Hierarchical Knowledge Extraction from Opaque Machine Learning Predictors

Federico Sabbatini[1] and Roberta Calegari[2]

[1] University of Urbino, Urbino, Italy f.sabbatini1@campus.uniurb.it
[2] University of Bologna, Bologna, Italy roberta.calegari@unibo.it

Abstract. Adopting opaque machine learning predictors, which achieve very high predictive performance, often necessitates incorporating symbolic knowledge-extraction techniques. These techniques aim to explain the opaque predictions, thus making them applicable in high-stakes scenarios. The development of symbolic knowledge-extraction procedures is evolving alongside the dynamic machine learning landscape. However, there are recurring drawbacks that tend to be overlooked or addressed in a suboptimum way. Common examples include the non-exhaustiveness of the global explanations generated for a black-box predictor or the unwanted discretisation introduced in the prediction of continuous variables. To tackle these challenges, in this work, we introduce the HEx algorithm, its formalisation and its properties. This algorithm aims to obtain a symbolic, hierarchical representation of the knowledge acquired by opaque machine learning classifiers and regressors, always ensuring knowledge exhaustiveness and avoiding any output discretisation. Experiments demonstrating the superior capabilities of HEx compared to state-of-the-art competitors in terms of predictive performance, completeness, and human readability are presented.

Keywords: Explainable artificial intelligence · Symbolic knowledge extraction · PSyKE

1 Introduction

Inherently interpretable models, where humans can understand the reasoning behind the outputs, may not be sufficiently complex to accurately capture the nuances of the domain, resulting in suboptimum predictive capabilities. Conversely, there are machine learning (ML) predictors with a high degree of complexity that enables them to achieve superior predictive performance. However, these models lack interpretability, especially concerning internal decision-making processes and input feature exploitation. This leads to an impossibility for human users to trace and thus understand the exact workflow leading to a given outcome starting from an input query [8]. Despite the unfortunate proof of an inverse proportionality relationship between human interpretability and predictive performance [10, 30], there is a growing demand for interpretable and explainable predictions. This demand has resulted in the rejection of opaque models

in critical decision systems, despite their capability to provide highly accurate predictions [17, 18]. The criticality of a system is determined by the extent to which important aspects of human lives are affected by the predictive system. For example, critical domains include finance, security, and medicine. In such contexts, transparent models are preferred over opaque ones, even if they may have limitations in predictive performance.

To keep the advantages coming along with complexity and opaqueness, different strategies have been proposed by the explainable artificial intelligence community [5, 22]. One consists of performing symbolic knowledge extraction (SKE; [26]) to provide human-interpretable surrogates of the opaque ML predictors. These surrogate models are interpretable for humans and preserve as much as possible the predictive capabilities of the original, opaque predictor.

SKE techniques recently proposed in the literature are mostly designed to be applied upon specific sorts of ML models, typically artificial neural networks [11, 15, 24, 43]. The adoption of model-agnostic extractors seems to have lost momentum, despite the increasing diversity of available ML models. When SKE techniques are tailored specifically to narrow clusters of opaque predictors, it can result in a fossilisation of the decision-making system. For instance, a system leveraging a fuzzy neural network may be empowered with the SKE algorithm proposed in [43] to achieve human interpretability. As a result, minimum changes in the model architecture (e.g., another kind of neural network or switching to a random forest) would result in the compelling modification of the SKE technique, which is only suitable for fuzzy neural networks. Conversely, model-agnostic extractors are not strictly bound to specific subsets of ML predictors. Therefore, they enable the avoidance of these unwanted dependencies.

Accordingly, we introduce here the HEx algorithm, a novel task- and model-agnostic SKE procedure to explain opaque ML predictors accepting continuous input features. HEx provides a symbolic human-interpretable model whose predictions are drawn according to a tree-based hypercubic partitioning of the input feature space [38, 40]. Our experiments show that opaque models of any kind can be explained via HEx with higher predictive performance, completeness and human-readability extent than state-of-the-art knowledge-extraction techniques.

The manuscript is organised as follows. Related works are described in Section 2. The design and implementation of the novel HEx algorithm for hierarchical symbolic knowledge extraction are reported in Section 3. Experiments assessing the effectiveness and usability of HEx compared to state-of-the-art competitors are detailed in Section 4. Conclusions are finally drawn in Section 5.

2 Related Works

When facing opaqueness deriving from sophisticated algebraic calculations constituting the core of sub-symbolic predictors, SKE procedures provide human interpretability through a symbolic approximation of the original, opaque predictor behaviour [28, 42]. The literature offers a wide range of different alternatives, even though these are mostly constrained in their actual applicability. The most

appropriate SKE technique to adopt in a specific scenario should be identified in the light of the candidates' peculiarities, constraints and overall achieved quality. Beyond mandatory requirements, also the kind of knowledge outputted by SKE methods should be taken into consideration. In particular, shape and expressiveness of the knowledge are relevant aspects to analyse. Knowledge shapes typically adopted are lists and trees of rules. The expressiveness of knowledge items ranges from propositional *if-then* rules to more sophisticated M-of-N, fuzzy and oblique rules.

Requirements of SKE techniques. The three main requirements constraining the applicability scope of SKE techniques are detailed in the following.

The degree of inspection inside the internal structure of the opaque model during the knowledge extraction is defined *translucency*. There are three distinct classes of SKE algorithms based on translucency, namely, decompositional, pedagogical and eclectic [2]. Decompositional SKE procedures analyse the model's internal parameters, for instance, support vectors in a support vector machine or connection weights in neural networks. Pedagogical extractors only consider the input/output response of the opaque predictor to generate symbolic knowledge. Eclectic procedures combine elements of both categories. It is worth emphasising that decompositional techniques are bounded to individual classes of black boxes, e.g., only tree ensembles, or fuzzy neural networks, or any kind of feed-forward neural networks for the most general procedures. Conversely, pedagogical models are model-agnostic and thus present fewer applicability constraints.

