shapiq: Shapley Interactions for Machine Learning

Maximilian Muschalik[*]
LMU Munich, MCML

Hubert Baniecki
University of Warsaw,
Warsaw University of Technology

Fabian Fumagalli
Bielefeld University

Patrick Kolpaczki
Paderborn University

Barbara Hammer
Bielefeld University

Eyke Hüllermeier
LMU Munich, MCML

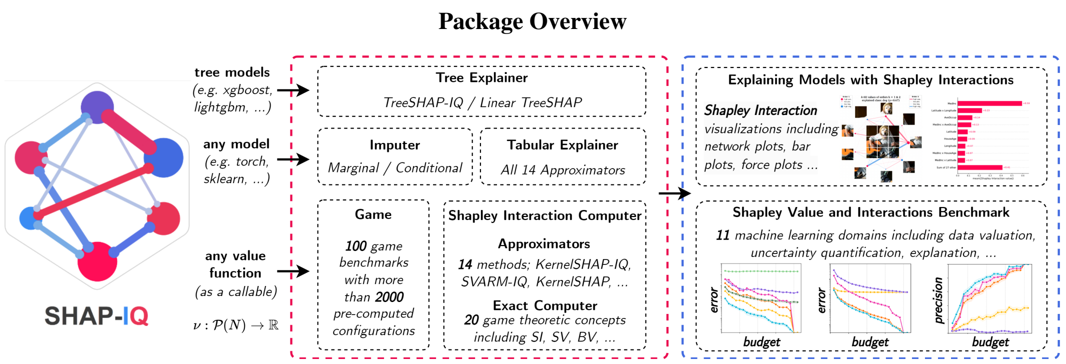
Abstract

Originally rooted in game theory, the Shapley Value (SV) has recently become an important tool in machine learning research. Perhaps most notably, it is used for feature attribution and data valuation in explainable artificial intelligence. Shapley Interactions (SIs) naturally extend the SV and address its limitations by assigning joint contributions to groups of entities, which enhance understanding of black box machine learning models. Due to the exponential complexity of computing SVs and SIs, various methods have been proposed that exploit structural assumptions or yield probabilistic estimates given limited resources. In this work, we introduce `shapiq`, an open-source Python package that unifies state-of-the-art algorithms to efficiently compute SVs and any-order SIs in an application-agnostic framework. Moreover, it includes a benchmarking suite containing 11 machine learning applications of SIs with pre-computed games and ground-truth values to systematically assess computational performance across domains. For practitioners, `shapiq` is able to explain and visualize any-order feature interactions in predictions of models, including vision transformers, language models, as well as XGBoost and LightGBM with TreeSHAP-IQ. With `shapiq`, we extend `shap` beyond feature attributions and consolidate the application of SVs and SIs in machine learning that facilitates future research. The source code and documentation are available at `https://github.com/mmschlk/shapiq`.

1 Introduction

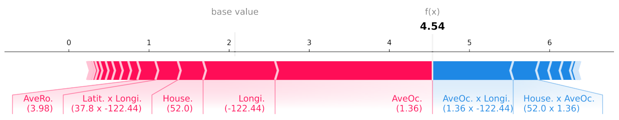
Assigning *value* to entities collectively performing a task is essential in various real-world applications of machine learning (ML) [62, 73]. For instance, when reimbursing data providers based on the *value of data* [30, 78], or justifying a model's prediction based on *value of feature information* [13, 18, 19, 54, 75]. The *fair* distribution of value among a group of entities is a central aspect of cooperative game theory, where the Shapley Value (SV) [74] defines a *unique* allocation scheme based on intuitive axioms. The SV is applicable to any *game*, i.e. a function that specifies the worth of all possible groups of entities, called coalitions. In ML, application-specific games were introduced [5, 30, 73, 78, 82], which typically require a definition of the overall worth and a notion of entities' absence [19]. The SV fairly distributes the overall worth among individuals by evaluating the game for all coalitions. However, it does not give insights on *synergies or redundancies* between entities. For instance, while two features such as *latitude* and *longitude* convey separate information, only their joint consideration reveals the synergy of encoding an *exact location*. The value of such a group of entities is known as an *interaction* [33], or in this context *feature interaction* [29], and is crucial to understand predictions of complex ML models [20, 47, 48, 60, 65, 76, 80, 84], as illustrated in Figure 1.

[*]Corresponding author (`maximilian.muschalik@ifi.lmu.de`)


Package Overview
SHAP-IQ
tree models
(e.g. xgboost, lightgbm, ...)
any model
(e.g. torch, sklearn, ...)
any value function
(as a callable)
$\nu : \mathcal{P}(N) \rightarrow \mathbb{R}$
Tree Explainer
TreeSHAP-IQ / Linear TreeSHAP
Imputer
Marginal / Conditional
Tabular Explainer
All 14 Approximators
Game
100 game benchmarks with more than 2000 pre-computed configurations
Shapley Interaction Computer
Approximators
14 methods; KernelSHAP-IQ, SVARM-IQ, KernelSHAP, ...
Exact Computer
20 game theoretic concepts including SI, SV, BV, ...
Explaining Models with Shapley Interactions
Shapley Interaction visualizations including network plots, bar plots, force plots ...
Shapley Value and Interactions Benchmark
11 machine learning domains including data valuation, uncertainty quantification, explanation, ...
error
budget
error
budget
precision
budget

(a) Explain Models with Shapley Interactions

```
# get your data and model
X, model = ...
from shapiq import Explainer
# create an explainer object
explainer = Explainer(model=model, data=X, max_order=2)
# get the feature interactions for the first observation
interaction_values = explainer.explain(X[0], budget=1024)
# visualize the 2-order feature interactions
interaction_values.force_plot(feature_names=...)
```



(b) Domain-Independent Game Theory

```
import shapiq
class CountGame(shapiq.Game):
  def __init__(self, n_players): ...
  def value_function(coalitions):
    # define the worth of a coalition
    return np.sum(coalitions, axis=1)
game = CountGame(n_players=12)
# approximate SIs with KernelSHAP-IQ
approx = shapiq.KernelSHAPIQ(n=12)
si = approx(game=game, budget=1000)
# compute the Moebius transform exactly
exact = shapiq.ExactComputer(game, 12)
mi = exact(index='Moebius')
print(si[(3, 7)], mi[(3,)]) # get values
>> 0.00, 1.00
```

Figure 1: The `shapiq` Python package facilitates research on game theory for machine learning, including state-of-the-art approximation algorithms and pre-computed benchmarks. It also provides a simple interface for explaining predictions **(a)** of machine learning models beyond feature attributions, enabling researchers to explore higher-order interactions or domain-independent game theory **(b)**.

Shapley Interactions (SIs) [10, 33, 76, 79] distribute the overall worth to all groups of entities up to a maximum *explanation order*. They satisfy axioms similar to the SV, to which they reduce for individuals, i.e. the lowest explanation order. In contrast, for the highest explanation order, which comprises an interaction for every coalition, the SIs yield the Möbius Interaction (MI), or Möbius transform [27, 35, 71]. The MIs are a fundamental concept in cooperative game theory that captures the *isolated joint contribution*, which allows to additively describe every coalition's worth by a sum of the contained MIs. With an increasing explanation order, the SIs comprise more components that finally yield the MIs as the most comprehensive explanation of the game at the cost of highest complexity [10, 79]. While the SV and SIs provide an appealing theoretical concept, computing them without structural assumptions on the game requires exponential complexity [22]. For tree-based models, it was shown that SVs [53, 85] and SIs [60, 86] can be efficiently computed by exploiting the architecture. Moreover, game-agnostic stochastic approximators estimate the SV [11, 17, 44, 54, 56, 63, 67] and SIs [28, 29, 46, 76, 79] with a limited budget of game evaluations.

Diverse applications of the SV have led to various techniques for its efficient computation [13]. Recently, extensions to any-order SIs addressed limitations of the SV and complemented interpretation of model predictions with higher-order feature interactions [10, 29, 76, 79, 80]. While stochastic approximators are applicable to any game, their evaluation is typically performed in an isolated application [29, 49], such as feature interactions. Moreover, implementing such algorithms requires a strong mathematical background and specific design choices. Existing Python packages, such as `shap` [54], provide a relatively small number of approximators, which are limited to the SV and feature attributions.

Contribution. In this work, we present shapiq, an open-source *Python* library for any-order SIs that consolidates research for computing SVs and SIs across ML domains. Our core contributions are summarized in Figure 1 and include,

(1) a general approximation and computation interface for state-of-the art SI algorithms and methods without focus on a specific application like explanations or data valuation,

(2) an explanation API for using SIs to explain ML models and visualizing interactions,

(3) a benchmarking suite to evaluate SI approximators across several real-world scenarios,

(4) and a cross-domain empirical evaluation of approximators guiding practitioners.

Related software tools and benchmarks. shapiq extends the popular shap [54] Python package beyond feature attributions aiming to fully embrace the application of SVs and SIs in ML. While shap implements a single index for 2-order feature interactions to explain the predictions of tree-based models, shapiq implements a dozen approximators for any-order SIs (Table 2) and offers a benchmarking suite for these algorithms across 10 different domains (Table 3). Related software such as aix360 [4], alibi [41] and dalex [6] are general toolboxes offering the implementation and visualization of the most popular ML explanations for end-users. We specify in SIs to provide a comprehensive tool facilitating research in game theory for ML, including the exact computation of 20 interaction indices and game-theoretic concepts (Table 1). Notably, the innvestigate [3] and captum [43] Python packages offer feature attribution methods specific to (deep) neural networks. Most recently, quantus [36] implements evaluation metrics for these explanation methods.

We build upon recent advances in benchmarking explainable artificial intelligence (XAI) methods such as feature attributions [2, 50, 52, 62] and algorithms for data valuation [38]. XAI-Bench [52] focuses on synthetic tabular data. OpenXAI [2] provides 7 real-world tabular datasets with pre-trained neural network models, 7 feature attribution methods and 8 metrics to compare them. $\mathcal{M}^4$ [50] extends OpenXAI to benchmark feature attributions of deep neural networks for image and text modalities. In [62], the authors benchmark several algorithms for approximating SVs based on the conditional feature distribution. OpenDataVal [38] provides 9 real-world datasets, 11 data valuation methods and 4 metrics to compare them. shapiq puts more focus on benchmarking higher-order SI algorithms and provides an interface to state-of-the-art explanation methods that base on SIs, e.g. TreeSHAP-IQ [60]. While open data repositories such as OpenML [9] offer easy access to datasets for ML, we pre-compute and share ground-truth SIs for various games (i.e. dataset–model pairs) that saves considerate time and resources when benchmarking approximation algorithms.

2 Theoretical Background

In ML, various concepts are based on synergies of entities to optimize performance in a given task. For example, weak learners construct powerful model ensembles [72], collected data instances and features are used to train supervised ML models [16, 30], where feature values collectively predict outputs. To better understand such processes, XAI quantifies the contributions of these entities to the task, most prominently for feature values in predictions (local feature attribution [54, 75]), features in models (global feature importance [16, 17, 68]), and data instances in model training (data valuation [30]). Assigning such contributions is closely related to the field of cooperative game theory [73], which studies the notion of value for players that collectively obtain a payout. To adequately assess the impact of individual players, it is necessary to analyze the payout for different coalitions. More formally, a cooperative game $\nu : \mathcal{P}(N) \to \mathbb{R}$ with $\nu(\emptyset) = 0$ is defined by a *value function* on the power set of $N := \{1, \ldots, n\}$ entities, which describes such payouts for all possible coalitions of players. We later summarize such prominent examples in the context of ML in Table 3. Here, we summarize existing contribution concepts for individuals and groups of entities, outlined in Table 1.

The **SV** [74] and **Banzhaf Value (BV)** [8] are instances of *semivalues* [23]. Semivalues assign contributions to individual players and adhere to intuitive axioms: *Linearity* enforces linearly composed contributions for linearly composed games; *Dummy* requires that players without impact receive zero contribution; *Symmetry* enforces that entities contributing equally to the payout receive equal value. The SV [74] is the unique semivalue that additionally satisfies *efficiency*, i.e. the sum of all contributions yields the total payout $\nu(N)$. In contrast, the BV [8] is the unique semivalue that additionally satisfies *2-Efficiency*, i.e. the contributions of two players sum to the contribution of a

Table 1: Available concepts in the `ExactComputer` class in `shapiq` with **SIs** in **bold**.

Setting	Interaction Index (II) [27]	Base Semivalue [23]	Generalized Value (GV) [57]
Machine Learning	**k-Shapley Values (k-SII)** [10]		
	Shapley Taylor II (STII) [76]	**Shapley (SV)** [74]	Joint SV [34]
	Faithful Shapley II (FSII) [79]		
	k_{ADD}-SHAP [67]		
	Faithful Banzhaf II (FBII) [79]	Banzhaf (BV) [8]	–
Game Theory	**Möbius (MI)** [27, 35, 71]	–	Internal GV (IGV) [57]
	Co-Möbius (Co-MI) [32]		External GV (EGV) [57]
	Shapley II (SII) [33]		Shapley GV (SGV) [58]
	Chaining II (CHII) [59]	**Shapley (SV)** [74]	Chaining GV (CHGV) [57]
	Banzhaf II (BII) [33]	Banzhaf (BV) [8]	Banzhaf GV (BGV) [59]

joint player in a reduced game, where both players are merged. The SV and BV are represented as a weighted average over *marginal contributions* $\Delta_i(T) := \nu(T \cup \{i\}) - \nu(T)$ for $i \in N$ as

$$\phi^{\text{SV}}(i) := \sum_{T \subseteq N \setminus \{i\}} \frac{1}{n\binom{n-1}{|T|}} \Delta_i(T) \qquad \text{and} \qquad \phi^{\text{BV}}(i) := \sum_{T \subseteq N \setminus \{i\}} \frac{1}{2^{n-1}} \Delta_i(T) .$$

In ML applications, the SV is typically preferred over the BV due to the efficiency axiom [73]. For instance, in local feature attribution, the SV is utilized to fairly distribute the model's prediction to individual features [54, 75]. However, it was shown that the SV is limited when explaining complex decision systems, and *feature interactions*, i.e. the joint contributions of features' groups, are required to understand such processes [20, 29, 47, 48, 60, 65, 76, 79, 80, 84].

