methods in SHDD can lead to overfitted models.

The proposed methodology is based on repeating HYB-PARSIMONY with different random seeds and by using hold-out validation. In this way, at each run the search for the best model is validated with a different part of the dataset. Averaging the feature probability vectors allows one to make a more robust selection of the final features. Once these are selected, with different thresholds, BO is used to fit the hyperparameters of the model.

Results demonstrated that it is possible to obtain more accurate models with a significant reduction in the number of features against other methodologies.

5.0.1. Acknowledgements

The work is supported by grant PID2020-116641GB-I00 and by Project PID2021-123219OB-I00, funded by MICIU/AEI/10.13039/501100011033 and by ERDF/EU. Work also supported by project Inicia 2023/02 funded by La Rioja Government (Spain). We are also greatly indebted to Banco Santander for the REGI2020/41 and REGI2022/60 fellowships.

References

[1] N. Erickson, J. Mueller, A. Shirkov, H. Zhang, P. Larroy, M. Li, A. Smola, Autogluon-tabular: Robust and accurate automl for structured data, arXiv preprint arXiv:2003.06505 (2020).

[2] T. Chen, C. Guestrin, XGBoost: A scalable tree boosting system, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD'16, ACM, New York, NY, USA, 2016, pp. 785–794. doi:10.1145/2939672.2939785.

[3] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, T.-Y. Liu, Lightgbm: A highly efficient gradient boosting decision tree, Advances in neural information processing systems 30 (2017) 3146–3154.

[4] A. V. Dorogush, V. Ershov, A. Gulin, Catboost: gradient boosting with categorical features support, 2018. arXiv:1810.11363.

[5] F. Doshi-Velez, B. Kim, Towards a rigorous science of interpretable machine learning, arXiv preprint arXiv:1702.08608 (2017). URL: https://arxiv.org/abs/1702.08608.

[6] M. T. Ribeiro, S. Singh, C. Guestrin, "why should i trust you?": Explaining the predictions of any classifier, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2016, pp. 1135–1144. doi:10.1145/2939672.2939778.

[7] S. M. Lundberg, S.-I. Lee, A unified approach to interpreting model predictions, in: I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, R. Garnett (Eds.), Advances in Neural Information Processing Systems 30, Curran Associates, Inc., 2017, pp. 4765–4774. URL: https://proceedings.neurips.cc/paper/2017/file/8a20a8621978632d76c43dfd28b67767-Paper.pdf.

[8] J. Divasón, A. Pernia-Espinoza, F. J. M. de Pison, Hyb-parsimony: A hybrid approach combining particle swarm optimization and genetic algorithms to find parsimonious models in high-dimensional datasets, Neurocomputing 560 (2023) 126840. doi:10.1016/j.neucom.2023.126840.

[9] S. Mirjalili, S. M. Mirjalili, A. Lewis, Grey wolf optimizer, Advances in Engineering Software 69 (2014) 46–61. doi:10.1016/j.advengsoft.2013.12.007.

[10] S. Mirjalili, A. H. Gandomi, S. Z. Mirjalili, S. Saremi, H. Faris, S. M. Mirjalili, Salp swarm algorithm: A bio-inspired optimizer for engineering design problems, Advances in Engineering Software 114 (2017) 163–191. doi:10.1016/j.advengsoft.2017.07.002.

[11] X.-S. Yang, A new metaheuristic bat-inspired algorithm, in: Nature inspired cooperative strategies for optimization (NICSO 2010), Springer, 2010, pp. 65–74.

[12] M. Marinaki, Y. Marinakis, A glowworm swarm optimization algorithm for the vehicle routing problem with stochastic demands, Expert Systems with Applications 46 (2016) 145–163. doi:10.1016/j.eswa.2015.10.012.

[13] D. Karaboga, B. Basturk, Artificial bee colony (abc) optimization algorithm for solving constrained optimization problems, in: P. Melin, O. Castillo, L. T. Aguilar, J. Kacprzyk, W. Pedrycz (Eds.), Foundations of Fuzzy Logic and Soft Computing, Springer Berlin Heidelberg, Berlin, Heidelberg, 2007, pp. 789–798.

[14] J. Kennedy, R. Eberhart, Particle swarm optimization, in: Proceedings of ICNN'95 - International Conference on Neural Networks, volume 4, 1995, pp. 1942–1948 vol.4. doi:10.1109/ICNN.1995.488968.

[15] L.-Y. Chuang, S.-W. Tsai, C.-H. Yang, Improved binary particle swarm optimization using catfish effect for feature selection, Expert Syst. Appl. 38 (2011) 12699–12707. doi:10.1016/j.eswa.2011.04.057.

[16] F. J. Martinez-de Pison, R. Gonzalez-Sendino, A. Aldama, J. Ferreiro-Cabello, E. Fraile-Garcia, Hybrid methodology based on bayesian optimization and ga-parsimony to search for parsimony models by combining hyperparameter optimization and feature selection, Neurocomputing 354 (2019) 20–26. doi:10.1016/j.neucom.2018.05.136.

[17] F. J. Martinez-de Pison, J. Ferreiro, E. Fraile, A. Pernia-Espinoza, A comparative study of six model complexity metrics to search for parsimonious models with GAparsimony R Package, Neurocomputing 452 (2021) 317–332. doi:10.1016/j.neucom.2020.02.135.

[18] F. Antonanzas-Torres, R. Urraca, J. Antonanzas, J. Fernandez-Ceniceros, F. J. Martinez-de Pison, Generation of daily global solar irradiation with support vector machines for regression, Energy Conversion and Management 96 (2015) 277–286. doi:10.1016/j.enconman.2015.02.086.

[19] E. Dulce-Chamorro, F. J. M. de Pison, An advanced methodology to enhance energy efficiency in a hospital cooling-water system, Journal of Building Engineering 43 (2021) 102839. doi:10.1016/j.jobe.2021.102839.

[20] J. Divasón, F. J. Martínez-de Pisón, A. Romero, E. Sáenz-de Cabezón, Artificial intelligence models for assessing the evaluation process of complex student projects, IEEE Transactions on Learning Technologies 16 (2023) 694–707. doi:10.1109/TLT.2023.3246589.

[21] J. Divasón, J. Fernandez-Ceniceros, A. Sanz-Garcia, A. Pernia-Espinoza, F. J. Martinez-de Pison, PSO-PARSIMONY: A method for finding parsimonious and accurate machine learning models with particle swarm optimization. Application for predicting force–displacement curves in T-stub steel connections, Neurocomputing 548 (2023) 126414. doi:10.1016/j.neucom.2023.126414.

[22] J. Divasón, A. Pernia-Espinoza, F. J. Martinez-de Pison, New hybrid methodology based on particle swarm optimization with genetic algorithms to improve the search of parsimonious models in high-dimensional databases, in: Hybrid Artificial Intelligent Systems, Springer International Publishing, Cham, 2022, pp. 335–347.