SKE techniques may be applicable only in some domains, depending if they are compatible only with opaque classifiers, only with regressors, or with both categories. The majority of extractors are limited to classification tasks [13, 16, 46, 48], with some exceptions for regression tasks [23, 43, 45]. A very narrow subset of extractors results to be task-agnostic [7, 27, 35, 37, 44].

Depending on the specific peculiarities of SKE algorithms, these may be compatible with different subsets of input features. More in detail, feature kinds typically accepted are binary and/or discrete and/or real-valued. It is worth highlighting that it is possible to adopt conversion routines to map features into the desired domain (e.g., one-hot encoding, discretisation and binarisation methods). These processing phases should be performed before training the opaque predictor and performing SKE.

Evolution of Translucency over Time. A review of recent literature about SKE shows an increasing propensity to design novel decompositional techniques. For instance, during the last two years the following knowledge extractors were proposed: 2 pedagogical algorithms for regression tasks [25, 31] and 1 for classification tasks [12]; 1 decompositional extractor for decision tree ensembles [29], 1 for support vector machines [3] and 6 for neural networks. More in detail, these latter are explicitly designed for convolutional neural networks [24], fuzzy neural networks [11, 43] or networks with exactly four layers [15]. The remaining 2 are more general and assume no constraints on the network structure [4, 20].

All these decompositional extractors are only applicable to classification tasks except the one proposed in [43].

An analogous trend may be observed in 2021 (10 decompositional and 3 pedagogical algorithms). As a result, it can be noticed that recent SKE procedures are mostly decompositional and focused on opaque classifiers. Given the undeniable general applicability of pedagogical techniques, we argue that the research community could benefit from the proposal of novel algorithms overcoming the hindrances of existing ones.

State-of-the-Art Pedagogical SKE Techniques. We conclude this section by briefly reviewing state-of-the-art pedagogical extraction algorithms inspiring HEx and adopted as benchmarks in our experiments.

The Iter SKE algorithm can be adopted to explain opaque regressors accepting continuous input features [23]. It is based on a bottom-up iterative strategy leading to the construction of a set of hypercubes within the input feature space. Cubes are disjoint and possibly non-exhaustive. Predictions are made based on the output values associated with the cubes enclosing the queries. Cubes' output values are calculated during the knowledge extraction and they are constant values, therefore a discretisation of the predicted outputs is introduced.

Other SKE techniques applicable to explain opaque regressors via disjoint hypercubes are GridEx and GridREx [31, 41] also accepting only continuous input features. They induce a recursive, top-down partitioning of the input feature space. As in the case of Iter, explainability is obtained through a human-interpretable hypercubic partitioning. However, GridEx and GridREx perform an input feature analysis to prune the least relevant features from the output knowledge. Only GridREx avoids prediction discretisation by adopting linear combinations of the input variables instead of constant values as cube outputs.

The CART algorithm may be exploited to induce a decision tree explaining the outcomes of opaque classifiers or regressors, without constraints on the input feature kind. The output of CART is a binary tree where leaves are associated with constant predictions and internal nodes are constraints on the input variables [7]. Complete paths from the tree root to each leaf correspond to the explanations for the opaque outcomes.

3 Hierarchical Knowledge Extraction

HEx (Hierarchical EXtractor) is a pedagogical SKE technique applicable to opaque predictors operating on continuous attributes. Its algorithm merges the advantages of existing SKE techniques to achieve better predictive performance and conciseness than its competitors. More in detail, it exploits the hierarchical nature of tree-based systems and the human interpretability of hypercubes. Similarly to CART, it can explain opaque classifiers as well as regressors and the hypercubes it identifies can be associated with constant values, as in Iter, or with linear combinations of the input variables, as in GridREx.

Given the domain of input features accepted by HEx, the hypercubes it identifies are described in terms of interval-inclusion constraints over continuous input attributes. Hypercubes are not disjoint, since smaller cubes may be enclosed within a bigger one. Nonetheless, predictions based on HEx hypercubes are not ambiguous, because cubes are ordered. Inner cubes have the highest priority than outer ones when predicting a query. Therefore, if a query is enclosed within more than a cube, only the innermost is considered.

The hierarchical, hypercubic partitioning provided by HEx is always exhaustive since the last identified cube is used as *default* cube. All queries not covered by other cubes are thus enclosed by it.

HEx requires a predictor and a training set to be provided by users. The training set may be the same as the one previously used to train the opaque model. During the knowledge extraction phase, HEx supports classification by finding the most frequently predicted class label inside each hypercube and by using it as an interpretable prediction for all queries falling inside that cube. Similarly, in regression tasks where constant outputs are used, the average of all opaque predictions within each cube is used. Otherwise, when expressing outputs as linear combinations of the input variables, a linear fit within each cube is performed to correlate the training inputs with the opaque predictions provided by the ML model.

The human-interpretable knowledge provided by HEx is generated by converting each hypercube identified during the extraction phase into an *if-then* rule having as precondition a formal description of the hypercube boundaries and as postcondition the associated prediction. The conversion is performed methodically from the innermost cubes to the default one.

Algorithmically, HEx partitions the input feature space according to a top-down strategy, preserving information about parent cubes and corresponding child cubes through a tree structure. It differs from existing hypercube-based extractors since its decisions are grounded on the notion of *gain* between cubes. More in detail, when splitting a parent cube, the gain from replacing the ancestor cube with its child sub-cubes is computed, and only descendants showing a favourable gain are retained. The gain is always calculated between a cube and its children/descendants.

To give an abstract evaluation of the quality of symbolic knowledge generated by HEx, we refer to the 3 most used proxies for knowledge quality: predictive performance, readability and completeness [14, 19, 32, 33, 47]. HEx completeness is ensured by design. Its predictive performance may be measured through the same scoring metric adopted for the underlying ML model, by comparing the HEx predictions to the data set ground truth or the opaque predictions. Finally, the human-readability extent is typically assessed based on the symbolic knowledge size, e.g., the number of items it contains. In the case of HEx, readability is bounded to the quantity of identified hypercubes, by considering that readability decreases if the number of cubes increases. Thanks to the knowledge-extraction strategy it employs, HEx can generate more concise knowledge pieces (i.e., with fewer items) than other state-of-the-art competitors.