The **Generalized Value (GV)** [57] and **Interaction Index (II)** [27] are two paradigms to extend the notion of value to groups of entities. The GVs are based on weighted averages over (joint) marginal contributions $\nu(T \cup S) - \nu(T)$ for $S \subseteq N$ given $T \subseteq N \setminus S$. In contrast, IIs are based on discrete derivatives that account for lower-order effects of subsets of S. For instance, for two players $i, j \in N$, the discrete derivative $\Delta_{\{i,j\}}(T)$ is defined as the joint marginal contribution $\nu(T \cup \{i,j\}) - \nu(T)$ minus the individual marginal contributions $\Delta_i(T)$ and $\Delta_j(T)$. More generally, the *discrete derivative* $\Delta_S(T)$ for $S \subseteq N$ in the presence of $T \subseteq N \setminus S$ is defined as

$$\Delta_S(T) := \sum_{L \subseteq S} (-1)^{|S|-|L|} \nu(T \cup L) \quad \text{with} \quad \Delta_S(T) = \underbrace{\nu(T \cup S) - \nu(T)}_{\text{joint marginal contribution}} - \sum_{\emptyset \neq L \subset S} \underbrace{\Delta_L(T)}_{\text{lower-order effects}} .$$

A positive value indicates synergy, whereas a negative value indicates redundancy of S given T. Lastly, a zero value indicates (additive) independence, i.e. the joint marginal contribution is equal to the sum of all lower-order effects. GVs and IIs are uniquely represented [27, 57] by

$$\phi^{\text{GV}}(S) := \sum_{T \subseteq N \setminus S} p_{|T|}^{|S|}(n) \left(\nu(T \cup S) - \nu(T)\right) \qquad \text{and} \qquad \phi^{\text{II}}(S) := \sum_{T \subseteq N \setminus S} p_{|T|}^{|S|}(n) \Delta_S(T) .$$

The most prominent examples are the *Shapley GV (SGV)* [58] and the *Shapley II (SII)* [33] with $p_t^s(n) = \left((n-s+1)\binom{n-s}{t}\right)^{-1}$, which naturally extend the SV (cf. Appendix A.1). While the SGV and SII are natural extensions to the SV, they are not suitable for interpretability, since they are defined on the powerset and comprise an exponential number of components. Moreover, neither GVs nor IIs satisfy the efficiency axiom for higher-orders, which is desirable for ML applications.

Shapley Interactions (SIs) for Machine Learning assign joint contribution Φ_k up to an *explanation order* k, i.e. for all coalitions $S \subseteq N$ with $|S| \leq k$, which satisfy generalized *efficiency* $\nu(N) = \sum_{S \subseteq N, |S| \leq k} \Phi_k(S)$. The *$k$-SVs ($k$-SIIs)* [10] are the unique SIs that coincide with SII for the highest order. The *Shapley Taylor II (STII)* [76] puts a stronger emphasis on the top-order interactions, and *Faithful SII (FSII)* [79] optimizes *Shapley-weighted faithfulness*

$$\mathcal{L}(\nu, \Phi_k) := \sum_{T \subseteq N} \mu(t) \left(\nu(T) - \sum_{S \subseteq T, |S| \leq k} \Phi_k(S)\right)^2 \text{ with } \mu(t) := \begin{cases} \mu_\infty & \text{if } t \in \{0, n\} \\ \frac{1}{\binom{n-2}{t-1}} & \text{else} \end{cases} .$$

Table 2: Overview of methods in `shapiq` and applicable SIs. Explainers rely on approximators or model assumptions. (✓) indicates only top-order approximation.

Class	Implementation	Source	SV	(k-)SII	STII	FSII
Approximator	SHAP-IQ	[29]	✓	✓	✓	(✓)
	SVARM-IQ	[46]	✓	✓	✓	(✓)
	Permutation Sampling (SII)	[79]	✓	✓	–	–
	Permutation Sampling (STII)	[76]	✓	–	✓	–
	KernelSHAP-IQ	[28]	✓	✓	–	–
	Inconsistent KernelSHAP-IQ	[28]	✓	✓	–	–
	FSII Regression	[79]	✓	–	–	✓
	KernelSHAP	[54]	✓	–	–	–
	k_{ADD}-SHAP	[67]	✓	–	–	–
	Unbiased KernelSHAP	[17]	✓	–	–	–
	SVARM	[44]	✓	–	–	–
	Permutation Sampling	[11]	✓	–	–	–
	Owen Sampling	[63]	✓	–	–	–
	Stratified Sampling	[56]	✓	–	–	–
Explainer	Agnostic (Marginal)	–	✓	✓	✓	✓
	Agnostic (Conditional)	–	✓	✓	✓	✓
	TreeSHAP-IQ	[60]	✓	✓	✓	(✓)
	Linear TreeSHAP	[53, 85]	✓	–	–	–
Computer	Möbius Converter	–	✓	✓	✓	✓
	Exact Computer	–	✓	✓	✓	✓

FSII is thus $\Phi_k^{\text{FSII}} := \arg\min_{\Phi_k} \mathcal{L}(\nu, \Phi_k)$, where $\mu_\infty \gg 1$ ensures efficiency. It was recently shown that pairwise SII and consequently k-SII with $k = 2$ optimize a faithfulness metric with slightly different weights [28]. For $k = 1$, all SIs reduce to the SV $\Phi_1 \equiv \phi^{\text{SV}}$, which minimizes faithfulness $\mathcal{L}(\nu, \Phi_1)$ [12] with the efficiency constraint, or equivalently $\mu_\infty \to \infty$ [28, 54]. Finally, for $k = n$, all SIs Φ_n are the *MIs* (cf. Appendix A.2), which are faithful to all game values, i.e. $\mathcal{L}(\nu, \Phi_n) = 0$. Notably, all SIs can be uniquely represented by the MIs [31]. In this context, SIs yield a complexity-accuracy trade-off, ranging from the least complex (SV) to the most comprehensive (MI) explanation. Other extensions include *k_{ADD}-SHAP* [67] of the SV and *Faithful BII (FBII)* [79] of the BV, which do not satisfy efficiency, as well as *Joint SVs* [34], a GV with efficiency. However, in the context of feature interactions and ML, SIs are preferred over GV-based (Joint SVs) or BV-based IIs (FBII), as they account for lower-order *interactions* and adhere to the SV and MI as edge cases.

3 Overview of the `shapiq` Python package

The `shapiq` package accelerates research on SIs for ML, and provides an intutitve interface for explaining any-order feature interactions in predictions of ML models. Its code is open-source on GitHub at `https://github.com/mmschlk` while the documentation with notebook examples and API reference is available at `https://shapiq.readthedocs.io`.

3.1 `shapiq` Facilitates Research on Shapley Interactions for Machine Learning

While SVs have been predominantly applied to explanation use cases by leveraging the `shap` [54] library, `shapiq` provides a more abstract perspective on cooperative games and allows researchers from various fields to interact with and extend the framework to incorporate SIs and SVs.

Approximators. We implement 7 algorithms for approximating SIs across 4 different interaction indices, and another 7 algorithms for approximating SVs. Table 2 provides a comprehensive overview of this effort, where the `shapiq.Approximator` class is extended with each implementation. We unify common approximation methods by including a general `shapiq.CoalitionSampler` interface offering approximation performance increases through sampling procedures like the *border-* and *pairing-tricks* introduced in [17, 29].

```python
X, model = ...
import shapiq
# create an explainer object
explainer = shapiq.Explainer(model=model, data=X, max_order=3)
# choose a sample point to be explained
x = X[0]
# approximate feature interactions given the specified budget
interaction_values = explainer.explain(x=x, budget=1024)
# retrieve 3-order feature interactions
interaction_values.get_n_order_values(3)
# visualize 1-order and 2-order feature interactions on a graph
interaction_values.plot_network(feature_names=...)
```

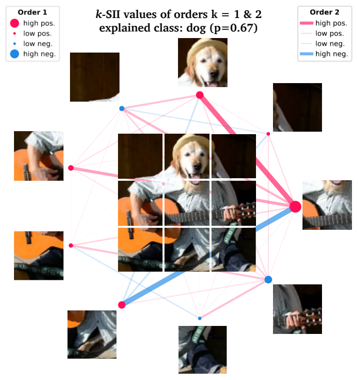


Figure 2: Left: Exemplary code for locally explaining a single model's prediction with `shapiq`. **Right:** Local feature interactions visualized on a network plot.

```python
X, model = ...
import shapiq
# create an explainer object
explainer = shapiq.Explainer(model=model, data=X, max_order=3)
# approximate feature interactions for multiple sample points
interaction_values_list = explainer.explain_X(X, budget=1024)
# retrieve 3-order feature interactions for the first point
interaction_values_list[0].get_n_order_values(3)
# visualize the global feature interaction importance
shapiq.plot.bar_plot(interaction_values_list, feature_names=...)
```

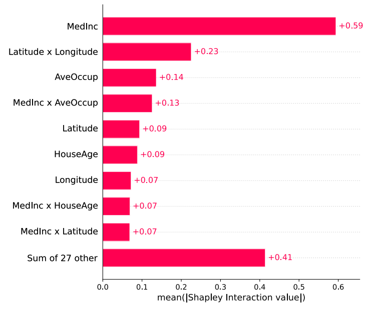


Figure 3: Left: Exemplary code for globally explaining multiple model's predictions with `shapiq`. **Right:** Global feature interaction importance visualized as a bar plot.

Exact computer. A key functionality of `shapiq` lies in computing the SIs *exactly*, which is feasible for smaller games, but reaches its limit for growing player numbers. The `shapiq.ExactComputer` class provides an interface for computing 20 interaction indices and game-theoretic concepts, including the MI, BV, SGV, among others (see Table 1).

Games. Approximators and computers require the specification of a *cooperative game*. Games make up a central level of abstraction in `shapiq`, and specifying a game only requires the implementation of a domain-specific value function. Table 3 describes in detail 11 benchmark games implemented in `shapiq`. Beyond synthetic games, our benchmark spans the 5 most prominent domains where SIs can be applied for ML. The `shapiq.Game` class can be easily extended to include future benchmarks in the package. We pre-compute and share exact SIs for 2 042 benchmark game configurations in total (see Appendix B), facilitating future work on improving the approximators, which we elaborate on further in Section 4.

3.2 Explaining Machine Learning Predictions with `shapiq`

In addition to facilitating theoretical advancements, `shapiq` also provides practical tools for applying SIs in ML. These tools streamline the process of explaining feature interactions in model predictions, allowing researchers and practitioners to easily compute and visualize interaction effects across a range of models and data types.

Explainer. The `shapiq.Explainer` class is a simplified interface to explain any-order feature interactions in ML models. Figure 2 goes through exemplary code used to approximate SIs for a single prediction and visualize them on a graph plot. Currently two classes are further distinguished within the API, but we envision extending `shapiq.Explainer` to include more data modalities and model algorithms. `shapiq.TabularExplainer` allows for model-agnostic explanation based on feature marginalization with either marginal or conditional imputation (refer to Appendix C for details). `shapiq.TreeExplainer` implements TreeSHAP-IQ [60] for efficient explanations

Table 3: Overview of the available benchmark games and domains in `shapiq`. Each benchmark can be instantiated with different datasets, models, player sizes, or benchmark-specific configuration parameters. This results in 2 042 pre-computed individual configurations (see Tables 4 and 5).

Domain	Benchmark (Game)	Source	Players	Coalition Worth
XAI	Local Explanation	[54, 75]	Features	Model Output
	Global Explanation	[18]	Features	Model Loss
	Tree Explanation	[53, 60]	Features	Model Output
Uncertainty	Uncertainty Explanation	[82]	Features	Prediction Entropy
Model Selection	Feature Selection	[16, 68]	Features	Performance
	Ensemble Selection	[72]	Weak Learners	Performance
	RF Ensemble Selection	[72]	Tree Models	Performance
Valuation	Data Valuation	[30]	Data Points	Performance
	Dataset Valuation	[78]	Data Subsets	Performance
Unsupervised Learning	Cluster Explanation	–	Features	Cluster Score
	Unsupervised Feature Importance	[5]	Features	Total Correlation
Synthetic	Sum of Unanimity Model	[28, 79]	Players	Sum of Unanimous Votes

specific to decision tree-based models, e.g. random forest or gradient boosting decision trees, with native support for `scikit-learn` [66], `xgboost` [15], and `lightgbm` [39]. Figure 3 goes through exemplary code for explaining a set of predictions and visualizing their aggregation in a bar plot, which represents the global feature interaction importance.

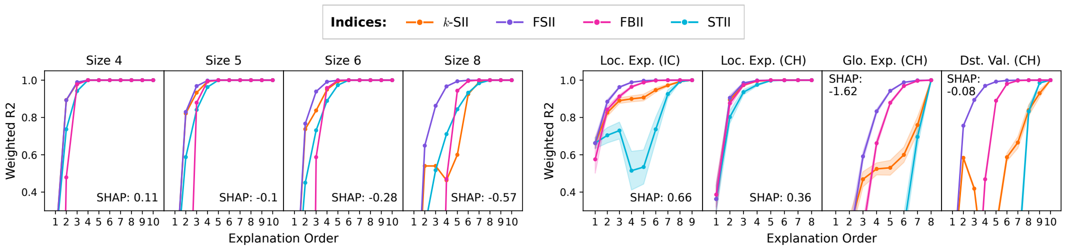
Visualizing interactions. `shapiq` supports the plotting and analysis of interaction values with different visualizations techniques. `shapiq.plot` offers custom visualizations including our custom network and SI graph plot, but also wraps established visualizations from `shap` [54] like the `force` and `bar` plots. For a detailed guide and summary of the visualizations, we kindly refer to Appendix D.

Utility functions. `shapiq` offers additional useful tools that are described in detail in the documentation. Interaction values are stored and processed using the `shapiq.InteractionValues` data class, which is rich in utility functions. Notably, useful set-based operators and generators exist for handling player sets $S \subseteq N$ or iterating over power sets $\mathcal{P}(N)$ of certain sizes with `shapiq.powerset`. Finally, `shapiq.datasets` loads datasets used for testing and examples.

