Variable Importance Estimation for High-Dimensional Optimisation

Kelly Hunter, Sarah L. Thomson, and Emma Hart

SCEBE, Edinburgh Napier University, Edinburgh, UK

Abstract. Machine learning models trained on the solution spaces of optimisation problems can potentially shed light on variable importance. In prior work the recently established combinatorial benchmark, Polynomial Unconstrained Binary Optimisation with variable importance (PUBO_i) was used in this way. Small search spaces were considered so that it was possible to fully enumerate as well as sample. The study confirmed that explainable artificial intelligence (XAI) feature attribution methods can detect these ground-truth importances in this combinatorial optimisation problem. In the present work, we consider larger problem dimensions with the aim of establishing whether the results and XAI methods scale. We compare the distributions of important and unimportant variables across PUBO_i instances for prevalent XAI methods to uncover how well important variables are captured. We found that in high-dimensional instances the important variables were captured but to a lesser extent than in low-dimensional instances. The analysis will help to inform future work in adapting search operators during optimisation.

Keywords: explainable artificial intelligence, variable importance, pseudo-boolean optimisation

1 Introduction

With the growing use of black-box optimisation, pinpointing which decision variables are important is of interest. A promising path to understanding this lies in explainable artificial intelligence (XAI) [3,16], which is a rapidly advancing paradigm that aims to open the 'black box' of machine learning models. Within XAI, feature attribution techniques quantify the contribution of individual features to machine learning models, thereby elucidating the drivers of the model [6]. In this work, we extend previous literature [7] exploring the potential for feature attribution methods to estimate variable importance in combinatorial optimisation problems. The search space is sampled and machine learning models — with solutions as input and their fitness as the response variable — are trained. Then, XAI feature attribution techniques are used. The recently proposed PUBO_i benchmark [20], which has tunable ground-truth variable importances, is considered. The problem dimension in the previous work [7] was set deliberately small at 14. The present work tests whether the approach scales when the problem dimension is much larger and also whether lowering the budget for sampling the space is effective. The goal of this work is to test whether

the same explainers provide an accurate view of importance when given a low budget in high dimensions.

The principal contribution of this work is this: *investigation of whether XAI methods can accurately identify important variables in high-dimensional optimisation through models trained on search spaces*. The remainder of this paper is structured as follows: Section 2 reviews related work for XAI in optimisation, work on mined search space models for explainability and evolutionary feature selection. Section 3 introduces the PUBOi benchmark, as well as the feature attribution methods. Section 4 outlines the Methodology and Section 5 presents and analyses the findings. Limitations and Conclusions can be found in Sections 6 and 7 respectively.

2 Background

There has been a recent uptick in interest in applying explainable AI (XAI) techniques within the context of evolutionary computation [25,22,23]. SHAP, which is a major feature attribution method described in Section 3.2, has already been adopted in evolutionary computation research to, for example: (*i*) rank problem landscape descriptors in algorithm performance models [22,23,8,18], (*ii*) quantify the contribution of hyperparameters in post-optimisation analyses [19], and (*iii*) to understand different high-quality breast cancer prediction feature sets by mining local optima via evolutionary feature selection [2].

Recent interest has increased surrounding the idea of mining search space models to improve explainability in evolutionary optimisation. For example, [24] integrate models – specifically linear regression and random forest – into a genetic algorithm framework to provide post-optimisation explanations. Although their approach was evaluated on problems with known variable importance, these were relatively simple examples like the checkerboard problem. Sensitivity analysis was conducted to produce a ranking of variable importance. They found that the estimated importances seemed to match their understanding of the problem importances.

Building on this idea, a later work [17] examined how mined search space models could uncover salient features of solutions. The authors showed that these models can identify which variables most strongly affect solution fitness across different landscape types. Their experiments involved binary optimisation problems – checkerboard, MAXSAT, and trap. Although variable importance was visually estimated using sensitivity analysis in this work, there was not a quantitative analysis of the extent to which known importances and interactions were captured.