The overall predictive performance and readability exhibited by the knowledge extracted with HEx depends on the user-defined hyper-parameters provided as inputs, clearly impacting the steps executed during the algorithm. These parameters and the HEx algorithm are detailed in the following subsections, respectively. It is important to notice that the user-defined parameters of HEx can be automatically tuned via the existing PEDRO procedure without any modification [31].

HEx Hyper-Parameters. The HEx knowledge extractor requires 4 input parameters to be tuned by users: the maximum depth of its recursions (δ), the minimum quantity of samples to consider within each hypercube (m), the error threshold deemed acceptable during the extraction (θ), and the strategy to adopt at each recursion to partition the input feature space ($S_1, \ldots, S_\delta$).

δ is the upper-bound for the number of recursive splits to perform. It is defined as an integer number greater or equal to 1, formally: $\delta \in \mathbb{N} \setminus \{0\}$. For $\delta = 1$ only a single partitioning step is performed. This parameter is an upper-bound since the algorithm may pre-emptively terminate if all hypercubes found during recursion $i, i < \delta$ have a predictive error below θ. δ is strictly bounded with the overall human-interpretability extent of the knowledge outputted by HEx, since growing δ values generally lead to larger quantities of identified hypercubes and therefore to larger knowledge pieces.

m is the lower-bound for the number of training instances to consider when creating a hypercubic partition. It is defined as an integer number greater or equal to 1, formally: $m \in \mathbb{N} \setminus \{0\}$. After the cube creation, if it contains less than m training instances, the training data set is augmented by generating random samples inside the cube, up to the user-defined value. Large values of m are employed to obtain more robust results in presence of outliers.

θ is the user-defined error threshold adopted to discern between hypercubic regions that need to be refined during successive recursive iterations of HEx and regions to be considered final, i.e., those having predictive error below the threshold. The definition of θ depends on the task at hand. When executing classification, θ is a threshold on the cubes' classification accuracy. It is thus defined as a real number in the [0, 1] interval. On the other hand, when performing regression, it represents a threshold on the mean absolute error of hypercubes. Therefore, $\theta \in \mathbb{R}^+$.

θ represents a sensitivity parameter controlling the overall predictive performance of the knowledge provided by HEx. Small values of θ indicate a priority for maximising predictive performance, potentially sacrificing human readability. This occurs because HEx recursions may not be terminated prematurely before reaching δ. This parameter is also used to check if adjacent hypercubes may be merged and if there is a gain in splitting a parent cube into child sub-cubes, as detailed in Section 3.

$S_1, \ldots, S_\delta$ are the splitting strategies to adopt during recursive iterations $1, \ldots, \delta$. Two different strategies are supported by HEx, namely, fixed or adaptive. Fixed strategies are parametrised with an integer value, k, identifying the

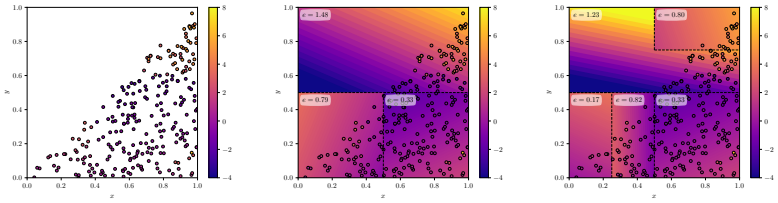


Fig. 1: Example of HEx instances applied to an artificial regression data set.

number of partitions that are created along *every* input variable. In this case, a parent cube is split into k^d child sub-cubes, by assuming d input features. Nonetheless, the actual number of produced cubes may be decreased after the HEx merging phase. Differently, adaptive strategies are parametrised with a monotone increasing function f defined as $f : [0, 1] \to \mathbb{N} \setminus \{0\}$ and generating $f(r)$ partitions along input dimensions having relevance equal to r. This allows for conducting a sensitivity analysis on the impact of individual input features on predictions and subsequently performing a finer slicing on the most important ones. For the least important features, it is possible to select a coarser partitioning as well as to avoid any slicing. In this case, input dimensions that are never sliced are considered irrelevant to the predictions, and therefore, they are not included in the generated symbolic knowledge. In other words, these features are not useful to explain the predictions of the underlying opaque model.

We highlight here that the feature relevance may be assessed through any procedure external to HEx, for instance via the `f_classif` and `f_regression` functions of the `sklearn.feature_selection` Python library. Other adequate methods may be found in the literature [1, 49, 50]. The only requirement of HEx is that the feature importance ranking should be normalised in the [0, 1] interval. The most relevant feature is thus associated with a value of 1.

HEx Algorithm. The HEx algorithm applied to explain a predictor P trained on a data set D is summarised in Algorithm 1. We point out here that G represents a grid object, concisely enclosing both concepts of maximum depth and splitting strategies. In other words, G encapsulates the δ and $S_1, \ldots, S_\delta$ user-defined parameters.

The HEx algorithm starts by identifying the *surrounding hypercube*, i.e., the minimal cube enclosing all the training instances. The root of the tree induced by HEx is therefore initialised with the surrounding cube, with no parent and no children. Then, the following steps are recursively executed for the tree root and any child node successively created (the *current node*, in the following): (i) the cube associated with the current node is split according to the strategy chosen by the user, as described by the grid G; (ii) the data set D is augmented to have at least m instances inside each created sub-cube. New samples are randomly generated and the predictor P is employed as an oracle to predict the corresponding outputs; (iii) the sub-cubes are merged, if possible. The merging