4 Benchmarking Analysis

The `shapiq` library enables computation of various SIs for a broad class of application domains. To illustrate its versatility, we conduct benchmarks across a wide variety of traditional ML-based SV application scenarios. The ML benchmark demonstrates how higher-order SIs enable an accuracy–complexity trade-off for model interpretability (Section 4.1) and highlights that different approximation techniques in `shapiq` achieve the state-of-the-art performance depending on the application domains (Section 4.2). Tables 3, 4 and 5 present an overview of different application domains and associated benchmarks. Depending on the benchmark, it can be instantiated with different datasets, models, player numbers or benchmark-specific configuration parameters, e.g. uncertainty type: *epistemic* for Uncertainty Explanation or imputer: *conditional* for Local Explanation. In total, `shapiq` offers 100 unique benchmark games, i.e. applications times dataset–model pairs.

For all games that include $n \leq 16$ players, the value functions have been pre-computed by evaluating all coalitions and storing the games to file. Reading a pre-computed game from file, instead of performing up to $2^{16} = 65\,536$ value function calls with each new experiment run, saves valuable computational time and contributes to reproducibility as well as sustainability. This is particularly beneficial for tasks that involve remove-and-refit strategies [19], such as Data Valuation or Feature Selection. For $n > 16$, where pre-computing a game and ground truth values becomes computationally prohibitive, we rely on analytical solutions to compute the ground truth like TreeSHAP-IQ [60] for tree-based ensembles or the MI representation of Sum of Unanimity Models [28, 29, 79]. For details regarding the experimental setting and reproducibility, we refer to Appendix E.



(a) Single synthetic interaction **(b) ML applications**

Figure 4: Shapley-weighted R^2 of interaction indices by explanation order for **(a)** single synthetic interactions and **(b)** ML applications. FSII is optimized for this metric, and increases faithfulness with each order. Interactions improve faithfulness over SHAP and yield an exact decomposition for the highest order. However, increasing interaction size negatively affects faithfulness.

4.1 Accuracy – Complexity Trade-Off of Shapley Interactions for Interpretability

In this experiment, we empirically assess how *faithfully* lower-order SIs representations capture higher-order effects for varying explanation orders k and choice of interaction index. To this end, we rely on the Shapley-weighted faithfulness $\mathcal{L}(\nu, \Phi_k)$ introduced in Section 2. The complexity of SIs ranges from SVs (least complex) to MIs (most complex), where SVs minimize $\mathcal{L}(\nu, \Phi_1)$ for $k = 1$ [12, 54], while MIs perfectly capture all game values with $\mathcal{L}(\nu, \Phi_n) = 0$ for $k = n$ [10, 79]. To quantify the faithfulness of SIs across different domains, we calculate interaction values for a given index Φ_k and explanation order k. We then approximate the game values for each subset $T \subseteq N$ of players as $\hat{\nu}_k(T) := \sum_{S \subseteq T : |S| \leq k} \Phi_k(S)$ and compare them to the ground truths using $\mathcal{L}(\nu, \Phi_k)$ through a Shapley-weighted R^2 $(\nu(T), \hat{\nu}_k(T))$ for all $T \subseteq N$. For context, a game with no interactions (a 1-additive game) will be perfectly reproduced by a 1-additive explanation, yielding $\mathcal{L}(\nu, \Phi_1) = 0$ and $R^2 = 1$. Conversely, games with substantial higher-order interactions will result in larger errors for lower-order explanations, with $\mathcal{L}(\nu, \Phi_k) \gg 0$ and $R^2 < 1$.

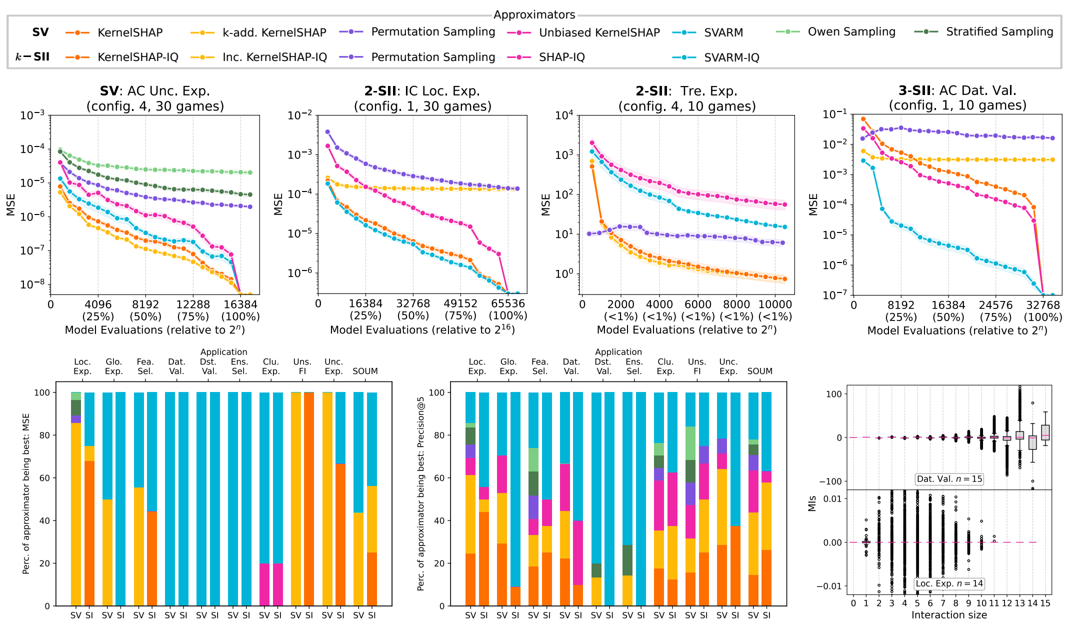
Figure 4 shows the Shapley-weighted R^2 value for $k = 1, \ldots, n$ for a synthetic game with a single interaction of varying size (a) and real-world ML applications (b). Here, we used $\mu_\infty = 1$ instead of $\mu_\infty \gg 1$, which affects FBII that violates efficiency. The results show that in general SIs become more faithful with higher explanation order. Notably, the difference between pairwise SIs and SVs (SHAP textbox) is remarkable, where pairwise interactions ($k = 2$) already yield a strong improvement in faithfulness. If higher-order interactions dominate, then SIs require a larger explanation order to maintain faithfulness. While FSII and FBII are optimized for faithfulness, STII and k-SII do not necessarily yield a strict improvement in this metric. In fact, it was shown that SII and k-SII optimize a slightly different faithfulness metric, which changes for every order [28]. Yet, we observe a consistent strong improvement of pairwise k-SII over the SV (SHAP). While FSII and FBII optimize faithfulness, k-SII and STII adhere to strict structural assumptions, where STII projects all higher-order interactions to the top-order SIs, and k-SII is consistent with SII. Practitioners may choose SIs tailored to their specific application, where k-SII is a good default choice for `shapiq`.

4.2 Comparison of Approximation Methods

Various approximation methods for computing SIs are included in `shapiq` for a variety of SIs (cf. Table 2). The possibility of attributing (domain-specific) state-of-the-art performance to a single algorithm has been investigated empirically by multiple works [28, 29, 44–46, 56, 63, 67, 79, 81, 87]. We use the collection of 100 unique benchmark games in `shapiq` to evaluate the performance of different SV and SI approximation methods on a broad spectrum of ML applications. For each domain and configuration (see Table 4 and 5 in Appendix B), we compute ground truth SVs, 2-SIIs, and 3-SIIs and compare them with estimates provided by all approximators from Table 2. The approximators are run with a wide range of budget values and assessed by their achieved mean squared error (MSE) or precision at five (Precision@5). Figure 5 summarizes the approximation results.

Most notably, the ranking of approximators varies strongly between the different applications domains, which is depicted in Figure 5 (a) and (b). This observation holds for both SVs and SIs. In general,

(a) Approximation quality in different application domains



(b) Overall approximator performance **(c) MIs per size**

Figure 5: Overview of the benchmark results containing **(a)** budget-dependent MSE approximation curves on different benchmark settings, **(b)** a summary of the best performing approximators per setting over all 100 benchmark games measured by MSE **(left)** and Precision@5 **(right)**, and **(c)** exemplary MIs for ten games of Data Valuation **(top)** and Local Explanation **(bottom)**.

two types of approximation methods dominate the application landscape in terms of MSE and Precision@5. First, *kernel-based* approaches including KernelSHAP, k_{ADD}-SHAP, KernelSHAP-IQ and Inconsistent KernelSHAP-IQ perform best for Local Explanation, Uncertainty Explanation, and Unsupervised Feature Importance. Second, the two *stratification-based* estimators SVARM and SVARM-IQ achieve state-of-the-art performance for Data Valuation, Dataset Valuation, or Ensemble Selection. Traditional *mean-estimation* methods including Permutation Sampling (SV and SII), Unbiased KernelSHAP, SHAPIQ, and Owen Sampling achieve moderate estimation qualities in comparison. Our findings give rise to the conclusion that *stratification-based* estimators perform superior in settings where the size of a coalition naturally impacts its worth (e.g. training size for Dataset Valuation), which is plausible as these methods group coalitions by size and thus leverage this dependency. Meanwhile, *kernel-based* estimators achieve state-of-the-art in settings where the dependency between size and worth of a coalition is less pronounced (e.g. sudden jumps of model predictions in Local Explanation).

Interestingly, the settings where *stratified-sampling* outperforms *kernel-based* variants exhibit different internal structures in the games' MIs. Generally, MIs disentangle a game into all of its additive components (cf. Section 2) and can be computed exactly with `shapiq`'s pre-computed games. The accuracy of *kernel-based* estimators drops when higher-order interactions dominate the games' structure instead of lower-order interactions. This is depicted by Figure 5 (c) where the MIs for Local Explanation are of lower order than the Data Valuation games.

5 Conclusion

As SIs are increasingly employed to analyze ML models, it becomes pivotal to ensure that these are accurately and efficiently approximated. To this end, we contributed `shapiq`, an open-source toolbox that implements state-of-the-art algorithms, defines a dozen of benchmark games, and provides ready-to-use explanations of any-order feature interactions. `shapiq` contains a comprehensive documentation and is designed to be extendable by contributors.

Limitations and future work. We identify three main limitations of `shapiq` that provide natural opportunities for future work. First, the TreeSHAP-IQ algorithm is currently implemented in Python, but by-design requires no access to model inference, which allows for a more efficient implementation in C++ alike TreeSHAP [53, 85]. Second, SIs can be misinterpreted based on choosing the wrong index for the application scenario, which we comment on across Sections 2 and 4.1. The selection of a particular SI index, enabled by `shapiq`, offers great opportunities for application-specific research. We also acknowledge that visualization of higher-order feature interactions is itself challenging and a potential research direction in human-computer interaction. Certainly, a human-centric evaluation of explanations may be required for their broader adoption in practical applications [70].

Broader impact. A potential negative societal implication of visualizing higher-order feature interactions may be an *information overload* [7, 69] that leads to users misinterpreting model explanations. Nevertheless, we hope our contribution sparks the advancement of game-theoretical indices motivated by various applications in ML. Specifically in the context of explainability, `shapiq` may impact the way users interact with ML models when having access to previously inaccessible information, e.g. higher-order feature interactions.

Acknowledgments and Disclosure of Funding

We gratefully thank the anonymous reviewers for their valuable feedback for improving this work. We also gratefully thank Santo Thies for supporting the implementation. Fabian Fumagalli and Maximilian Muschalik gratefully acknowledge funding by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation): TRR 318/1 2021 – 438445824. Hubert Baniecki was supported from the state budget within the Polish Ministry of Education and Science program “Pearls of Science” project number PN/01/0087/2022. Patrick Kolpaczki was supported by the research training group Dataninja (Trustworthy AI for Seamless Problem Solving: Next Generation Intelligence Joins Robust Data Analysis) funded by the German federal state of North Rhine-Westphalia.

References

[1] Aas, K., Jullum, M., and Løland, A. (2021). Explaining individual predictions when features are dependent: More accurate approximations to Shapley values. *Artificial Intelligence*, 298:103502.

[2] Agarwal, C., Krishna, S., Saxena, E., Pawelczyk, M., Johnson, N., Puri, I., Zitnik, M., and Lakkaraju, H. (2022). OpenXAI: Towards a transparent evaluation of model explanations. In *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*.

[3] Alber, M., Lapuschkin, S., Seegerer, P., Hägele, M., Schütt, K. T., Montavon, G., Samek, W., Müller, K.-R., Dähne, S., and Kindermans, P.-J. (2019). iNNvestigate neural networks! *Journal of Machine Learning Research*, 20(93):1–8.

[4] Arya, V., Bellamy, R. K. E., Chen, P.-Y., Dhurandhar, A., Hind, M., Hoffman, S. C., Houde, S., Liao, Q. V., Luss, R., MojsiloviÄ‡, A., Mourad, S., Pedemonte, P., Raghavendra, R., Richards, J. T., Sattigeri, P., Shanmugam, K., Singh, M., Varshney, K. R., Wei, D., and Zhang, Y. (2020). AI Explainability 360: An extensible toolkit for understanding data and machine learning models. *Journal of Machine Learning Research*, 21(130):1–6.

[5] Balestra, C., Huber, F., Mayr, A., and Müller, E. (2022). Unsupervised features ranking via coalitional game theory for categorical data. In *Proceedings of Big Data Analytics and Knowledge Discovery (DaWaK)*, pages 97–111.

[6] Baniecki, H., Kretowicz, W., Piatyszek, P., Wisniewski, J., and Biecek, P. (2021). dalex: Responsible machine learning with interactive explainability and fairness in Python. *Journal of Machine Learning Research*, 22(214):1–7.

[7] Baniecki, H., Parzych, D., and Biecek, P. (2024). The grammar of interactive explanatory model analysis. *Data Mining and Knowledge Discovery*, 38:2596–2632.

[8] Banzhaf III, J. F. (1964). Weighted voting doesn't work: A mathematical analysis. *Rutgers Law Review*, 19:317.

[9] Bischl, B., Casalicchio, G., Feurer, M., Gijsbers, P., Hutter, F., Lang, M., Mantovani, R. G., van Rijn, J. N., and Vanschoren, J. (2021). OpenML benchmarking suites. In *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*.

[10] Bordt, S. and von Luxburg, U. (2023). From Shapley Values to Generalized Additive Models and back. In *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 709–745.

[11] Castro, J., Gómez, D., and Tejada, J. (2009). Polynomial calculation of the Shapley value based on sampling. *Computers & Operations Research*, 36(5):1726–1730.