3 Methodology

3.1 PUBOi Benchmark

Following the work in [7], we use the recently introduced combinatorial optimisation benchmark, PUBOi (**P**olynomial **U**nconstrained **B**inary **O**ptimisation

with tunable variable importance) [20]. PUBO$_i$ allows researchers to generate problem instances in which the importance of each decision variable is precisely adjustable. Methodologically, PUBOi builds on the Polynomial Unconstrained Binary Optimisation (PUBO) framework. Quadratic Unconstrained Binary Optimisation (QUBO) is a special case of PUBO that restricts interactions to pairs of variables; PUBO — and therefore PUBOi — admits higher-order interactions among binary variables. The general PUBO objective function can be expressed as:

$$f(x) = \sum_i a_i x_i + \sum_{i<j} b_{ij} x_i x_j + \sum_{i<j<k} c_{ijk} x_i x_j x_k + \dots \tag{1}$$

where $x \in \{0,1\}^n$ represents a binary solution vector, and $a_i, b_{ij}, c_{ijk}, \dots$ are the interaction coefficients. Instance construction in PUBOi relies on the Chook generator [13], which first produces instances of the Tile Planting (TP) problem and then reformulates them as PUBOi tasks. TP asks for a placement of tiles on a grid that minimises overlaps and gaps. PUBOi extends TP by superimposing order-2 Walsh functions, thereby providing tunable variable importances and giving each instance a known ground-truth importance profile [20]. Walsh functions form an orthogonal basis for pseudo-boolean functions [20]. A Walsh function of order k is $\varphi_k(x) = (-1)^{\sum_{i=0}^{n-1} k_i x_i}$, where $x = (x_0, x_1, \dots, x_{n-1})$ is a binary vector and $k = (k_0, k_1, \dots, k_{n-1})$ specifies the interaction pattern. The overall objective function in PUBO$_i$ is an aggregate of m sub-functions, each drawn from a portfolio of Walsh-based components whose selection probabilities encode their *apriori* importance $f(x) = \sum_{i=1}^{m} f_i(x)$.

3.2 XAI Techniques

SHAP: SHapley Additive exPlanations SHAP is a model-agnostic unified approach inspired by cooperative game-theory for machine learning model predictions [10]. Each feature is attributed an importance value for a specific prediction based on Shapely values that are collected by training models using distinctive sets of features. The marginal contribution of that feature for an observation can be calculated by subtracting the prediction of a model that excludes that feature from a model prediction that includes that feature [21].

LIME: Local Interpretable Model-agnostic Explanations LIME is another widely used XAI method that provides local, instance-specific explanations by approximating the model locally with an interpretable surrogate [15]. LIME focuses on perturbing the input instance and fitting a simple model (e.g., linear regression) to these perturbations to understand the influence of each feature on the prediction. For a given instance x, LIME generates a neighbourhood of perturbed samples around x, obtains predictions for these samples using the black-box model, and then fits an interpretable model to this local data. The coefficients of the interpretable model serve as the feature importance scores for the original instance. Although LIME is local, we make this global by aggregating these local explanations across a set of instances.

Permutation Feature Importance Permutation feature importance (PFI) [12] evaluates the contribution of a feature by measuring the increase in prediction error that occurs when its values are randomly permuted, thus breaking the feature-response relationship. A variable is deemed *important* if this permutation increases the error ; if the error is unaffected or low, the variable is considered *unimportant* [12]. Originally introduced for random forests in [4], PFI was later generalised to a model-agnostic form in [5], which is the variant used in the present study.

3.3 Iterated Local Search (ILS)

In this work, we use Iterated local search (ILS) [9] to sample the solution space of low-dimensional and high-dimensional PUBOi instances. In previous work[7], this was used to sample PUBOi instances only in low-dimensional spaces and proved successful. Therefore, we wish to scale the same approach to larger-dimensional spaces. ILS is a single-point metaheuristic consisting of local searches chained together by large random perturbations. The algorithm we use starts from a randomly generated solution and a first-improvement hill-climber repeatedly flips bits in a binary-encoded solution using a fixed *mutation strength* m. This defined as a constant fraction of the decision variables, and a choice is made after every mutation operation regarding acceptance. The same faction is maintained when the problem dimension changes, so the relative size of each move remains constant across experimental setups. Upon reaching a local optimum, the algorithm then applies a single perturbation that flips solution bits exactly three times the mutation strength – i.e., a neighbourhood radius of $p = 3m$. This is designed to move the search trajectory beyond the current basin of attraction while still preserving useful structural information. The perturbed solution is then subject to the same hill-climber, with the resulting local optimum replacing the incumbent only if it is at least as good. The cycle of *hill-climb*, *kick*, *hill-climb* repeats until the evaluation budget is exhausted.