Algorithm 1 HEx pseudocode

```
1: function HEx(P, D, G, θ, m)
2:     H_0 ← the minimal hypercube including all the samples of D
3:     root ← NewNode(H_0, ∅, P, D, θ)
4:     Split(1, root, P, D, G, θ, m)
5:     Π ← post-ordered depth-first search from root
6:     Π ← {node.cube | node ∈ Π ∧ node.gain}
7:     if D \ ⋃_{H∈Π} H ≠ ∅ then Π ← Π ∪ {H_0}
8:     return Π
9: function NewNode(H, parent, P, D, θ)
10:    node ← new Node()
11:    node.cube ← H    node.parent ← parent    node.children ← ∅
12:    node.gain ← Gain(node, P, D, θ)
13:    return node
14: function Gain(node, P, D, θ)
15:    parent ← node.parent                                          ▷ Outer cube
16:    while !parent.gain do
17:        parent ← parent.parent
18:    if task is classification ∧ output of node.cube ≠ output of parent.cube then return true
19:    else if task is regression then
20:        e_o ← PredictiveError(parent.cube \ node.cube, P, D)      ▷ Outer cube error
21:        e_i ← PredictiveError(node.cube, P, D)                    ▷ Inner cube error
22:        if e_o − e_i > 0.6 θ then return true
23:    else return false
24: function PredictiveError(H, P, D)
25:    return the average predictive error of {P(x) | x ∈ H ∩ D}
26: function Split(i, node, P, D, G, θ, m)
27:    if i > depth of G ∨ PredictiveError(node.cube) ≤ θ then return
28:    Π ← all the partitions of node.cube according to the i-th level of grid G
29:    D ← D ∪ ⋃_{H∈Π} GenerateSamplesIn(node.cube, D, m)
30:    Π ← Merge(Π, P, D, θ)
31:    node.children ← ⋃_{H∈Π} NewNode(H, node, P, D, θ)
32:    for all n ∈ node.children do
33:        Split(i + 1, n, P, D, G, θ, m)                            ▷ Recursion
34: function Merge(Π, P, D, θ)
35:    C ← AdjacentCubes(Π)
36:    while (|C| > 0) do
37:        (H_1^*, H_2^*) ← arg min_{(H_1,H_2)∈C} { PredictiveError(H_1 ∪ H_2, P, D) }
38:        H ← H_1^* ∪ H_2^*
39:        if PredictiveError(H, P, D) ≤ θ then
40:            Π ← Π \ {H_1^*, H_2^*} ∪ {H}
41:            C ← AdjacentCubes(Π)
42:        else return Π
43:    return Π
44: function GenerateSamplesIn(H, D, m)
45:    c ←| H ∩ D |
46:    if c < m then return { (m − c) random points in H }
47:    else return ∅
48: function AdjacentCubes(Π)
49:    return ⋃_{H∈Π} { (H, H′) | (H′ ∈ Π \ {H}) ∧ (H and H′ are adjacent) }
```

phase consists of iteratively trying to unify pairs of adjacent cubes until there are eligible pairs. Two cubes are merged only if the resulting cube has a predictive error below the user-defined θ threshold; (iv) the resulting cubes are encapsulated within tree nodes, having the current node as parent; (v) the gain of new nodes is calculated. This measure is a Boolean value indicating whether preserving a child node improves predictive performance or not. It is calculated relative

to the nearest ancestor with a positive gain. In other words, ancestors with negative gain are disregarded as they will not contribute to the resulting symbolic knowledge. The gain is defined differently based on the performed ML task. For classification, a child node has positive gain if its cube is associated with a class label that is different from that of the closest ancestor (i.e., the child cube identifies a region where a different prediction is dominant). For regression tasks, a child node has positive gain if the predictive error of its cube (e_i) is smaller than that of the closest ancestor (e_o). More in detail, this inequality should hold: $e_o - e_i > 0.6\ \theta$; (vi) the workflow is repeated for each one of these child nodes if their cubes have predictive error above the θ threshold, until the maximum user-defined depth.

After the tree induction, all nodes are linearised according to a post-ordered depth-first search starting from the root. However, the root is not included in the ordered list at this stage. All nodes having negative gain are discarded; the cubes associated with the remaining nodes are translated into ordered symbolic rules composing the output human-interpretable knowledge. As a last step, HEx checks if the knowledge covers the whole input feature space. If not, the surrounding cube is also translated into a rule and queued to the knowledge. The final rule of knowledge is always generalised to a default rule.

Figure 1 exemplifies the execution of HEx applied to a synthetic regression data set with two continuous input features (x and y, see left panel). An instance of HEx with $\theta = 0.5$ and $\delta = 1$ is depicted in the middle panel of Figure 1. A fixed splitting strategy performing 2 splits for each input dimension has been adopted. As a result, 4 hypercubes are identified, one containing no training samples (top-left), one with a predictive error $e < \theta$ ($e = 0.33$; bottom-right) and two with an error above θ. In this case, the cube with no samples is merged with an adjacent cube (top-right) without losses in the predictive performance. The resulting cubes are finally translated into knowledge with as many *if-then* rules.

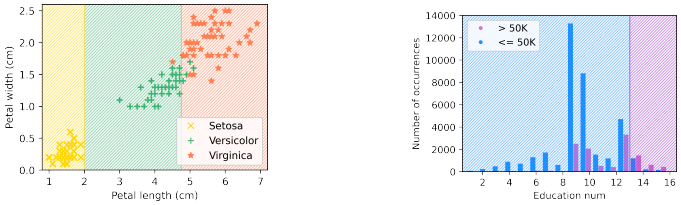
An instance of HEx with $\delta = 2$ and the same other parameters as described above is shown in the right panel of Figure 1. The second recursion of HEx starts after the merging of the two top cubes. In this scenario, the bottom-right cube is no further partitioned, provided its error is below θ. The bottom-left cube is split into four sub-cubes, vertically merged pairwise. One of these merged sub-cubes is marked with positive gain (the leftmost, $e = 0.17$), the other with negative gain. As a result, in the final knowledge there is a rule corresponding to the inner region ($0.00 < x < 0.25, 0.00 < y < 0.50$) followed by a rule derived for the closest parent region having positive gain ($0.00 < x < 0.50, 0.00 < y < 0.50$). Analogously, the top merged cube is split into four sub-cubes and only one is marked with positive gain (the top-right one, with $e = 0.80$). All the others are marked with negative gain and neglected when generating the output knowledge. The overall knowledge of the HEx instance with $\delta = 2$ has five rules corresponding to as many hierarchical hypercubes, with a weighted average predictive performance of 0.61, in contrast to the 3 rules and the average error of 0.73 exhibited by the HEx instance with $\delta = 1$. This constitutes an example of the fidelity/readability

Listing 1 Example of knowledge extracted with HEx for the Iris data set.

```
iris is Setosa if petal length <= 1.93
iris is Virginica if petal length > 4.73
iris is Versicolor otherwise
```

Listing 2 Example of knowledge extracted with HEx for the Adult data set.

```
income is > 50K if education-num is > 13
income is <= 50K otherwise
```



(a) Iris data set. (b) Adult Income data set.