[12] Charnes, A., Golany, B., Keane, M., and Rousseau, J. (1988). *Extremal Principle Solutions of Games in Characteristic Function Form: Core, Chebychev and Shapley Value Generalizations*, volume 11, page 123–133. Springer Netherlands.

[13] Chen, H., Covert, I., Lundberg, S., et al. (2023). Algorithms to estimate Shapley value feature attributions. *Nature Machine Intelligence*, 5:590–601.

[14] Chen, H., Janizek, J. D., Lundberg, S. M., and Lee, S. (2020). True to the Model or True to the Data? *CoRR*, 2006.16234.

[15] Chen, T. and Guestrin, C. (2016). XGBoost: A scalable tree boosting system. In *Proceedings of International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 785–794.

[16] Cohen, S. B., Dror, G., and Ruppin, E. (2007). Feature selection via coalitional game theory. *Neural Computing*, 19(7):1939–1961.

[17] Covert, I. and Lee, S. (2021). Improving KernelSHAP: Practical Shapley Value Estimation Using Linear Regression. In *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 3457–3465.

[18] Covert, I., Lundberg, S. M., and Lee, S. (2020). Understanding global feature contributions with additive importance measures. In *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*.

[19] Covert, I., Lundberg, S. M., and Lee, S. (2021). Explaining by Removing: A Unified Framework for Model Explanation. *Journal of Machine Learning Research*, 22(209):1–90.

[20] Deng, H., Ren, Q., Zhang, H., and Zhang, Q. (2022). Discovering and explaining the representation bottleneck of DNNS. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

[21] Deng, J., Dong, W., Socher, R., Li, L., Li, K., and Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 248–255.

[22] Deng, X. and Papadimitriou, C. H. (1994). On the complexity of cooperative solution concepts. *Mathematics of Operations Research*, 19(2):257–266.

[23] Dubey, P., Neyman, A., and Weber, R. J. (1981). Value theory without efficiency. *Mathematics of Operations Research*, 6(1):122–128.

[24] Fanaee-T, H. and Gama, J. (2014). Event Labeling Combining Ensemble Detectors and Background Knowledge. *Progress in Artificial Intelligence*, 2(2):113–127.

[25] Feurer, M., van Rijn, J. N., Kadra, A., Gijsbers, P., Mallik, N., Ravi, S., Mueller, A., Vanschoren, J., and Hutter, F. (2020). OpenML-Python: an extensible Python API for OpenML. *CoRR*, abs/1911.02490.

[26] Frye, C., de Mijolla, D., Begley, T., Cowton, L., Stanley, M., and Feige, I. (2021). Shapley explainability on the data manifold. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

[27] Fujimoto, K., Kojadinovic, I., and Marichal, J. (2006). Axiomatic characterizations of probabilistic and cardinal-probabilistic interaction indices. *Games and Economic Behavior*, 55(1):72–99.

[28] Fumagalli, F., Muschalik, M., Kolpaczki, P., Hüllermeier, E., and Hammer, B. (2024). Kernelshap-iq: Weighted least square optimization for shapley interactions. In *Forty-first International Conference on Machine Learning, (ICML 2024)*.

[29] Fumagalli, F., Muschalik, M., Kolpaczki, P., Hüllermeier, E., and Hammer, B. E. (2023). SHAP-IQ: Unified approximation of any-order shapley interactions. In *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*.

[30] Ghorbani, A. and Zou, J. Y. (2019). Data shapley: Equitable valuation of data for machine learning. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 2242–2251.

[31] Grabisch, M. (2016). *Set Functions, Games and Capacities in Decision Making*, volume 46. Springer International Publishing Switzerland.

[32] Grabisch, M., Marichal, J., and Roubens, M. (2000). Equivalent representations of set functions. *Mathematics of Operations Research*, 25(2):157–178.

[33] Grabisch, M. and Roubens, M. (1999). An axiomatic approach to the concept of interaction among players in cooperative games. *International Journal of Game Theory*, 28(4):547–565.

[34] Harris, C., Pymar, R., and Rowat, C. (2022). Joint shapley values: a measure of joint feature importance. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

[35] Harsanyi, J. C. (1963). A simplified bargaining model for the n-person cooperative game. *International Economic Review*, 4(2):194–220.

[36] Hedstrom, A., Weber, L., Krakowczyk, D., Bareeva, D., Motzkus, F., Samek, W., Lapuschkin, S., and Hohne, M. M.-C. (2023). Quantus: An explainable AI toolkit for responsible evaluation of neural network explanations and beyond. *Journal of Machine Learning Research*, 24(34):1–11.

[37] Inglis, A., Parnell, A., and Hurley, C. B. (2022). Visualizing Variable Importance and Variable Interaction Effects in Machine Learning Models. *Journal of Computational and Graphical Statistics*, 31(3):766–778.

[38] Jiang, K. F., Liang, W., Zou, J., and Kwon, Y. (2023). OpenDataVal: a unified benchmark for data valuation. In *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*.

[39] Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., and Liu, T.-Y. (2017). LightGBM: A highly efficient gradient boosting decision tree. In *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*, pages 3149–3157.

[40] Kelley Pace, R. and Barry, R. (1997). Sparse spatial autoregressions. *Statistics & Probability Letters*, 33(3):291–297.

[41] Klaise, J., Looveren, A. V., Vacanti, G., and Coca, A. (2021). Alibi explain: Algorithms for explaining machine learning models. *Journal of Machine Learning Research*, 22(181):1–7.

[42] Kohavi, R. (1996). Scaling up the accuracy of naive-bayes classifiers: A decision-tree hybrid. In *Proceedings of International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 202–207.

[43] Kokhlikyan, N., Miglani, V., Martin, M., Wang, E., Alsallakh, B., Reynolds, J., Melnikov, A., Kliushkina, N., Araya, C., Yan, S., and Reblitz-Richardson, O. (2020). Captum: A unified and generic model interpretability library for PyTorch. *arXiv preprint arXiv:2009.07896*.

[44] Kolpaczki, P., Bengs, V., Muschalik, M., and Hüllermeier, E. (2024a). Approximating the shapley value without marginal contributions. In *Proceeedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pages 13246–13255.

[45] Kolpaczki, P., Haselbeck, G., and Hüllermeier, E. (2024b). How much can stratification improve the approximation of shapley values? In *Proceeedings of World Conference on Explainable Artifical Intelligence (xAI)*, pages 489–512.

[46] Kolpaczki, P., Muschalik, M., Fumagalli, F., Hammer, B., and Hüllermeier, E. (2024c). SVARM-IQ: efficient approximation of any-order shapley interactions through stratification. In *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 3520–3528.

[47] Kumar, I., Scheidegger, C., Venkatasubramanian, S., and Friedler, S. A. (2021). Shapley Residuals: Quantifying the limits of the Shapley value for explanations. In *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*, pages 26598–26608.

[48] Kumar, I. E., Venkatasubramanian, S., Scheidegger, C., and Friedler, S. A. (2020). Problems with Shapley-value-based explanations as feature importance measures. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 5491–5500.

[49] Li, W. and Yu, Y. (2024). Faster approximation of probabilistic and distributional values via least squares. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

[50] Li, X., Du, M., Chen, J., Chai, Y., Lakkaraju, H., and Xiong, H. (2023). $\mathcal{M}^4$: A unified XAI benchmark for faithfulness evaluation of feature attribution methods across metrics, modalities and models. In *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*.

[51] Lin, C., Covert, I., and Lee, S.-I. (2023). On the robustness of removal-based feature attributions. In *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*.

[52] Liu, Y., Khandagale, S., White, C., and Neiswanger, W. (2021). Synthetic benchmarks for scientific research in explainable machine learning. In *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*.

[53] Lundberg, S. M., Erion, G. G., Chen, H., DeGrave, A. J., Prutkin, J. M., Nair, B., Katz, R., Himmelfarb, J., Bansal, N., and Lee, S. (2020). From local explanations to global understanding with explainable AI for trees. *Nature Machine Intelligence*, 2(1):56–67.

[54] Lundberg, S. M. and Lee, S. (2017). A Unified Approach to Interpreting Model Predictions. In *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*, pages 4765–4774.

[55] Maas, A. L., Daly, R. E., Pham, P. T., Huang, D., Ng, A. Y., and Potts, C. (2011). Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (HLT)*, pages 142–150.

[56] Maleki, S., Tran-Thanh, L., Hines, G., Rahwan, T., and Rogers, A. (2013). Bounding the estimation error of sampling-based shapley value approximation with/without stratifying. *CoRR*, abs/1306.4265.

[57] Marichal, J., Kojadinovic, I., and Fujimoto, K. (2007). Axiomatic characterizations of generalized values. *Discrete Applied Mathematics*, 155(1):26–43.

[58] Marichal, J.-L. (2000). The influence of variables on pseudo-boolean functions with applications to game theory and multicriteria decision making. *Discrete Applied Mathematics*, 107:139–164.

[59] Marichal, J.-L. and Roubens, M. (1999). *The Chaining Interaction Index among Players in Cooperative Games*, page 69–85. Springer Netherlands.

[60] Muschalik, M., Fumagalli, F., Hammer, B., and Hüllermeier, E. (2024). Beyond treeshap: Efficient computation of any-order shapley interactions for tree ensembles. In *Proceeedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pages 14388–14396.

[61] Olsen, L. H. B., Glad, I. K., Jullum, M., and Aas, K. (2022). Using Shapley values and variational autoencoders to explain predictive models with dependent mixed features. *Journal of Machine Learning Research*, 23(213):1–51.

[62] Olsen, L. H. B., Glad, I. K., Jullum, M., and Aas, K. (2024). A comparative study of methods for estimating model-agnostic Shapley value explanations. *Data Mining and Knowledge Discovery*, pages 1–48.

[63] Owen, A. B. (2014). Sobol' indices and shapley value. *SIAM/ASA Journal for Uncertainty Quantification*, 2(1):245–251.

[64] Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. (2019). Pytorch: An imperative style, high-performance deep learning library. In *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*, pages 8024–8035.

[65] Patel, N., Strobel, M., and Zick, Y. (2021). High dimensional model explanations: An axiomatic approach. In *Proceedings of ACM Conference on Fairness, Accountability, and Transparency, Virtual Event (FAccT)*, pages 401–411.

[66] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., VanderPlas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12:2825–2830.

[67] Pelegrina, G. D., Duarte, L. T., and Grabisch, M. (2023). A k-additive choquet integral-based approach to approximate the SHAP values for local interpretability in machine learning. *Artificial Intelligence*, 325:104014.

[68] Pfannschmidt, K., Hüllermeier, E., Held, S., and Neiger, R. (2016). Evaluating tests in medical diagnosis: Combining machine learning with game-theoretical concepts. In *Proceedings of Information Processing and Management of Uncertainty in Knowledge-Based Systems (IPMU)*, pages 450–461.

[69] Poursabzi-Sangdeh, F., Goldstein, D. G., Hofman, J. M., Vaughan, J. W., and Wallach, H. (2021). Manipulating and measuring model interpretability. In *Proceedings of CHI Conference on Human Factors in Computing Systems (CHI)*.

[70] Rong, Y., Leemann, T., Nguyen, T.-T., Fiedler, L., Qian, P., Unhelkar, V., Seidel, T., Kasneci, G., and Kasneci, E. (2024). Towards human-centered explainable AI: A survey of user studies for model explanations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(4):2104–2122.

[71] Rota, G.-C. (1964). On the foundations of combinatorial theory: I. theory of möbius functions. In *Classic Papers in Combinatorics*, pages 332–360. Springer.

[72] Rozemberczki, B. and Sarkar, R. (2021). The shapley value of classifiers in ensemble games. In *Proceedings of the International Conference on Information and Knowledge Management (CIKM)*, pages 1558–1567.

[73] Rozemberczki, B., Watson, L., Bayer, P., Yang, H., Kiss, O., Nilsson, S., and Sarkar, R. (2022). The shapley value in machine learning. In *Proceedings of International Joint Conference on Artificial Intelligence (IJCAI)*, pages 5572–5579.

[74] Shapley, L. S. (1953). A Value for n-Person Games. In *Contributions to the Theory of Games (AM-28), Volume II*, pages 307–318. Princeton University Press.

[75] Strumbelj, E. and Kononenko, I. (2014). Explaining prediction models and individual predictions with feature contributions. *Knowledge and Information Systems*, 41(3):647–665.

[76] Sundararajan, M., Dhamdhere, K., and Agarwal, A. (2020). The Shapley Taylor Interaction Index. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 9259–9268.

[77] Sundararajan, M. and Najmi, A. (2020). The Many Shapley Values for Model Explanation. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 9269–9278.

[78] Tay, S. S., Xu, X., Foo, C. S., and Low, B. K. H. (2022). Incentivizing collaboration in machine learning via synthetic data rewards. In *Proceeedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pages 9448–9456.

[79] Tsai, C., Yeh, C., and Ravikumar, P. (2023). Faith-Shap: The Faithful Shapley Interaction Index. *Journal of Machine Learning Research*, 24(94):1–42.

[80] Tsang, M., Rambhatla, S., and Liu, Y. (2020). How does This Interaction Affect Me? Interpretable Attribution for Feature Interactions. In *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*, pages 6147–6159.

[81] van Campen, T., Hamers, H., Husslage, B., and Lindelauf, R. (2018). A new approximation method for the shapley value applied to the WTC 9/11 terrorist attack. *Social Network Analysis and Mining*, 8(1):3:1–3:12.

[82] Watson, D. S., O'Hara, J., Tax, N., Mudd, R., and Guy, I. (2023). Explaining predictive uncertainty with information theoretic shapley values. In *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*.

[83] Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., Davison, J., Shleifer, S., von Platen, P., Ma, C., Jernite, Y., Plu, J., Xu, C., Scao, T. L., Gugger, S., Drame, M., Lhoest, Q., and Rush, A. M. (2020). Transformers: State-of-the-art natural language processing. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing: System Demonstrations (EMNLP)*, pages 38–45.

[84] Wright, M. N., Ziegler, A., and König, I. R. (2016). Do little interactions get lost in dark random forests? *BMC Bioinformatics*, 17:145.

[85] Yu, P., Bifet, A., Read, J., and Xu, C. (2022). Linear tree shap. In *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*.

[86] Zern, A., Broelemann, K., and Kasneci, G. (2023). Interventional SHAP values and interaction values for piecewise linear regression trees. In *Proceeedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pages 11164–11173.