4 Experimental setup

4.1 Instance Generation

Instances are created using the PUBOi problem generator[1]. A set of parameters used in these experiments can be seen in Table 1 — these are the ones which vary between our problem sizes; there are other PUBOi parameters for which we use those suggested in the generator [20]. The problem dimension n refers to the size of the binary solution; sub-functions m relate to the objective function – the variables of importance are present in the sub-functions as a way of constructing variable importance. The degree of importance refers to the magnitude of difference in importance between important and unimportant variables: each class of importance (important and unimportant) appears with a probability

[1] https://gitlab.com/verel/pubo-importance-benchmark

proportionally to its degree of importance $d_i \in \mathbb{R}^+$[20]. The degree d_0 represents important variables, and d_1 for unimportant variables. Previous work [7] set the *degree* of important variables $d_0 = 2$, the number of sub-functions $m = 5$, and problem dimension $n = 14$. This setup allowed for full enumeration of the space.

Table 1: PUBOi parameters. Settings are shown as SMALL and LARGE for the small (n=14) and large (n=1000) instances, respectively.

Param	**Description**	**Setting**	
		SMALL	LARGE
n	Problem dimension	14	1000
m	Number of sub-functions	5	$[0.01, 0.2] \times \frac{n(n-1)}{2}$
d_i	Degree of importance per class	$d_0 = 2, d_1 = 1$	$d_0 \in [1, 10],\ d_1 = 1$

In this work, we first investigate $d_0 = 2$ and $m = 5$ in SMALL $n = 14$. Then, for the much larger problem dimension LARGE $n = 1000$, we set $m = [0.01, 0.2] \times \frac{n(n-1)}{2}$ and $d_0 \in [1, 10]$ — these are the parameters originally used for the benchmark, where problem sizes of 1000 were also considered [20]. In initial analysis we observed these parameters did not seem to ensure a clear distinction between important and unimportant variables for SMALL, which is why that problem dimension has different parameters. For SMALL and LARGE, we generate a total of 60 instances: 30 of size 14, and 30 of size 1000.

4.2 Training Data Generation

We sample solutions and their fitnesses as training data for the models by conducting iterated local searches (introduced in Section 3.3) and logging every solution seen. The algorithm is executed 10 times for a budget of 200 fitness evaluations on 30 instances each for SMALL and LARGE. During these runs, for each instance, we record all solutions encountered in a sample. The algorithm is coded from scatch in Python.

In the previous work [7] the ILS mutation and perturbation strengths (i.e. the number of bits flipped) were 1 and 3, respectively. We use that here for SMALL. For LARGE, we apply a scaled equivalent of them: mutation strength $M(n) = \frac{n}{14}$ and then perturbation strength $P = 3M$. LARGE has n=1000, which results in M=71 and P=213.

The fitness of the solutions is evaluated according to the PUBOi fitness function defined in Section 3.1. In SMALL and LARGE, we wish to investigate whether variables are captured across instances. Previous work for SMALL [20] aggregated median importances across folds, runs, and instances for each variable — however, this has the potential to obscure higher or lower importances. We instead investigate the entire distribution of important and unimportant variables across each instance.

4.3 Model Training

For both SMALL and LARGE, we train a random forest regressor[2] on binary solutions as input and their fitness values as response. During model training, 3-fold cross-validation is employed. In prior work [7], it was clear random forest and multi-layer perceptron had similar performance (0.99 R^2), so we choose a single model for the purposes of this study. We use random forest mostly according to the default configuration but choose to reduce *n_estimators* to 50.

4.4 Explainers

For SHAP [10] we use the Python package [11] and its PermutationExplainer, which reduces computational overhead by sampling only a subset of possible feature combinations rather than evaluating them exhaustively. This approach produces feature attributions by observing how the model's predictions change when the feature sets are perturbed. For PFI, we use the model-agnostic Python package [1] where we set $n\_repeats = 10$. The default for this parameter is 5 repeats, and for more accurate importances, this parameter can be increased but this is at the cost of computation.