Fig. 2: Decision boundaries identified by HEx.

trade-off typically observed for knowledge-extraction techniques, where an enhancement in predictive performance is often counterbalanced with a worsening in the knowledge human interpretability [10, 30]. Examples of classification rules extracted with HEx are reported in Listings 1 and 2 for the Iris [21] and Adult Income [6] data sets, respectively. The corresponding decision boundaries are shown in Figures 2a and 2b, respectively.

4 Experiments

The effectiveness of HEx has been assessed through experiments involving classification and regression tasks. It has been compared to state-of-the-art SKE techniques available in the PSyKE framework [9, 34, 39] applied to different black boxes. Knowledge quality has been evaluated on human readability (i.e., knowledge size), input space coverage and F_1 or R^2 scores. The Q_s score [36] is also shown as a concise quality assessment encompassing the aforementioned indices (low scores are associated with good quality). Results reported here are averaged over 10 runs. Given the standard deviations close to 0, we only report averages.

Our classification case study is based on the Adult Income data set [6]. It comprises 14 input attributes and 48 842 instances representing information about adult persons, whereas the output feature is a binary label expressing if an individual is likely to receive an income below or above 50 000 dollars. Three different black-box classifiers have been trained on the Adult Income data set, namely,

Table 1: Results for the Adult Income data set.

		CART max depth = 1	CART max depth = 2	CART max leaves = 3	ITER	GRIDEX	HEX
GB	F_1	0.65	**0.68**	0.67	0.66	0.66	0.66
	Rules	**2.00**	4.00	3.00	8.20	3.00	**2.00**
	Coverage	**1.00**	**1.00**	**1.00**	0.86	**1.00**	**1.00**
	Q_s	0.70	1.28	1.00	3.19	1.02	**0.68**
RF	F_1	0.60	**0.67**	**0.67**	0.65	0.66	0.66
	Rules	**2.00**	4.00	3.00	7.90	3.00	**2.00**
	Coverage	**1.00**	**1.00**	**1.00**	0.83	**1.00**	**1.00**
	Q_s	0.80	1.31	1.00	3.24	1.02	**0.68**
DT	F_1	0.60	**0.67**	**0.67**	0.63	0.66	0.66
	Rules	**2.00**	4.00	3.00	8.40	3.00	**2.00**
	Coverage	**1.00**	**1.00**	**1.00**	0.79	**1.00**	**1.00**
	Q_s	0.80	1.31	1.00	3.81	1.02	**0.68**

a gradient boosting (GB) predictor, a random forest (RF) and a decision tree (DT). A train/test splitting with a 1:1 ratio has been performed. The training set has been employed for hyper-parameter tuning via 3-fold cross-validation.

Six knowledge extractors were successively applied to each one of these black boxes, namely, ITER, GRIDEX, HEX and three instances of CART.Knowledge size, coverage and F_1 scores with respect to the data set ground truth are reported and compared in Table 1, with best values highlighted in bold.

It can be noticed that HEX provides knowledge pieces with only two rules (one for the positive class and one for the negative) with an F_1 score of 0.66, slightly lower with respect to the highest score of 0.68 observed for CART. However, this CART instance produces 4 rules, thus denoting a stark worsening in the human-readability extent. The comparison between ITER and HEX is favourable for the latter, achieving the highest F_1 score, readability and completeness simultaneously. Also, the comparison with GRIDEX privileges HEX, since they have the same coverage and F_1 scores, however HEX outputs more concise knowledge bases. In general, by observing the Q_s scores computed for the HEX knowledge pieces, our proposed algorithm results capable of providing the highest quality knowledge.

Our regression case study employs six data sets from real use cases taken from the StairwAI EU Project and composed of up to 5 continuous input features. A different DT with maximum depth equal to 50 and an unbounded number of leaves has been trained for each data set. The pool of SKE algorithms adopted for the data sets is composed of CART, GRIDEX, GRIDREX and HEX. ITER proved incapable of providing comparable results, and therefore it is not considered in this case study.

Table 2: Results for the StairwAI data sets.

	DT	CART			GridEx			GridREx			HEx		
Data set	R^2	R^2	Rules	Q_s	R^2	Rules	Q_s	R^2	Rules	Q_s	R^2	Rules	Q_s
#1	1.00	0.93	4	0.299	0.64	**2**	0.713	**1.00**	**2**	**0.009**	**1.00**	**2**	**0.009**
#2	0.92	0.89	4	0.458	0.69	**2**	0.627	**0.90**	**2**	**0.199**	**0.90**	**2**	**0.199**
#3	1.00	0.70	4	1.200	0.34	**2**	1.316	**0.99**	**2**	**0.024**	**0.99**	**2**	**0.024**
#4	1.00	0.88	4	0.488	-1.44	**2**	4.879	**0.99**	**2**	**0.013**	**0.99**	**2**	**0.013**
#5	1.00	0.93	4	0.285	0.63	2	0.738	**1.00**	2	0.008	**1.00**	**1**	**0.005**
#6	0.99	**0.09**	4	3.648	-1.11	**2**	4.222	-0.19	**2**	**2.381**	-0.19	**2**	**2.381**

Several combinations of hyper-parameters have been tested for the extractors. We report in Table 2 the assessments conducted for the underlying black boxes and for the instances of knowledge extractors providing the best results for each black box. The coverage of SKE techniques is not shown, since it is equal to 1.00 for all of them. It is clear how GridREx and HEx, being able to describe the outputs of regression rules as linear functions, outperform CART and GridEx, only providing constant outputs. HEx provides results very similar to those obtained with GridREx, nonetheless for one data set it can achieve the same predictive performance of GridREx with a single regression rule, resulting in higher knowledge quality. This can be verified by comparing the Q_s score of HEx (0.005) with that of GridREx (0.008).