[87] Zhou, Z., Xu, X., Sim, R. H. L., Foo, C. S., and Low, B. K. H. (2023). Probably approximate shapley fairness with applications in machine learning. In *Proceeedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pages 5910–5918.

Organization of the Supplementary Material

The supplementary material is organized as follows:

A Extended Theoretical Background

In this section, we introduce further theoretical background. Specifically, we discuss in more detail the class of GVs and IIs in Appendix A.1, and the MIs in Appendix A.2.

A.1 Probabilistic and Cardinal-Probabilistic GVs and IIs

Probabilistic GVs [57] extend semivalues with a focus on *monotonicity*, i.e. games that satisfy $\nu(S) \leq \nu(T)$, if $S \subseteq T \subseteq N$. GVs satisfy the *positivity axiom*, which requires non-negative joint contributions, i.e. $\phi^{\text{GV}}(S) \geq 0$, for all $S \subseteq N$ in monotone games [57]. It was shown that GVs are uniquely represented as weighted averages over (joint) marginal contributions $\nu(T \cup S) - \nu(T)$. On the other hand, cardinal-probabilistic IIs [27] are centered around synergy, independence and redundancy between entities. IIs are based on discrete derivatives, which extend (joint) marginal contributions by accounting for lower-order effects. IIs focus on k-*monotonicity*, i.e. games that have non-negative discrete derivatives $\Delta_S(T) \geq 0$ for $S \subseteq U \subseteq N$ with $2 \leq |S| \leq k$. IIs satisfy the k-*monotonicity axiom*, i.e. non-negative interactions $\phi^{\text{II}}(S) \geq 0$ for k-monotone games. Both, GVs and IIs are uniquely represented [27, 57] as

$$\phi^{\text{GV}}(S) := \sum_{T \subseteq N \setminus S} p_{|T|}^{|S|}(n) \cdot (\nu(T \cup S) - \nu(T)) \quad \text{and} \quad \phi^{\text{II}}(S) := \sum_{T \subseteq N \setminus S} p_{|T|}^{|S|}(n) \cdot \Delta_S(T),$$

where $p_t^s(n)$ are index-specific weights based on the sizes of S, T and N. The *SGV* [58] and the *SII* [33] with

$$\textbf{Shapley: } p_t^s(n) = \frac{1}{n-s+1}\binom{n-s}{t}^{-1}$$

naturally extend the SV. An alternative extension for the SV is the *Chaining GV (CHGV)* [57] and *Chaining II (CHII)* [59] with

$$\textbf{Chaining: } p_t^s(n) = \frac{s}{s+t}\binom{n}{s+t}^{-1}.$$

The main difference of the SGV/SII and the CHGV/CHII is the quantification of so-called *partnerships* [27, 57], i.e. coalitions that only influence the value of the game if all members of the partnership are present. The CHGV and CHII adhere to the *partnership-allocation axiom* [27, 57], which states that the contribution of an individual member of the partnership and the interaction of the whole partnership are proportional. In contrast, the SGV and SII satisfy the *reduced partnership consistency axiom* [27, 57], which states that the interaction of the whole partnership is equal to the contribution of the partnership in a game, where the partnership is a single player.

On the other hand, the *Banzhaf GV (BGV)* [58] and *Banzhaf II (BII)* [33] extend the BV with

$$\textbf{Banzhaf: } p_t^s(n) := \frac{1}{2^{n-s}}.$$

A.2 Möbius Interactions (MIs)

The MIs Φ_n are a prominent concept in discrete mathematics, which appears in many different forms. In discrete mathematics, it also known as the Möbius transform [71]. In cooperative game theory, the concept is known as Harsanyi dividend [35] or internal II [27]. The MI for $S \subseteq N$ is defined as

$$\Phi_n(S) := \sum_{T \subseteq S} (-1)^{|S|-|T|} \nu(T).$$

In this context, the MIs are the unique measure that satisfy the *recovery property*

$$\nu(T) = \sum_{S \subseteq T} \Phi_n(S) \text{ for every } T \subseteq N.$$

The MIs are a basis of the vector space of cooperative games, and thus every game can be uniquely represented in terms of its MIs. The *Co-Möbius transform (Co-MI)* [32] is another fundamental concept linked to the MIs of the conjugate game, i.e. $\bar{\nu}(T) := \nu(N \setminus T)$ [31].

B Detailed Overview of all Benchmark Games and Configurations

B.1 Benchmark Overview

We list in Table 4 and 5 all configurations available within our benchmark. Depending on the application task, a configuration represents a combination of multiple parameters which specify the generated cooperative games. For ML games such a combination includes at least the used dataset and number of features or datapaoints. If a prediction model or imputer for feature values is employed, as for example for Local XAI games, these are also specified.

Table 4: Overview of all Benchmark Configurations: Each configuration is assigned a distinctive identifier (ID), has a name (Benchmark) indicating dataset and application if available, is pre-computed (P.) if the player number n does not exceed 16, consists of $|\mathcal{P}(N)|$ many coalitions to be evaluated, is iterated over multiple game instances (g), and has a set of parameters (Game Configuration).

ID	Benchmark	Data	P.	n	$\|\mathcal{P}(N)\|$	g	Game Configuration
1	Data Valuation	AD	✓	15	32768	10	model_name=decision_tree, n_data_points=15
2	Data Valuation	AD	✓	15	32768	10	model_name=random_forest, n_data_points=15
3	Data Valuation	AD	✓	10	1024	30	model_name=decision_tree, player_sizes=increasing, n_players=10
4	Data Valuation	AD	✓	10	1024	30	model_name=random_forest, player_sizes=increasing, n_players=10
5	Data Valuation	AD	✓	10	1024	30	model_name=gradient_boosting, player_sizes=increasing, n_players=10
6	Data Valuation	AD	✓	14	16384	5	model_name=decision_tree, player_sizes=increasing, n_players=14
7	Ensemble Selection	AD	✓	10	1024	30	loss_function=accuracy_score, n_members=10
8	Feature Selection	AD	✓	14	16384	30	model_name=decision_tree
9	Feature Selection	AD	✓	14	16384	30	model_name=random_forest
10	Feature Selection	AD	✓	14	16384	30	model_name=gradient_boosting
11	Global Explanation	AD	✓	14	16384	30	model_name=decision_tree, loss_function=accuracy_score
12	Global Explanation	AD	✓	14	16384	30	model_name=random_forest, loss_function=accuracy_score
13	Global Explanation	AD	✓	14	16384	30	model_name=gradient_boosting, loss_function=accuracy_score
14	Local Explanation	AD	✓	14	16384	30	model_name=decision_tree, imputer=marginal
15	Local Explanation	AD	✓	14	16384	30	model_name=random_forest, imputer=marginal
16	Local Explanation	AD	✓	14	16384	30	model_name=gradient_boosting, imputer=marginal
17	Local Explanation	AD	✓	14	16384	30	model_name=decision_tree, imputer=conditional
18	Local Explanation	AD	✓	14	16384	30	model_name=random_forest, imputer=conditional
19	Local Explanation	AD	✓	14	16384	30	model_name=gradient_boosting, imputer=conditional
20	RF Ensemble Selection	AD	✓	10	1024	30	loss_function=accuracy_score, n_members=10
21	Uncertainty Explanation	AD	✓	14	16384	30	uncertainty_to_explain=total, imputer=marginal
22	Uncertainty Explanation	AD	✓	14	16384	30	uncertainty_to_explain=total, imputer=conditional
23	Uncertainty Explanation	AD	✓	14	16384	30	uncertainty_to_explain=aleatoric, imputer=marginal
24	Uncertainty Explanation	AD	✓	14	16384	30	uncertainty_to_explain=aleatoric, imputer=conditional
25	Uncertainty Explanation	AD	✓	14	16384	30	uncertainty_to_explain=epistemic, imputer=marginal
26	Uncertainty Explanation	AD	✓	14	16384	30	uncertainty_to_explain=epistemic, imputer=conditional
27	Unsupervised FI.	AD	✓	14	16384	1	-
28	Cluster Explanation	BS	✓	12	4096	1	cluster_method=kmeans, score_method=silhouette_score
29	Cluster Explanation	BS	✓	12	4096	1	cluster_method=agglomerative, score_method=calinski_harabasz_score
30	Data Valuation	BS	✓	15	32768	10	model_name=decision_tree, n_data_points=15
31	Data Valuation	BS	✓	15	32768	10	model_name=random_forest, n_data_points=15
32	Data Valuation	BS	✓	10	1024	30	model_name=decision_tree, player_sizes=increasing, n_players=10
33	Data Valuation	BS	✓	10	1024	30	model_name=random_forest, player_sizes=increasing, n_players=10
34	Data Valuation	BS	✓	10	1024	30	model_name=gradient_boosting, player_sizes=increasing, n_players=10
35	Data Valuation	BS	✓	14	16384	5	model_name=decision_tree, player_sizes=increasing, n_players=14
36	Ensemble Selection	BS	✓	10	1024	30	loss_function=r2_score, n_members=10
37	Feature Selection	BS	✓	12	4096	30	model_name=decision_tree
38	Feature Selection	BS	✓	12	4096	30	model_name=random_forest
39	Feature Selection	BS	✓	12	4096	30	model_name=gradient_boosting
40	Global Explanation	BS	✓	12	4096	30	model_name=decision_tree, loss_function=r2_score
41	Global Explanation	BS	✓	12	4096	30	model_name=random_forest, loss_function=r2_score
42	Global Explanation	BS	✓	12	4096	30	model_name=gradient_boosting, loss_function=r2_score
43	Local Explanation	BS	✓	12	4096	30	model_name=decision_tree, imputer=marginal
44	Local Explanation	BS	✓	12	4096	30	model_name=random_forest, imputer=marginal
45	Local Explanation	BS	✓	12	4096	30	model_name=gradient_boosting, imputer=marginal
46	Local Explanation	BS	✓	12	4096	30	model_name=decision_tree, imputer=conditional
47	Local Explanation	BS	✓	12	4096	30	model_name=random_forest, imputer=conditional
48	Local Explanation	BS	✓	12	4096	30	model_name=gradient_boosting, imputer=conditional
49	RF Ensemble Selection	BS	✓	10	1024	30	loss_function=r2_score, n_members=10
50	Unsupervised FI.	BS	✓	12	4096	1	–

Table 5: Overview of all Benchmark Configurations: Each configuration is assigned a distinctive identifier (ID), has a name (Benchmark) indicating dataset and application if available, is pre-computed (P.) if the player number n does not exceed 16, consists of $|\mathcal{P}(N)|$ many coalitions to be evaluated, is iterated over multiple game instances (g), and has a set of parameters (Game Configuration).

ID	Benchmark	Data	P.	n	$\|\mathcal{P}(N)\|$	g	Game Configuration
51	Cluster Explanation	CH	✓	8	256	1	cluster_method=kmeans, score_method=silhouette_score
52	Cluster Explanation	CH	✓	8	256	1	cluster_method=agglomerative, score_method=calinski_harabasz_score
53	Data Valuation	CH	✓	15	32768	10	model_name=decision_tree, n_data_points=15
54	Data Valuation	CH	✓	15	32768	10	model_name=random_forest, n_data_points=15
55	Data Valuation	CH	✓	10	1024	30	model_name=decision_tree, player_sizes=increasing, n_players=10
56	Data Valuation	CH	✓	10	1024	30	model_name=random_forest, player_sizes=increasing, n_players=10
57	Data Valuation	CH	✓	10	1024	30	model_name=gradient_boosting, player_sizes=increasing, n_players=10
58	Data Valuation	CH	✓	14	16384	5	model_name=decision_tree, player_sizes=increasing, n_players=14
59	Ensemble Selection	CH	✓	10	1024	30	loss_function=r2_score, n_members=10
60	Feature Selection	CH	✓	8	256	30	model_name=decision_tree
61	Feature Selection	CH	✓	8	256	30	model_name=random_forest
62	Feature Selection	CH	✓	8	256	30	model_name=gradient_boosting
63	Global Explanation	CH	✓	8	256	30	model_name=decision_tree, loss_function=r2_score
64	Global Explanation	CH	✓	8	256	30	model_name=random_forest, loss_function=r2_score
65	Global Explanation	CH	✓	8	256	30	model_name=gradient_boosting, loss_function=r2_score
66	Global Explanation	CH	✓	8	256	30	model_name=neural_network, loss_function=r2_score
67	Local Explanation	CH	✓	8	256	30	model_name=decision_tree, imputer=marginal
68	Local Explanation	CH	✓	8	256	30	model_name=random_forest, imputer=marginal
69	Local Explanation	CH	✓	8	256	30	model_name=gradient_boosting, imputer=marginal
70	Local Explanation	CH	✓	8	256	30	model_name=neural_network, imputer=marginal
71	Local Explanation	CH	✓	8	256	30	model_name=decision_tree, imputer=conditional
72	Local Explanation	CH	✓	8	256	30	model_name=random_forest, imputer=conditional
73	Local Explanation	CH	✓	8	256	30	model_name=gradient_boosting, imputer=conditional
74	Local Explanation	CH	✓	8	256	30	model_name=neural_network, imputer=conditional
75	RF Ensemble Selection	CH	✓	10	1024	30	loss_function=r2_score, n_members=10
76	Unsupervised FI.	CH	✓	8	256	1	–
77	Local Explanation	IC	✓	14	16384	30	model_name=resnet_18, n_superpixel_resnet=14
78	Local Explanation	IC	✓	9	512	30	model_name=vit_9_patches
79	Local Explanation	IC	✓	16	65536	30	model_name=vit_16_patches
80	Sum of Unanimity Model	Syn	✓	15	32768	10	n=15, n_basis_games=30, min_interaction_size=1, max_interaction_size=5
81	Sum of Unanimity Model	Syn	✓	15	32768	10	n=15, n_basis_games=30, min_interaction_size=1, max_interaction_size=15
82	Sum of Unanimity Model	Syn	✓	15	32768	10	n=15, n_basis_games=150, min_interaction_size=1, max_interaction_size=5
83	Sum of Unanimity Model	Syn	✓	15	32768	10	n=15, n_basis_games=150, min_interaction_size=1, max_interaction_size=15
84	Sum of Unanimity Model	Syn	**X**	30	$> 2^{16}$	10	n=30, n_basis_games=30, min_interaction_size=1, max_interaction_size=5
85	Sum of Unanimity Model	Syn	**X**	30	$> 2^{16}$	10	n=30, n_basis_games=30, min_interaction_size=1, max_interaction_size=15
86	Sum of Unanimity Model	Syn	**X**	30	$> 2^{16}$	10	n=30, n_basis_games=30, min_interaction_size=1, max_interaction_size=25
87	Sum of Unanimity Model	Syn	**X**	30	$> 2^{16}$	10	n=30, n_basis_games=150, min_interaction_size=1, max_interaction_size=5
88	Sum of Unanimity Model	Syn	**X**	30	$> 2^{16}$	10	n=30, n_basis_games=150, min_interaction_size=1, max_interaction_size=15
89	Sum of Unanimity Model	Syn	**X**	30	$> 2^{16}$	10	n=30, n_basis_games=150, min_interaction_size=1, max_interaction_size=25
90	Sum of Unanimity Model	Syn	**X**	50	$> 2^{16}$	10	n=50, n_basis_games=30, min_interaction_size=1, max_interaction_size=5
91	Sum of Unanimity Model	Syn	**X**	50	$> 2^{16}$	10	n=50, n_basis_games=30, min_interaction_size=1, max_interaction_size=15
92	Sum of Unanimity Model	Syn	**X**	50	$> 2^{16}$	10	n=50, n_basis_games=30, min_interaction_size=1, max_interaction_size=25
93	Sum of Unanimity Model	Syn	**X**	50	$> 2^{16}$	10	n=50, n_basis_games=150, min_interaction_size=1, max_interaction_size=5
94	Sum of Unanimity Model	Syn	**X**	50	$> 2^{16}$	10	n=50, n_basis_games=150, min_interaction_size=1, max_interaction_size=15
95	Sum of Unanimity Model	Syn	**X**	50	$> 2^{16}$	10	n=50, n_basis_games=150, min_interaction_size=1, max_interaction_size=25
96	Local Explanation	MR	✓	14	16384	30	mask_strategy=mask
97	Tree Explanation	Syn	**X**	30	$> 2^{16}$	10	model_name=decision_tree, classification=True, n_features=30
98	Tree Explanation	Syn	**X**	30	$> 2^{16}$	10	model_name=random_forest, classification=True, n_features=30
99	Tree Explanation	Syn	**X**	30	$> 2^{16}$	10	model_name=decision_tree, classification=False, n_features=30
100	Tree Explanation	Syn	**X**	30	$> 2^{16}$	10	model_name=random_forest, classification=False, n_features=30