LIME [15] is implemented through the Python package for TabularExplainer [14], which focuses on explaining a single prediction, but can also be used as a global explainer, which we use in this study: explanations are obtained for several predictions together (i.e. the validation split of the data).

4.5 Statistical testing

For both SMALL and LARGE we perform tests to compare the distributions of the important and unimportant variables for each explainer. These tests for explainer distributions will assess how well the known importances are captured. A distribution consists of the explainer values across all folds and runs for each instance. The null hypothesis here is that there is no significant difference between important and unimportant variables in SMALL and LARGE. As we conduct multiple tests, we use a Bonferroni correction for the p-values. Rather than use a dichotomous approach of *significant* or *not significant*, we instead use the following significance levels to denote magnitude differences in p-values: not significant (ns) means $p > 0.05$; * is $0.01 < p \leq 0.05$; ** $0.001 < p \leq 0.0001$; *** $0.0001 < p \leq 0.001$; **** $p \leq 0.0001$.

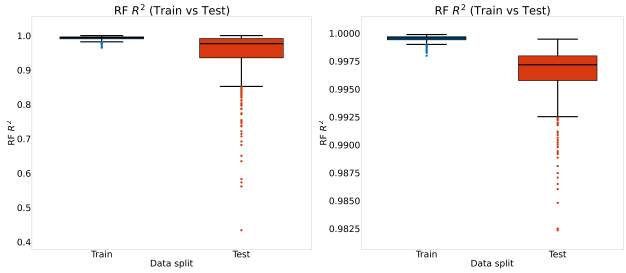
5 Results

5.1 Scalability of model performance

Before utilising XAI methods, it is sensible to first assess the scalability and quality of model performance; otherwise, we may be explaining noise. The distribution of R^2 scores can be seen in Figure 1, where SMALL and LARGE can be

[2] https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html

seen in 1a and 1b respectively. For both SMALL and LARGE, the distribution refers to all gathered R^2 scores taken across 300 models [30 instances with 10 ILS runs subject to 200 evaluations for sampling each]. From the plot, we can see that the R^2 scores for the training and test (i.e. validation in this case) set are plotted. The y-axis represents the R^2 value for the model, and the x-axis separates the train and test sets. It is clear that high-quality models for both SMALL and LARGEare obtained, with ≈ 0.99 median R^2 for the test set, with larger interquartile ranges for LARGE presumably due to the dimensionality.



(a) Model performance for RF in SMALL.

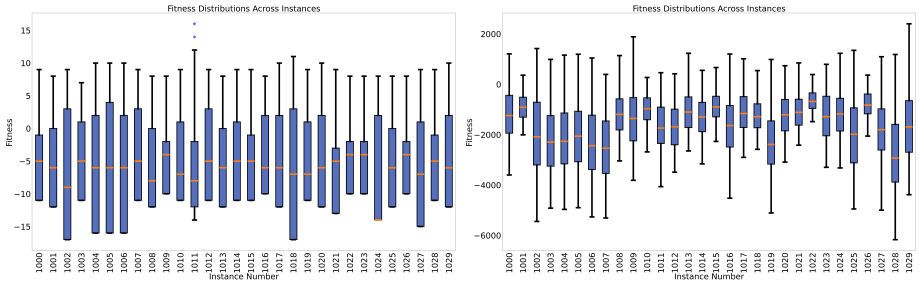
(b) Model performance for RF in LARGE.

Fig. 1: Model performance, where ILS is capped at a 200 evaluation budget across 10 runs for an instance. This is over 300 sets of model training data.

5.2 Scalability of importance estimation quality

Fitness distributions We begin by considering the fitness ranges sampled from the PUBOi instances. This information will be helpful in understanding the obtained SHAP and LIME values in the next section — their ranges will be on the same scale as the model response variable, which in our case is fitness. From Figure 2, we see the range in fitness values for solutions across 30 instances with 300 sets of model training data for SMALL and LARGE in Figure 2a and 2b respectively. Individual istances are located along the x-axis, while the y-axis represents the fitness values for an instance with 10 sets of model training data (sampled using ILS). In SMALL, we can see that the fitness values range between ≈ -15 and ≈ 15, and for LARGE this is between ≈ -6000 and ≈ 2000. These should be the bounds for our SHAP and LIME values. In the case of PFI, however, the explainer values are in the same range as the model quality metric — in our case, R^2.