5 Conclusions

In this work, we introduce HEx to perform SKE from any kind of opaque ML model designed for classification or regression tasks with continuous input attributes. HEx demonstrates superior compared to existing alternative techniques in terms of predictive accuracy, completeness and human-interpretability extent.

The HEx algorithm ensures a total input feature space coverage thanks to its tree-based knowledge extraction. Human readability is achieved via a hierarchical input space partitioning aimed at identifying only relevant predictive rules to be included in a concise output knowledge. Finally, HEx offers sensitive enhancements in the predictive performance when applied to regression tasks given its non-constant predictions based on linear combinations of the input variables.

HEx can be tuned by users via a set of hyper-parameters, that may also be handled automatically with the existing PEDRO procedure, designed for similar SKE algorithms. This ensures that HEx provides high-quality knowledge with minimum user effort.

Our future research will focus on developing more sophisticated and effective regression rules, to overcome the limitations of constant values and linear combinations. Furthermore, we plan to extend the identification of decision boundaries from perpendicular to oblique with respect to the data set axes.

6 Acknowledgments

This work has been supported by PNRR – M4C2 – Investimento 1.3, Partenariato Esteso PE00000013 – "FAIR—Future Artificial Intelligence Research" – Spoke 8 "Pervasive AI", funded by the European Commission under the NextGenerationEU programme and by the European Union's Horizon Europe AEQUITAS research and innovation programme under grant number 101070363.

References

1. Altmann, A., Toloşi, L., Sander, O., Lengauer, T.: Permutation importance: a corrected feature importance measure. Bioinformatics **26**(10), 1340–1347 (2010)
2. Andrews, R., Diederich, J., Tickle, A.B.: Survey and critique of techniques for extracting rules from trained artificial neural networks. Knowledge-Based Systems **8**(6), 373–389 (1995). https://doi.org//10.1016/0950-7051(96)81920-4
3. Barbado, A., Corcho, Ó., Benjamins, R.: Rule extraction in unsupervised anomaly detection for model explainability: Application to oneclass SVM. Expert Syst. Appl. **189**, 116100 (2022). https://doi.org/10.1016/j.eswa.2021.116100, https://doi.org/10.1016/j.eswa.2021.116100
4. Barbiero, P., Ciravegna, G., Giannini, F., Liò, P., Gori, M., Melacci, S.: Entropy-based logic explanations of neural networks. In: Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022. pp. 6046–6054. AAAI Press (2022), https://ojs.aaai.org/index.php/AAAI/article/view/20551
5. Barredo Arrieta, A., Díaz-Rodríguez, N., Del Ser, J., Bennetot, A., Tabik, S., Barbado, A., Garcia, S., Gil-Lopez, S., Molina, D., Benjamins, R., Chatila, R., Herrera, F.: Explainable explainable artificial intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. Information Fusion **58**(December 2019), 82–115 (2020). https://doi.org/10.1016/j.inffus.2019.12.012
6. Becker, B., Kohavi, R.: Adult. UCI Machine Learning Repository (1996), DOI: https://doi.org/10.24432/C5XW20
7. Breiman, L., Friedman, J., Stone, C.J., Olshen, R.A.: Classification and Regression Trees. CRC Press (1984)
8. Burrell, J.: How the machine 'thinks': Understanding opacity in machine learning algorithms. Big Data & Society **3**(1) (2016). https://doi.org/10.1177/2053951715622512
9. Calegari, R., Sabbatini, F.: The PSyKE technology for trustworthy artificial intelligence **13796**, 3–16 (Mar 2023). https://doi.org/10.1007/978-3-031-27181-6_1, https://doi.org/10.1007/978-3-031-27181-6_1, xXI International Conference of the Italian Association for Artificial Intelligence, AIxIA 2022, Udine, Italy, November 28 – December 2, 2022, Proceedings
10. Calvaresi, D., Ciatto, G., Najjar, A., Aydoğan, R., Van der Torre, L., Omicini, A., Schumacher, M.: Expectation: Personalized explainable artificial intelligence for decentralized agents with heterogeneous knowledge. In: Calvaresi, D., Najjar, A., Winikoff, M., Främling, K. (eds.) Explainable and Transparent AI and Multi-Agent Systems. Third International Workshop, EXTRAAMAS 2021, Virtual Event, May