B.2 Datasets and Models Used

Our benchmark games are based on five datasets. All of these datasets are publicly available. The following contains a small description of all datasets:

- **AC:** The *AdultCensus* [42, CC BY 4.0 license] dataset is a tabular classification dataset containing $n = 14$ features. The dataset was obtained from `openml` [25] (id: *1590*) and is available at `https://github.com/mmschlk/shapiq/blob/v1/data/adult_census.csv` for reproducibility.
- **BS:** The *BikeSharing* [24, CC BY 4.0 license] dataset is a tabular regression dataset containing $n = 12$ features. The dataset was obtained from `openml` [25] (id: *42712*) and is available at `https://github.com/mmschlk/shapiq/blob/v1/data/bike.csv` for reproducibility.
- **CH:** The *CaliforniaHousing* [40, CC0 public domain] dataset is a tabular regression dataset containing $n = 8$ features. The target of this dataset is to predict property prices. The dataset was obtained from `scikit-learn` [66] and is available at `https://github.com/mmschlk/shapiq/blob/v1/data/california_housing.csv` for reproducibility.
- **MR:** The *MovieReview* is also known as the IMBD dataset [55, custom research license] contains moview review excerpts. We simplifiy the dataset to only contain sentences parts containing $n \leq 14$ words. The simplified dataset can be found at `https://github.com/mmschlk/shapiq/blob/v1/benchmark/data/simplified_imdb.csv` for reproducibility.
- **IC:** The *ImageClassification* data contains test images from *Imagenet* [21, custom research license]. The example images can be found at `https://github.com/mmschlk/shapiq/tree/v1/shapiq/games/benchmark/imagenet_examples` for reproducibility.

All models used for the benchmark games are defined in the code repository. We use decision tree, random forest, k-nearest neighbour, linear/logistic regression models from `scikit-learn` [66]. Moreover, we use gradient-boosted tree classifiers and regressors from `xgboost` [15]. We train small neural networks with `PyTorch` [64] and use `PyTorch`'s `ResNet18` architecture. The movie review language model and the vision transformer is derived from the `transformers` API [83].

B.3 Benchmarking Approximators of SIs with `shapiq`

Listing 1 shows an API for benchmarking 4 approximation algorithms on an Dataset Valuation game based on the *AdultCensus* dataset and a gradient boosting decision tree model.[2]

```
import shapiq
from shapiq.games.benchmark.benchmark_config import (
    load_games_from_configuration,
    print_benchmark_configurations
)
from shapiq.games.benchmark.plot import plot_approximation_quality, add_legend
# print all available games and benchmark configurations
print_benchmark_configurations()
>> Game: AdultCensusDatasetValuation
>> Player ID: 0
>> Number of Players: 10
>> Number of configurations: 3
>> Is the Benchmark Pre-computed: True
>> Iteration Parameter: random_state
>> Configurations:
>> Configuration 1: {'model_name': 'decision_tree', 'player_sizes': 'increasing', 'n_players': 10}
>> Configuration 2: {'model_name': 'random_forest', 'player_sizes': 'increasing', 'n_players': 10}
>> Configuration 3: {'model_name': 'gradient_boosting', 'player_sizes': 'increasing', 'n_players': 10}
>> ...
# load the game files from disk / or download
games = load_games_from_configuration(game_class="AdultCensusDataValuation", n_player_id=0, config_id=2)
games = list(games)  # convert to list (the generator is consumed)
n_players = games[0].n_players
# define the approximators to benchmark
sv_approximators = [
    shapiq.PermutationSamplingSII(n=n_players, index="k-SII", random_state=0),
    shapiq.SHAPIQ(n=n_players, random_state=0),
    shapiq.SVARMIQ(n=n_players, random_state=0),
    shapiq.KernelSHAPIQ(n=n_players, random_state=0)
]
# run the benchmark with the chosen parameters
results = shapiq.games.benchmark.run_benchmark(
    index="k-SII",
    order=2,
    games=games,
    approximators=sv_approximators,
    save_path="benchmark_results.json",
    budget_steps=[500, 1000, 2000, 4000],
    n_jobs=8
)
# plot the results
plot_approximation_quality(results)
```

Listing 1: Exemplary code for benchmarking approximators with `shapiq`.

[2]For details, refer to the notebook example at https://shapiq.readthedocs.io/en/latest/notebooks/benchmark_approximators.html.

C Marginal and Conditional Imputers

When computing SVs and SIs, especially for structured tabular data that has a natural interpretation of feature distribution, there is a choice for marginalizing feature influence over either a marginal or a conditional distribution [1, 14, 18, 51, 53, 54, 61, 62, 77].

For a concrete example [51], consider a supervised learning task where a model $f : \mathcal{X} \to \mathbb{R}$ is used to predict the response variable given an input $\mathbf{x}$, which consists of individual features $(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n)$. Let $p(\mathbf{x})$ to represent the data distribution with support on $\mathcal{X} \subseteq \mathbb{R}^n$. We use bold symbols $\mathbf{x}$ to denote random variables and normal symbols x to denote values. Let $\mathbf{x}_S$ and x_S denote a subset of features, i.e. players in a game, and values for different $S \subseteq N$, respectively. Then, a cooperative game $\nu : \mathcal{P}(N) \to \mathbb{R}$ for estimating Shapley-based feature attributions and interactions is defined as

$$\nu(S) \coloneqq f_S(x_S) \coloneqq \mathbb{E}_{q(\mathbf{x}_{\bar{S}})}\left[f(x_S, \mathbf{x}_{\bar{S}})\right] = \int f(x_S, x_{\bar{S}}) q(x_{\bar{S}}) dx_{\bar{S}},$$

where $\bar{S} = N \setminus S$ denotes the set complement. The feature distribution $q(\mathbf{x}_{\bar{S}})$ most often considered in the literature is either a *marginal distribution* when $q(\mathbf{x}_{\bar{S}}) \coloneqq p(\mathbf{x}_{\bar{S}})$ [18, 54], or a *conditional distribution* when $q(\mathbf{x}_{\bar{S}}) \coloneqq p(\mathbf{x}_{\bar{S}} \mid x_S)$ [1, 26, 61].

In general, empirical estimation of a conditional feature distribution is challenging [1, 18, 54, 61]. Most recently in [62], the authors benchmark several methods for approximating SVs based on marginalizing features with a conditional distribution $p(\mathbf{x}_{\bar{S}} \mid x_S)$, without a clear best, i.e. different methods are appropriate in different practical situations. Thus, we combine the decision tree-based and sampling approaches [62] to implement a baseline *conditional imputer* in `shapiq.ConditionalImputer`. The class can be easily extended to include more algorithms, which we leave as future work. The rather standard imputation with a marginal distribution $p(\mathbf{x}_{\bar{S}})$ is implemented in `shapiq.MarginalImputer`. Both imputers are used by the appropriate game benchmarks, and available for approximating feature interaction explanations in `shapiq.TabularExplainer` via the `imputer` parameter.

Listing 2 shows a more advanced API for setting a specific imputer and approximator in `shapiq`.[3]

```python
X, model = ...
import shapiq
# create an imputer object
imputer = shapiq.ConditionalImputer(model=model, data=X, sample_size=100)
# create an approximator object
approximator = shapiq.KernelSHAPIQ(n=X.shape[1], index="SII", max_order=3)
# create an explainer object
explainer = shapiq.Explainer(model, X, imputer=imputer, approximator=approximator)
# choose a sample point to be explained
x = X[0]
# approximate feature interactions given the specificed budget
interaction_values = explainer.explain(x=x, budget=1024)
# retrieve 3-order feature interactions
interaction_values.get_n_order_values(3)
# visualize 1-order and 2-order feature interactions on a graph
interaction_values.plot_network(feature_names=...)
```

Listing 2: Exemplary code for defining an imputer and approximator for explanation with `shapiq`.

[3]For details, refer to the notebook example at https://shapiq.readthedocs.io/en/latest/notebooks/conditional_imputer.html.

D Guide for Interpreting Shapley Interaction Visualizations

This section provides information for interpreting visual representations of SIs, offering insights into how interactions between players—such as features in XAI or individual observations for data valuation—are depicted in network and SI graph plots. We propose two types of visualizations: the **network plot** [37, 60] and the **SI graph plot**, with the latter generalizing the former.

General description. In both visualizations, players are represented as nodes, while explanations in the form of interactions are depicted as edges linking these nodes. The network plot is limited to second-order interactions, meaning it only displays edges between two nodes, whereas the SI graph plot accommodates interactions of any order, with interactions involving more than two players represented as hyper-edges connecting three or more nodes. Single-order interactions are represented by the size of the nodes, with larger nodes indicating stronger main effects. The strength and direction of these interactions are encoded through the color, transparency, and thickness of the edges; stronger interactions are shown as thicker and more opaque edges, while weaker interactions are represented by thinner and more transparent edges. Consistent with established conventions from `shap` [54] visualizations, red indicates positive interactions and blue indicates negative interactions. In both visualizations, nodes are drawn in a circular layout by default, but can be positioned also based on a predefined graphical structure. The network plot originates in [37] to illustrate second-order interactions for global effects. It was further adapted for local SIs in [60], whereas the SI graph plot extends this concept by allowing for the visualization of higher-order interactions, thus providing a more comprehensive view of the cooperative game structure.

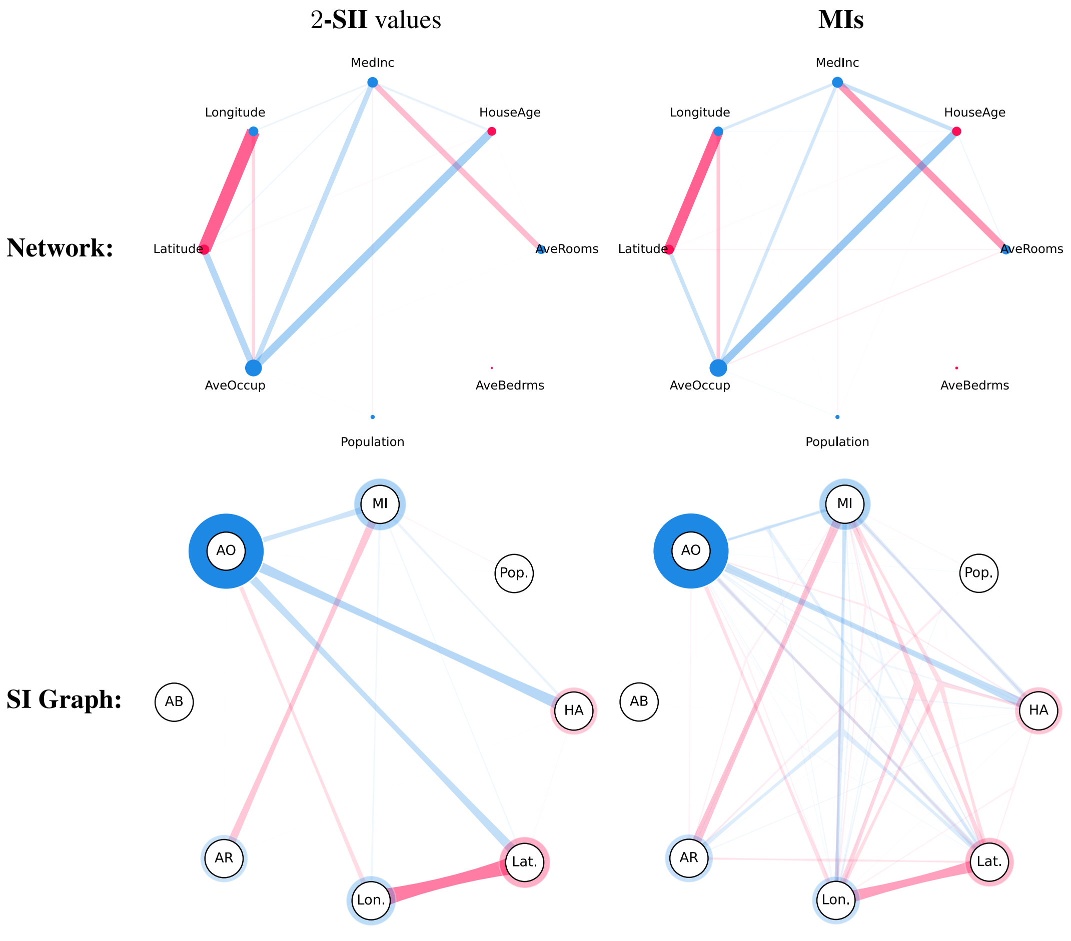


Figure 6: Network plots (top row) and SI graph plots (bottom row) for 2-SII scores (left) and MIs (right) as explanations for an observation from the *CaliforniaHousing* dataset and a random forest.