Important versus unimportant variable estimations We now assess how well important and unimportant variables are estimated by the explainers in 30 in-



(a) Fitness distributions across 30 for SMALL.

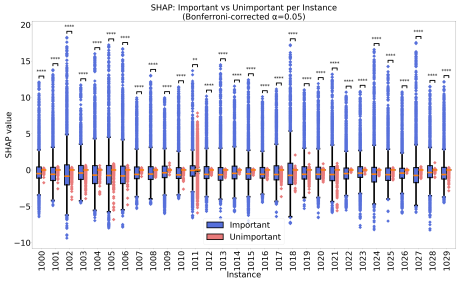
(b) Fitness distributions across 30 instances for LARGE.

Fig. 2: Here we see the fitness distributions across 30 instances with 10 runs of ILS, covering 300 sets of model training data each for SMALL and LARGE.

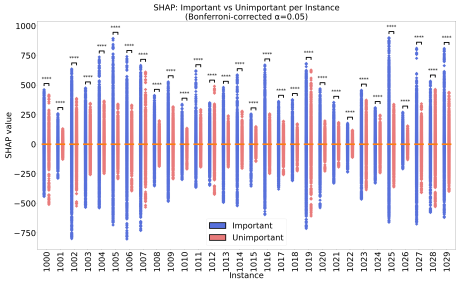
stances each for SMALL and LARGE. Statistical tests are performed to compare the explainer distributions for important and unimportant variables, and we plotted separately for each explainer. The difference is according to the *'s, which denote a difference in magnitude based on ranges we set. Larger magnitude differences indicate that there is a statistical difference between important and unimportant variables; this is desirable as it indicates that known important variables are captured. The plots can be seen in Figure 3 for SHAP (3a and 3b), LIME (3c and 3d), and PFI (3e and 3f). SMALL is located on the left, and LARGE on the right. The x-axis represents each individual instance [10 sets of explainer values], and the y-axis shows the estimated importance.

It is immediately clear that in SMALL known important and unimportant variables are captured well by the explainer values; this ratifies the findings in the previous work [7]. For SHAP and PFI, all distributions show significant differences across the 30 instances, showing that there is a clear distinction between important and unimportant variables, which can be seen in Figure 3a and 3e respectively. Conversely, for LIME, 5 out of 30 instances show that there is no statistical difference between important and unimportant variables, which may cause issues when scaling this to LARGE. Overall, it is clear that important variables are captured for all three explainers in this dimension.

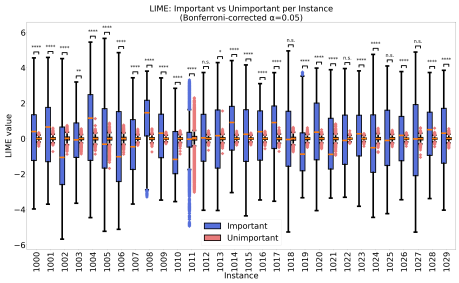
In LARGE, we can see that according to the statistical tests, the distributions of important and unimportant variables are all statistically different in Figure 3b for SHAP and 3f for PFI. This indicates that important variables have higher estimated importances than unimportant variables. That being said, for LIME 16 of the 30 instances do not show significant differences in the distributions. This indicates that the method does not scale well — although in some cases, important variables were captured. In the case of PFI, all instances show significant differences in the distribution of important and unimportant variables.



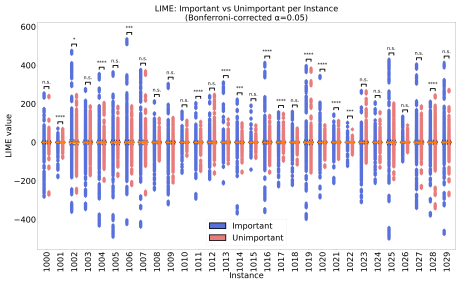
(a) SHAP Explainer data for SMALL– distributions of important and unimportant variables



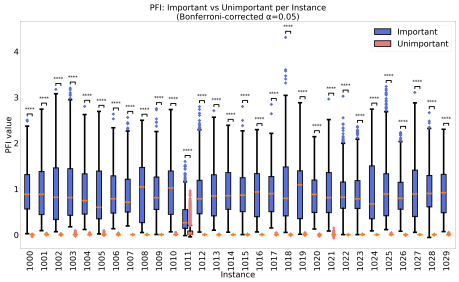
(b) SHAP Explainer data for LARGE– distributions of important and unimportant variables