3–7, 2021, Revised Selected Papers, LNCS, vol. 12688, pp. 331–343. Springer Nature, Basel, Switzerland (2021). https://doi.org/10.1007/978-3-030-82017-6_20, http://link.springer.com/10.1007/978-3-030-82017-6_20
11. de Campos Souza, P.V., Lughofer, E.: Efnn-nulluni: An evolving fuzzy neural network based on null-uninorm. Fuzzy Sets Syst. **449**, 1–31 (2022). https://doi.org/10.1016/j.fss.2022.01.010, https://doi.org/10.1016/j.fss.2022.01.010
12. Ciravegna, G., Barbiero, P., Giannini, F., Gori, M., Liò, P., Maggini, M., Melacci, S.: Logic explained networks. Artif. Intell. **314**, 103822 (2023). https://doi.org/10.1016/j.artint.2022.103822, https://doi.org/10.1016/j.artint.2022.103822
13. Dattachaudhuri, A., Biswas, S.K., Chakraborty, M., Sarkar, S.: A transparent rule-based expert system using neural network. Soft Comput. **25**(12), 7731–7744 (2021). https://doi.org/10.1007/s00500-020-05547-7, https://doi.org/10.1007/s00500-020-05547-7
14. Demner-Fushman, D., Rogers, W.J., Aronson, A.R.: Metamap lite: an evaluation of a new java implementation of metamap. Journal of the American Medical Informatics Association **24**(4), 841–844 (2017)
15. Diao, H., Lu, Y., Deng, A., Zou, L., Li, X., Pedrycz, W.: Convolutional rule inference network based on belief rule-based system using an evidential reasoning approach. Knowl. Based Syst. **237**, 107713 (2022). https://doi.org/10.1016/j.knosys.2021.107713, https://doi.org/10.1016/j.knosys.2021.107713
16. Espinosa Zarlenga, M., Shams, Z., Jamnik, M.: Efficient decompositional rule extraction for deep neural networks. CoRR **abs/2111.12628** (2021), https://arxiv.org/abs/2111.12628
17. European Commission: AI Act – Proposal for a regulation of the european parliament and the council laying down harmonised rules on artificial intelligence (Artificial Intelligence Act) and amending certain union legislative acts. https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:52021PC0206 (2021)
18. European Commission, Directorate-General for Communications Networks, C., Technology: Ethics guidelines for trustworthy AI. Publications Office (2019). https://doi.org/doi/10.2759/346720
19. Fan, J., Kalyanpur, A., Gondek, D.C., Ferrucci, D.A.: Automatic knowledge extraction from documents. IBM Journal of Research and Development **56**(3.4), 5–1 (2012)
20. Ferreira, J., de Sousa Ribeiro, M., Gonçalves, R., Leite, J.: Looking inside the black-box: Logic-based explanations for neural networks. In: Kern-Isberner, G., Lakemeyer, G., Meyer, T. (eds.) Proceedings of the 19th International Conference on Principles of Knowledge Representation and Reasoning, KR 2022, Haifa, Israel. July 31 - August 5, 2022 (2022), https://proceedings.kr.org/2022/45/
21. Fisher, R.A.: The use of multiple measurements in taxonomic problems. Annals of Eugenics **7**(2), 179–188 (1936). https://doi.org/https://doi.org/10.1111/j.1469-1809.1936.tb02137.x, https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1469-1809.1936.tb02137.x
22. Guidotti, R., Monreale, A., Ruggieri, S., Turini, F., Giannotti, F., Pedreschi, D.: A survey of methods for explaining black box models. ACM Computing Surveys **51**(5), 1–42 (2018). https://doi.org/10.1145/3236009
23. Huysmans, J., Baesens, B., Vanthienen, J.: ITER: An algorithm for predictive regression rule extraction. In: Data Warehousing and Knowledge Discovery (DaWaK 2006). pp. 270–279. Springer (2006). https://doi.org/10.1007/11823728_26
24. Irfan, M., Zheng, J., Iqbal, M., Masood, Z., Arif, M.H.: Knowledge extraction and retention based continual learning by using convolutional autoencoder-based

learning classifier system. Inf. Sci. **591**, 287–305 (2022). https://doi.org/10.1016/j.ins.2022.01.043, https://doi.org/10.1016/j.ins.2022.01.043
25. Johansson, U., Sönströd, C., Löfström, T., Boström, H.: Rule extraction with guarantees from regression models. Pattern Recognit. **126**, 108554 (2022). https://doi.org/10.1016/j.patcog.2022.108554, https://doi.org/10.1016/j.patcog.2022.108554
26. Kenny, E.M., Ford, C., Quinn, M., Keane, M.T.: Explaining black-box classifiers using post-hoc explanations-by-example: The effect of explanations and error-rates in XAI user studies. Artificial Intelligence **294**, 103459 (2021). https://doi.org/10.1016/j.artint.2021.103459
27. Konig, R., Johansson, U., Niklasson, L.: G-REX: A versatile framework for evolutionary data mining. In: 2008 IEEE International Conference on Data Mining Workshops (ICDM 2008 Workshops). pp. 971–974 (2008). https://doi.org/10.1109/ICDMW.2008.117
28. Lipton, Z.C.: The mythos of model interpretability. Queue **16**(3), 31–57 (Jun 2018). https://doi.org/10.1145/3236386.3241340
29. Obregon, J., Jung, J.: Rulecosi+: Rule extraction for interpreting classification tree ensembles. Inf. Fusion **89**, 355–381 (2023). https://doi.org/10.1016/j.inffus.2022.08.021, https://doi.org/10.1016/j.inffus.2022.08.021
30. Rudin, C.: Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. Nature Machine Intelligence **1**(5), 206–215 (2019). https://doi.org/10.1038/s42256-019-0048-x
31. Sabbatini, F., Calegari, R.: Symbolic knowledge extraction from opaque machine learning predictors: GridREx & PEDRO. In: Kern-Isberner, G., Lakemeyer, G., Meyer, T. (eds.) Proceedings of the 19th International Conference on Principles of Knowledge Representation and Reasoning, KR 2022, Haifa, Israel. July 31 - August 5, 2022 (2022). https://doi.org/10.24963/kr.2022/57, https://proceedings.kr.org/2022/57/
32. Sabbatini, F., Calegari, R.: Achieving complete coverage with hypercube-based symbolic knowledge-extraction techniques. In: Nowaczyk, S., Biecek, P., Chung, N.C., Vallati, M., Skruch, P., Jaworek-Korjakowska, J., Parkinson, S., Nikitas, A., Atzmüller, M., Kliegr, T., et al. (eds.) Artificial Intelligence. ECAI 2023 International Workshops – XAI^3, TACTIFUL, XI-ML, SEDAMI, RAAIT, AI4S, HYDRA, AI4AI, Kraków, Poland, September 30 – October 4, 2023, Proceedings, Part I. Communications in Computer and Information Science, vol. 1947, pp. 179–197. Springer (2023). https://doi.org/10.1007/978-3-031-50396-2_10, https://doi.org/10.1007/978-3-031-50396-2_10
33. Sabbatini, F., Calegari, R.: Symbolic knowledge-extraction evaluation metrics: The FiRe score. In: Gal, K., Nowé, A., Nalepa, G.J., Fairstein, R., Rădulescu, R. (eds.) Proceedings of the 26th European Conference on Artificial Intelligence, ECAI 2023, Kraków, Poland. September 30 – October 4, 2023 (2023). https://doi.org/10.3233/FAIA230496, https://ebooks.iospress.nl/doi/10.3233/FAIA230496
34. Sabbatini, F., Calegari, R.: Unlocking insights and trust: The value of explainable clustering algorithms for cognitive agents. In: Falcone, R., Castelfranchi, C., Sapienza, A., Cantucci, F. (eds.) Proceedings of the 24th Workshop "From Objects to Agents", Roma, Italy, November 6–8, 2023. CEUR Workshop Proceedings, vol. 3579, pp. 232–245. CEUR-WS.org (2023), https://ceur-ws.org/Vol-3579/paper18.pdf
35. Sabbatini, F., Calegari, R.: Unveiling opaque predictors via explainable clustering: The CReEPy algorithm. In: Boella, G., D'Asaro, F.A., Dyoub, A., Gorrieri, L., Lisi, F.A., Manganini, C., Primiero, G. (eds.) Proceedings of the 2nd