Interpretation of the example visualizations. Figure 6 shows example network and SI graph plots. We divide the *CaliforniaHousing* dataset into train and test splits. A `RandomForestRegressor` from `scikit-learn` [66] is fitted to the training data, achieving an R^2 score of 80%. We select an observation from the test set and compute 2-SII scores and MIs using `shapiq`'s implementation of TreeSHAP-IQ [60]. For further details regarding the dataset, training, and comparison with other visualizations, we refer to the accompanying notebook on "Visualizing Shapley Interactions" in `shapiq`'s documentation. The observation to be explained has a ground truth property value of 1.575 (in $100,000$ USD) and is predicted to be worth 1.62 (in $100,000$ USD). The baseline prediction of the model is around 2.071 (in $100,000$ USD). This means that with all features provided, the model predicts the property to be worth less than the average house in the dataset. From both the network plots and SI graph plots in Figure 6, it is clear that the *AveOccup* (AO) feature has a strong negative influence on the prediction compared to the baseline, as indicated by the large blue node. However, some features and interactions have positive effects (red edges). Specifically, the interaction between *Latitude* (Lat.) and *Longitude* (Lon.), which encodes the exact location of the property, has a positive influence on the property's valuation. The MI graph plot further reveals that higher-order interactions exist, as shown by hyperedges connecting more than two features. For example, there is a sizable positive third-order interaction between *Longitude* (Lon.), *Latitude* (Lat.), and *MedianIncome* (MI). A positive fourth-order interaction involving the same three features and the *HouseAge* (HA) feature also exists.

E Details of the Experimental Setting and Reproducibility

This section contains additional information regarding the experimental setup and definition of the cooperative games. For further information we refer to the benchmark configuration as part of `shapiq.benchmark`.

E.1 Generated Cooperative Games

The game-theoretical quantification of interaction demands a formal cooperative game specified by a player set N and value function $\nu : \mathcal{P}(N) \to \mathbb{R}$. The players for each benchmark game are already given in Table 3, leaving the value functions left open to be specified, with what we catch up here.

Local Explanation. For a specified datapoint x, the worth of a coalition of features S is given by the model's predicted value $h(x_S)$ using only the features in S. The features outside of S are made absent in x_S by imputing them with a surrogate value in order to remove their information. For tabular datasets such as *AdultCensus*, *BikeSharing*, and *CaliforniaHousing* this is done by marginal or conditional imputation. For the language model predicting the sentiment of movie review excerpts, missing words are set to the masked token. Missing pixels (patches) for the vision transformer image classifier are also set to the masked token. For the ResNet image classifier removed superpixels are collectively set to a mean value (gray).

Global Explanation. Instead of specifying a single datapoint and considering the model's output, the model's loss is averaged over a number of fixed datapoints $x_1, \ldots, x_M$. The model's loss for a coalition S and datapoint x_m is computed by comparing its prediction $h(x_{mS})$ with the ground truth target value. The imputation of absent features is done as for local explanations.

Tree Explanation. This is a specialization of local explanations for tree models, made feasible by the capabilities of TreeSHAP-IQ to compute ground-truth SVs and SIs values, which allows the evaluation games with substantially more features. Features are imputed according to the tree distribution [53, 60]. Consequently, the worth of the empty coalition containing no features is the tree's average prediction, e.g. baseline value.

Uncertainty Explanation. Similar to local explanations, the model's prediction with missing features imputed to a fixed datapoint is evaluated. Instead of referring to the predicted value, the value function is given by the prediction's uncertainty for which three measures are available: total, epistemic, and aleatoric uncertainty. Hence, the Shapley values of the features attribute their individual contribution to the decrease in uncertainty caused by their information.

Feature Selection. The available data is split into a training set $\mathcal{D}_{\text{Train}}$ and test set $\mathcal{D}_{\text{Test}}$. Given a learning algorithm $\mathcal{A}$, a coalitions worth $\nu(S)$ is given by the generalization performance of the model h_S on $\mathcal{D}_{\text{Test}}$ that results from applying $\mathcal{A}$ on $\mathcal{D}_{\text{Train}}$ using only features in S, known as *remove-and-refit*. The worth of the empty coalition is set to 0.

Ensemble Selection. Replacing features in feature selection by weak learners, and adapting the learning algorithm to construct an ensemble out of those, leads to ensemble selection. Each coalition S of base learners is evaluated by the performance of the resulting ensemble on a separate test set, known as *remove-and-re-evaluate*. Likewise, we set $\nu(\emptyset) = 0$.

Data Valuation. Continuing in the spirit of *remove-and-refit*, a new model is fitted to each coalition of datapoints. The generalization performance of the resulting model on a separate test set is set to be the coalition's worth. The value of the empty coalition is set to 0.

Dataset Valuation. The setup is analogous to data valuation, where instead of single datapoints are being understood as players, the available data is partioned and each subset is viewd as a player.

Cluster Explanation. Similar to feature selection, *remove-and-refit* is applied. Instead of fitting a model, a clustering algorithm forms multiple clusters on the dataset using only the available features of a coalition S. The worth $\nu(s)$ is given by a cluster evaluation score (see Tables 4 and 5 for details). A cluster score of 0 is assigned to the empty coalition.

Unsupervised Feature Importance. Given a coalition of features S, a set of datapoints can be understood as observations generated by a joint distribution of S and used to estimate this distribution by measuring the frequencies of feature values. This in turn allows to measure entropy and thus also total correlation of a subset of features which is used as the worth of S. Since the total correlation measures the amount of shared information, each feature's assigned Shapley value quantifies its contributed information to the group. The total correlation of the empty set is naturally 0.

Sum of Unanimity Models (SOUMs). A unanimity game is a synthetic game for a coalition $U \subseteq N$ with $\nu_U(T) := \mathbf{1}_{T \supseteq U}$, i.e. outputs one, if all players of U are present, and zero otherwise. The sum of unanimity model (SOUM) is a linear combination of randomly sampled unanimity games. For uniformly sampled coefficients $a_1, \dots, a_m \in [-1, 1]$ and subsets $U_1, \dots, U_m \subseteq N$ uniformly sampled by size, where we restrict the SOUM to specific maximum subset sizes. The value function then reads as

$$\nu(T) := \sum_{\ell=1}^{m} a_\ell \nu_{U_\ell}(T).$$

For SOUMs, the MIs as well as all SIs can be efficiently computed in linear time, cf. [28, Appendix B.7].

E.2 Computational Resources

This section contains additional information regarding the computational resources required for the empirical evaluation of this work. The main computational burden stems from pre-computing the benchmark games for $n \leq 16$ players and from running all of `shapiq`'s SV and 2-SII approximation methods. Still, the experiments require only a modest range of computational resources. The games are pre-computed on a "11th Gen Intel(R) Core(TM) i7–11800H 2.30GHz" machine requiring around 240 CPU hours. The approximation experiments have been run on a compute cluster using 80 CPUs of four "AMD EPYC 7513 32–Core Processor" units for 24 hours resulting in about 1920 CPU hours.

E.3 Data Availability and Reproducability

The data to the pre-computed games is available at `https://github.com/mmschlk/shapiq/tree/v1`. Utility functions exist in `shapiq` that automatically download and instantiate the games. The code for reproducing the experimental evaluation can be found at `https://github.com/mmschlk/shapiq/tree/v1/benchmark` and `https://github.com/mmschlk/shapiq/tree/v1/complexity_accuracy`.

F Additional Benchmark Approximation Results

This section contains additional experimental results. Mainly, this section contains exemplary MSE and Precision@5 approximation curves for a benchmark game of each application domain. These results can be found in Figures 9 to 11. The dataset names used to for the benchmark games are abbreviated as described in Appendix B. Figure 7 shows the overview of the benchmark results additionally to Figure 5 for the mean absolute error (MAE).

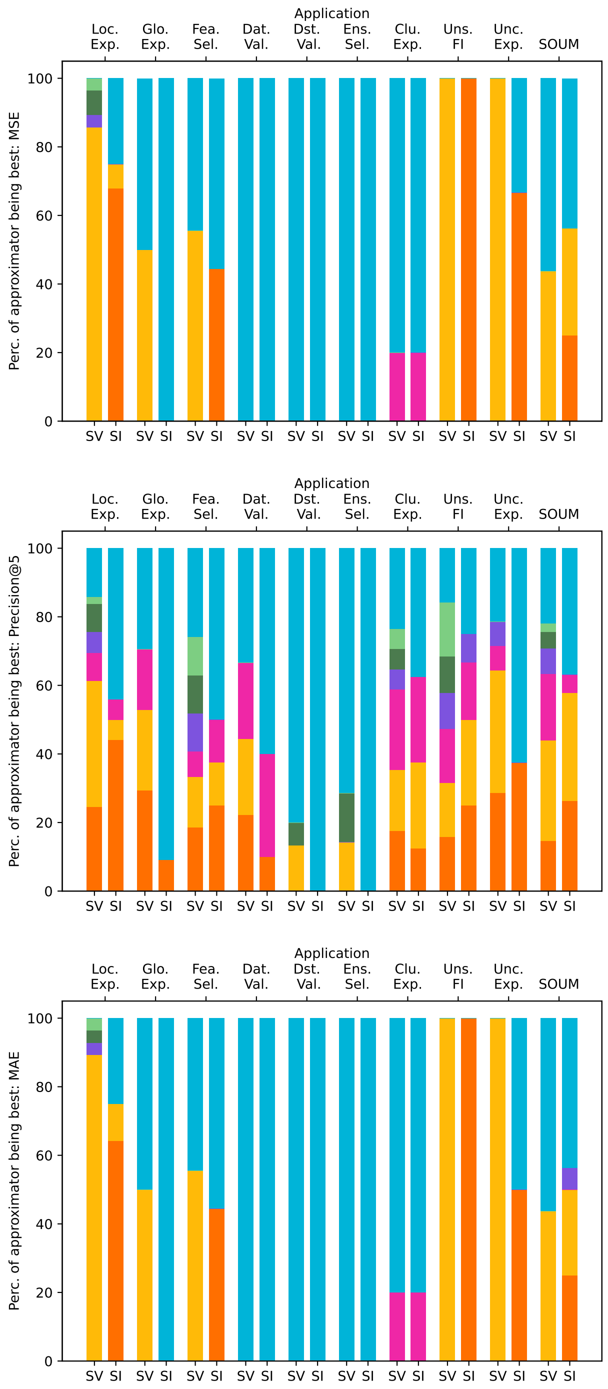


Figure 7: Benchmark overview approximation results for MSE (top), Precision@5 (middle), and MAE (bottom).

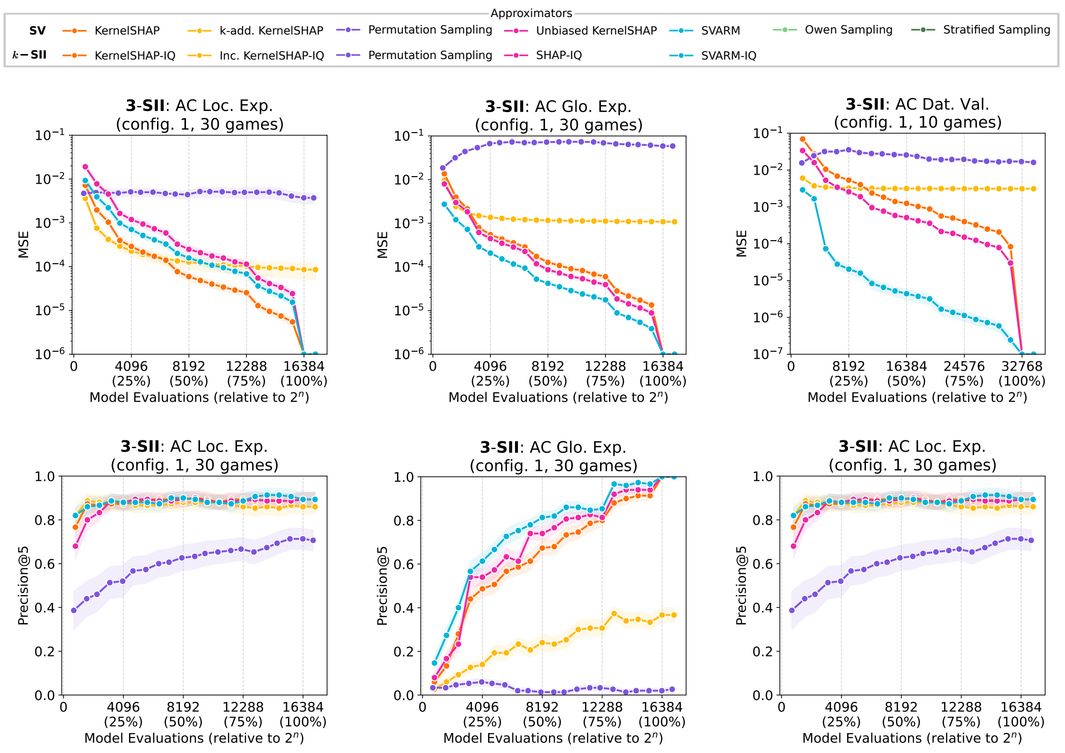


Figure 8: Approximation qualities in terms of MSE **(top)** and Precision@5 **(bottom)** for 3-SII higher-order interactions for three benchmark settings based on the *AdultCensus* (AC) dataset including Local Explanation **(left)**, Global Explanation **(middle)**, and Data Valuation **(right)**.