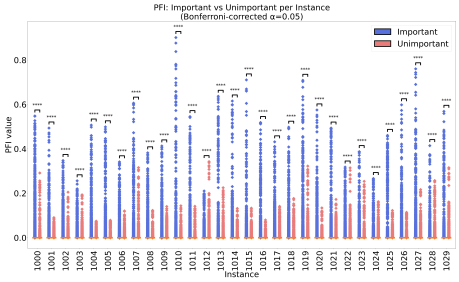
(c) LIME Explainer data for SMALL– distributions of important and unimportant variables



(d) LIME Explainer data for LARGE– distributions of important and unimportant variables



(e) PFI Explainer data for SMALL– distributions of important and unimportant variables



(f) PFI Explainer data for LARGE– distributions of important and unimportant variables

Fig. 3: For the three XAI attribution methods, we see the distributions of important and unimportant variables. This is across 300 models: 30 instances and 10 ILS runs to sample for each instance

6 Limitations

Although PUBOi problems are constructed to facilitate study of variable importance, they also contain variable *interactions*. The polynomial terms which

are passed to the Chook functions include two variables. It therefore follows that, if two variables co-occur in a term, then they interact (they affect fitness together). Feature attribution methods were used in this study, but these may not properly consider feature interactivity [12]. This likely affected the results in LARGE due to the presence of more features.

Empirically, we found a good budget in SMALL and this was also used in LARGE, however this is a limitation as it is not relative to the size of the search space. Future work will consider dimension-dependent budgeting.

7 Conclusion

In this work, we explored the potential of using explainable AI (XAI) methods to estimate variable importance in high-dimensional optimisation problems using a recently proposed benchmark, PUBO_i. This is used as a test-bed for experiments as it facilitates tunable variable importance, providing a ground-truth environment. We trained random forests on the sampled search spaces of PUBO_i instances for low and high-dimensional data. XAI feature attribution methods were applied to the models. We performed statistical tests to compare the distribution of important and unimportant variables and found that in high-dimensional instances, the important variables were captured but to a lesser extent than in low-dimensional instances. This shows potential that the method has potential in high dimensions. Data and code for this work can be found in a public Zenodo repository[3].

References

1. Permutation feature importance, https://scikit-learn.org/stable/modules/permutation_importance.html
2. Adair, J., Thomson, S.L., Brownlee, A.E.: Explaining evolutionary feature selection via local optima networks. In: Proceedings of the Genetic and Evolutionary Computation Conference Companion. pp. 1573–1581 (2024)
3. Arrieta, A.B., Díaz-Rodríguez, N., Ser, J.D., Bennetot, A., Tabik, S., Barbado, A., García, S., Gil-López, S., Molina, D., Benjamins, R., Chatila, R., Herrera, F.: Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai (2019)
4. Breiman, L.: Random forests. Machine Learning 45(1), 5–32 (2001)
5. Fisher, A., Rudin, C., Dominici, F.: All models are wrong, but many are useful: Learning a variable's importance by studying an entire class of prediction models simultaneously (2018), https://arxiv.org/abs/1801.01489
6. Fyvie, M., McCall, J.A., Christie, L.A., Brownlee, A.E., Singh, M.: Towards explainable metaheuristics: Feature extraction from trajectory mining. Expert Systems p. e13494 (2023)
7. Hunter, K., Thomson, S.L., Hart, E.: Into the black box: Mining variable importance with xai. In: García-Sánchez, P., Hart, E., Thomson, S.L. (eds.) Applications of Evolutionary Computation. pp. 20–35. Springer Nature Switzerland, Cham (2025)