Workshop on Bias, Ethical AI, Explainability and the role of Logic and Logic Programming co-located with the 22nd International Conference of the Italian Association for Artificial Intelligence (AI*IA 2023), Rome, Italy, November 6, 2023. CEUR Workshop Proceedings, vol. 3615, pp. 1–14. CEUR-WS.org (2023), https://ceur-ws.org/Vol-3615/paper1.pdf
36. Sabbatini, F., Calegari, R.: On the evaluation of the symbolic knowledge extracted from black boxes. AI and Ethics **4**(1), 65–74 (January 2024). https://doi.org/https://doi.org/10.1007/s43681-023-00406-1
37. Sabbatini, F., Calegari, R.: Untying black boxes with clustering-based symbolic knowledge extraction. Intelligenza Artificiale **18**(1), 21–34 (2024). https://doi.org/10.3233/IA-240026, https://doi.org/10.3233/IA-240026
38. Sabbatini, F., Ciatto, G., Calegari, R., Omicini, A.: Hypercube-based methods for symbolic knowledge extraction: Towards a unified model. In: Ferrando, A., Mascardi, V. (eds.) WOA 2022 – 23rd Workshop "From Objects to Agents", CEUR Workshop Proceedings, vol. 3261, pp. 48–60. Sun SITE Central Europe, RWTH Aachen University (Nov 2022), http://ceur-ws.org/Vol-3261/paper4.pdf
39. Sabbatini, F., Ciatto, G., Calegari, R., Omicini, A.: Symbolic knowledge extraction from opaque ML predictors in PSyKE: Platform design & experiments. Intelligenza Artificiale **16**(1), 27–48 (2022). https://doi.org/10.3233/IA-210120, https://doi.org/10.3233/IA-210120
40. Sabbatini, F., Ciatto, G., Calegari, R., Omicini, A.: Towards a unified model for symbolic knowledge extraction with hypercube-based methods. Intelligenza Artificiale **17**(1), 63–75 (2023). https://doi.org/10.3233/IA-230001, https://doi.org/10.3233/IA-230001
41. Sabbatini, F., Ciatto, G., Omicini, A.: GridEx: An algorithm for knowledge extraction from black-box regressors. In: Calvaresi, D., Najjar, A., Winikoff, M., Främling, K. (eds.) Explainable and Transparent AI and Multi-Agent Systems. Third International Workshop, EXTRAAMAS 2021, Virtual Event, May 3–7, 2021, Revised Selected Papers, LNCS, vol. 12688, pp. 18–38. Springer Nature, Basel, Switzerland (2021). https://doi.org/10.1007/978-3-030-82017-6_2
42. Sabbatini, F., Grimani, C., Calegari, R.: Bridging machine learning and diagnostics of the esa lisa space mission with equation discovery via explainable artificial intelligence. Advances in Space Research **74**(1), 505–517 (2024). https://doi.org/https://doi.org/10.1016/j.asr.2024.04.041, https://www.sciencedirect.com/science/article/pii/S0273117724003880
43. Salimi-Badr, A., Ebadzadeh, M.M.: A novel learning algorithm based on computing the rules' desired outputs of a TSK fuzzy neural network with non-separable fuzzy rules. Neurocomputing **470**, 139–153 (2022). https://doi.org/10.1016/j.neucom.2021.10.103, https://doi.org/10.1016/j.neucom.2021.10.103
44. Schmitz, G.P.J., Aldrich, C., Gouws, F.S.: ANN-DT: an algorithm for extraction of decision trees from artificial neural networks. IEEE Transactions on Neural Networks **10**(6), 1392–1401 (1999). https://doi.org/10.1109/72.809084
45. Setiono, R., Leow, W.K., Zurada, J.M.: Extraction of rules from artificial neural networks for nonlinear regression. IEEE Transactions on Neural Networks **13**(3), 564–577 (2002). https://doi.org/10.1109/TNN.2002.1000125
46. Setiono, R., Liu, H.: NeuroLinear: From neural networks to oblique decision rules. Neurocomputing **17**(1), 1–24 (1997). https://doi.org/10.1016/S0925-2312(97)00038-6, https://doi.org/10.1016/S0925-2312(97)00038-6
47. Smith, C.A., Hetzel, S., Dalrymple, P., Keselman, A.: Beyond readability: investigating coherence of clinical text for consumers. Journal of medical Internet research **13**(4), e1842 (2011)

48. Towell, G.G., Shavlik, J.W.: Extracting refined rules from knowledge-based neural networks. Machine Learning **13**(1), 71–101 (1993). https://doi.org/10.1007/BF00993103
49. Zhuang, J., Dvornek, N.C., Li, X., Yang, J., Duncan, J.: Decision explanation and feature importance for invertible networks. In: 2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW). pp. 4235–4239. IEEE (2019)
50. Zien, A., Krämer, N., Sonnenburg, S., Rätsch, G.: The feature importance ranking measure. In: Joint European Conference on Machine Learning and Knowledge Discovery in Databases (ECML PKDD 2009). pp. 694–709. Springer (2009). https://doi.org/10.1007/978-3-642-04174-7_45