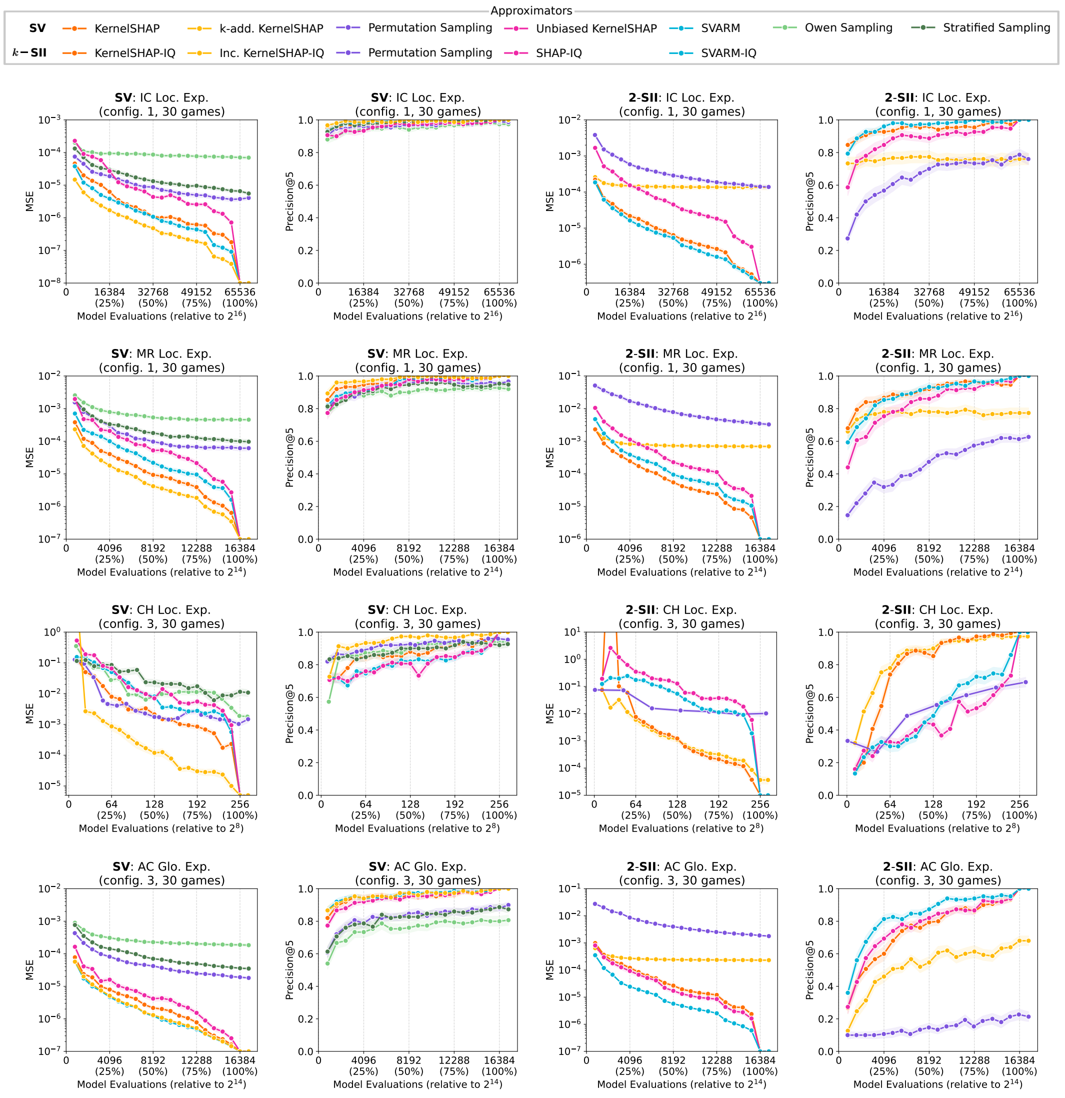


Figure 9: Additional SV (column one and two) and SI (column three and four) approximation results for different benchmark games from the Local Explanation (first row, vision transformer image classifier with $n = 16$ patches), Local Explanation (second row, language model predicting movie review sentiment with $n = 14$ words), Local Explanation (third row, dataset *CaliforniaHousing* with $n = 8$ features) and Global Explanation (fourth row, dataset *AdultCensus* with $n = 14$ features) domain.

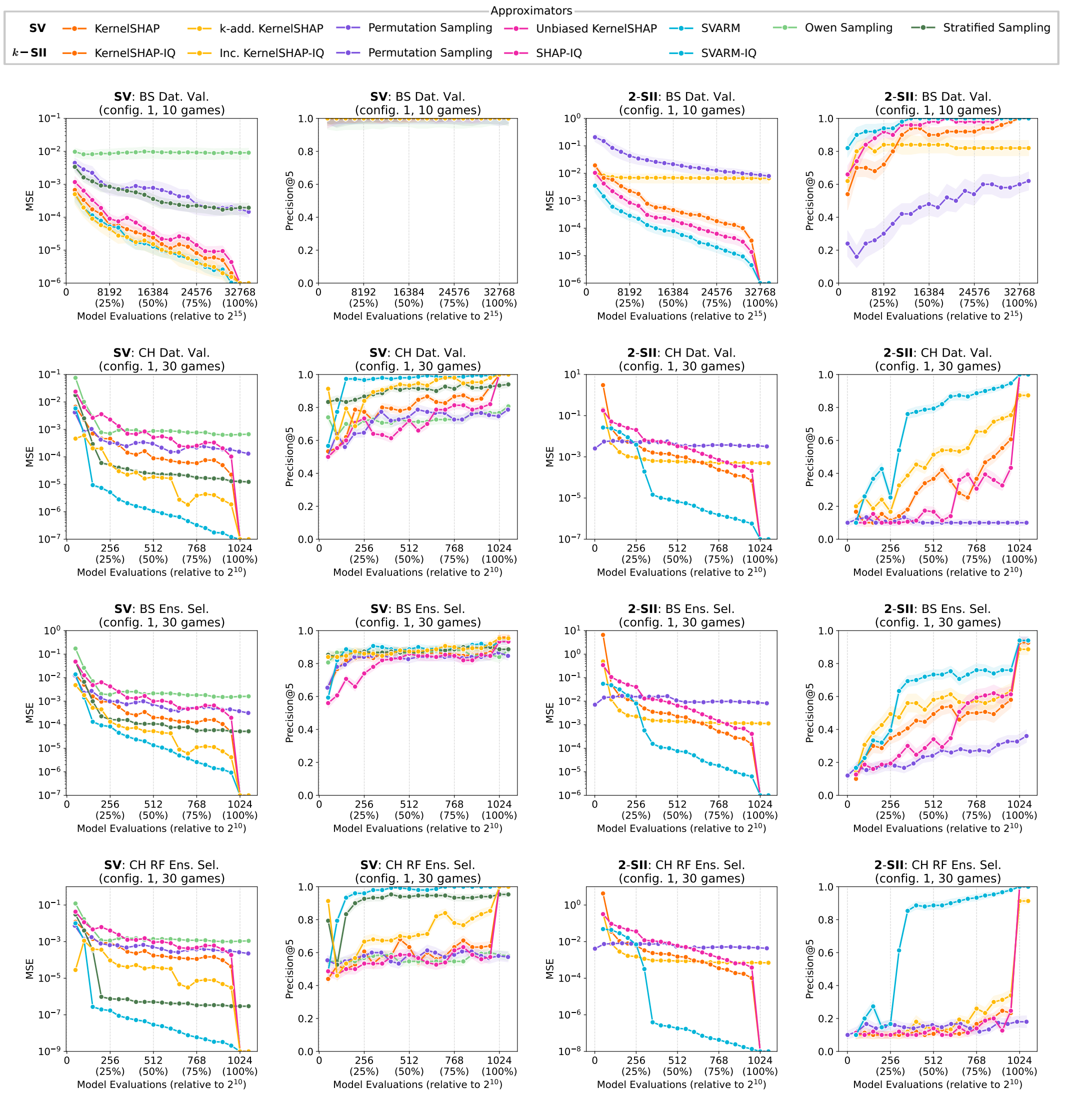


Figure 10: Additional SV (column one and two) and SI (column three and four) approximation results for different benchmark games from the Data Valuation (first row, *BikeSharing* with $n = 12$ features), Dataset Valuation (second row, *CaliforniaHousing* with $n = 8$ features), Ensemble Selection (third row, dataset *BikeSharing* with $n = 12$ features) and Random Forest Ensemble Selection (fourth row, dataset *CaliforniaHousing* with $n = 8$ features) domain.

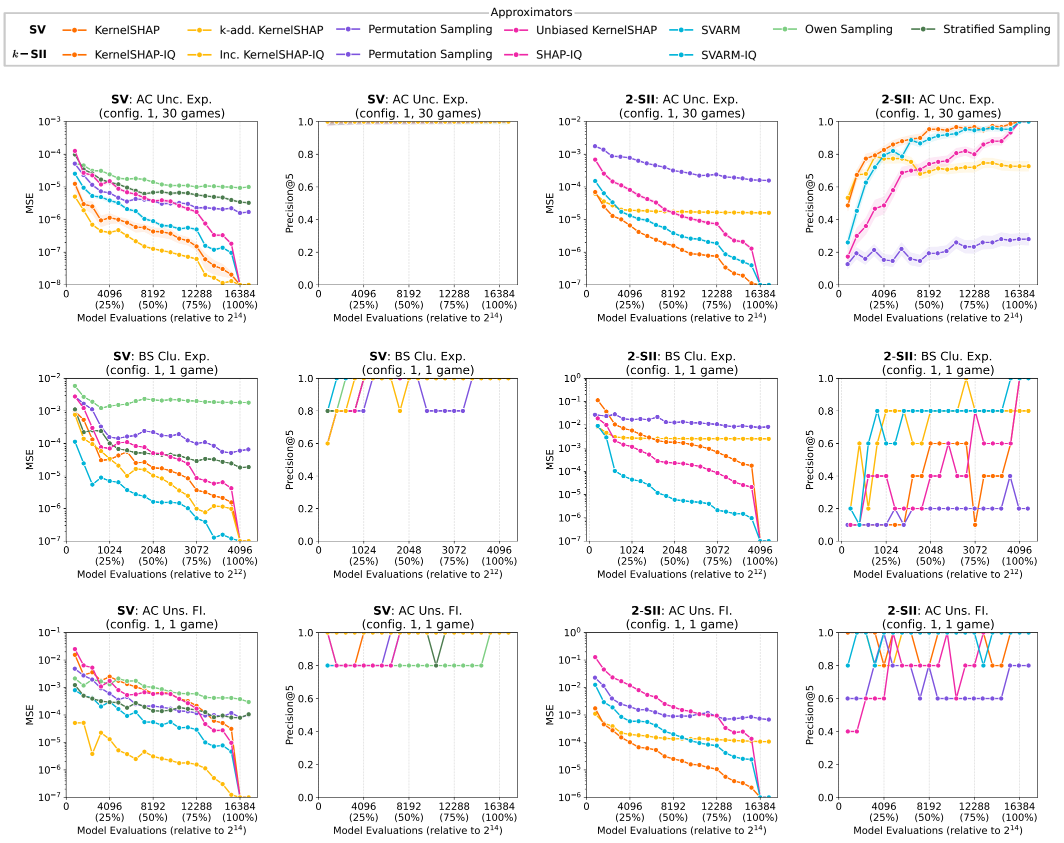


Figure 11: Additional SV (column one and two) and SI (column three and four) approximation results for different benchmark games from the Uncertainty Explanation (first row, *AdultCensus* with $n = 14$ features), Cluster Explanation (second row, *BikeSharing* with $n = 12$ features), and Unsupervised Feature Importance (third row, dataset *AdultCensus* with $n = 14$ features) domain.

G Glossary of Acronyms

***k*-SII** k-SV. 4, 5, 8

BGV Banzhaf GV. 17
BII Banzhaf II. 17
BV Banzhaf Value. 3–6, 17

CHGV Chaining GV. 17
CHII Chaining II. 17

FBII Faithful BII. 5, 8
FSII Faithful SII. 4, 5, 8

GV Generalized Value. 4, 5, 17

II Interaction Index. 4, 5, 17

MAE mean absolute error. 27
MI Möbius Interaction. 2, 5–9, 17, 23, 24, 26
ML machine learning. 1, 3–10, 18
MSE mean squared error. 8, 9, 27, 28

Precision@5 precision at five. 8, 9, 27, 28

SGV Shapley GV. 4, 6, 17
SI Shapley Interaction. 1–10, 22–26, 29–31
SII Shapley II. 4, 8, 17
STII Shapley Taylor II. 4, 8
SV Shapley Value. 1–5, 7, 8, 17, 22, 25, 26, 29–31

XAI explainable artificial intelligence. 3, 23

Checklist

The checklist follows the references. Please read the checklist guidelines carefully for information on how to answer these questions. For each question, change the default **[TODO]** to [Yes] , [No] , or [N/A] . You are strongly encouraged to include a **justification to your answer**, either by referencing the appropriate section of your paper or providing a brief inline description. For example:

- Did you include the license to the code and datasets? [Yes] See Section 1.
- Did you include the license to the code and datasets? [No] The code and the data are proprietary.
- Did you include the license to the code and datasets? [N/A]

Please do not modify the questions and only use the provided macros for your answers. Note that the Checklist section does not count towards the page limit. In your paper, please delete this instructions block and only keep the Checklist section heading above along with the questions/answers below.

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] *See Section 5, Limitations.*
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] *See Section 5, Broader impact.*
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes] *We read the ethics review guidelines.*
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments (e.g. for benchmarks)...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] *We ran experiments. All of the experimental data is available online at* `https://github.com/mmschlk/shapiq/tree/v1` *(directories benchmark & complexity accuracy), and described in Appendix E.3.*
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] *We train a collection of off-the-shelve models with mostly default parameters. All parameters are described in the technical supplement and or in the aformentioned code repository.*
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] *Experimental evaluations show also error bars.*
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] *We use no GPUs in the experiments. Compute resources used are outlined in Appendix E.2.*
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] *We cite all datasets used in Appendix B.2.*
   (b) Did you mention the license of the assets? [Yes] *We mention the licenses of all datasets used for which we could directly locate them in Appendix B.2.*
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes] *We included URLs to the source code and documentation in Section 3.*
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...

(a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
(b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]