[3] https://zenodo.org/records/15671240

8. Kostovska, A., Vermetten, D., Džeroski, S., Doerr, C., Korosec, P., Eftimov, T.: The importance of landscape features for performance prediction of modular cma-es variants. In: Proceedings of the Genetic and Evolutionary Computation Conference. pp. 648–656 (2022)
9. Lourenço, M.A., Torgo, L., Pereira, L.: Iterated local search. In: European Conference on Combinatorial Optimization. pp. 38–51. Springer (1997)
10. Lundberg, S.M., Lee, S.: A unified approach to interpreting model predictions. CoRR abs/1705.07874 (2017), http://arxiv.org/abs/1705.07874
11. Lundenberg, M.S., Lee, S.: Shap docuementation. https://shap.readthedocs.io/en/latest/example_notebooks/api_examples/explainers/Permutation.html (2017), [Accessed 21-08-2024]
12. Molnar, C.: Interpretable Machine Learning. https://christophm.github.io/interpretable-ml-book/
13. Perera, D., Akpabio, I., Hamze, F., Mandra, S., Rose, N., Aramon, M., Katzgraber, H.G.: Chook – a comprehensive suite for generating binary optimization problems with planted solutions (2021), https://arxiv.org/abs/2005.14344
14. Ribeiro, M.T., Singh, S., Guestrin, C.: "why should i trust you?": Explaining the predictions of any classifier. https://lime-ml.readthedocs.io/en/latest/ (2016), [Accessed 16-11-24]
15. Ribeiro, M.T., Singh, S., Guestrin, C.: "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. p. 1135–1144. KDD '16, Association for Computing Machinery, New York, NY, USA (2016), https://doi.org/10.1145/2939672.2939778
16. Saranya, A., Subhashini, R.: A systematic review of explainable artificial intelligence models and applications: Recent developments and future trends. Decision Analytics Journal 7, 100230 (2023), https://www.sciencedirect.com/science/article/pii/S277266222300070X
17. Singh, M., Brownlee, A.E., Cairns, D.: Towards explainable metaheuristic: mining surrogate fitness models for importance of variables. In: Proceedings of the Genetic and Evolutionary Computation Conference Companion. pp. 1785–1793 (2022)
18. Ŝkvorc, U., Eftimov, T., Koro, P., et al.: Analyzing the generalizability of automated algorithm selection: A case study for numerical optimization. In: 2023 IEEE Symposium Series on Computational Intelligence (SSCI). pp. 335–340. IEEE (2023)
19. van Stein, N., Vermetten, D., Kononova, A.V., Bäck, T.: Explainable benchmarking for iterative optimization heuristics. arXiv preprint arXiv:2401.17842 (2024)
20. Tari, S., Verel, S., Omidvar, M.: Puboi: A tunable benchmark with variable importance. In: Evolutionary Computation in Combinatorial Optimization: 22nd European Conference, EvoCOP 2022, Held as Part of EvoStar 2022, Madrid, Spain, April 20–22, 2022, Proceedings. p. 175–190. Springer-Verlag, Berlin, Heidelberg (2022), https://doi.org/10.1007/978-3-031-04148-8_12
21. Thomson, S.L., Adair, J., Brownlee, A.E.I., van den Berg, D.: From Fitness Landscapes to Explainable AI and Back. In: Proceedings of the Companion Conference on Genetic and Evolutionary Computation. pp. 1663–1667. GECCO '23 Companion, Association for Computing Machinery, New York, NY, USA (Jul 2023), https://dl.acm.org/doi/10.1145/3583133.3596395
22. Trajanov, R., Dimeski, S., Popovski, M., Korošec, P., Eftimov, T.: Explainable landscape-aware optimization performance prediction. In: 2021 IEEE Symposium Series on Computational Intelligence (SSCI). pp. 01–08. IEEE (2021)

23. Trajanov, R., Dimeski, S., Popovski, M., Korošec, P., Eftimov, T.: Explainable landscape analysis in automated algorithm performance prediction. arXiv (2022), https://arxiv.org/abs/2203.11828
24. Wallace, A., Brownlee, A.E., Cairns, D.: Towards explaining metaheuristic solution quality by data mining surrogate fitness models for importance of variables. In: Artificial Intelligence XXXVIII: 41st SGAI International Conference on Artificial Intelligence, AI 2021, Cambridge, UK, December 14–16, 2021, Proceedings 41. pp. 58–72. Springer (2021)
25. Zhou, R., Bacardit, J., Brownlee, A., Cagnoni, S., Fyvie, M., Iacca, G., McCall, J., van Stein, N., Walker, D., Hu, T.: Evolutionary computation and explainable ai: A roadmap to transparent intelligent systems. arXiv preprint arXiv:2406.07811 (2024)