shapiq: Shapley Interactions for Machine Learning

Maximilian Muschalik[1], Hubert Baniecki[2], Fabian Fumagalli[3], Patrick Kolpaczki[4], Barbara Hammer[3], and Eyke Hüllermeier[1]

NEURAL INFORMATION PROCESSING SYSTEMS

How do I measure **interactions** between multiple features for **black box** models beyond feature attributions?

SHAP

I want to use Shapley values for **other ML applications**. How do I compute them?

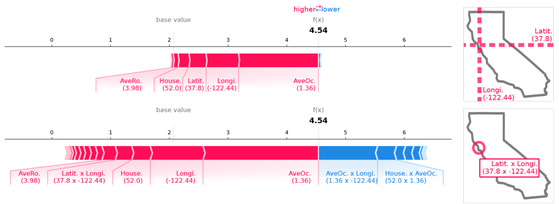
Explain Models with Shapley Interactions

Explaining models with shapiq is **easy**:

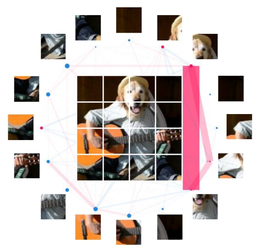
- Agnostic Explainer and Imputers
- Tree Explainer

```python
# 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=...)
```

"Does the ***location*** of my property affect its price?"



"Why is this a ***dog***?"

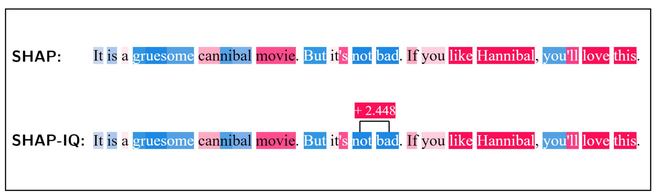


"How does my ***language model*** predict a positive sentiment?"

Sentiment Analysis Model

"It is a gruesome cannibal movie. But it's not bad. If you like Hannibal, you'll love this."

Explanation

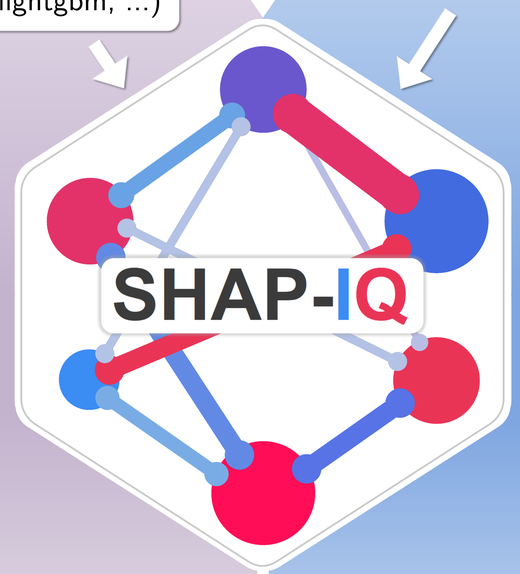


Any Model (e.g., torch, sklearn, ...)

Tree Model (e.g., xgboost lightgbm, ...)

Any Value Function (as a callable)

$\nu : \mathcal{P}(N) \to \mathbb{R}$



Interpretation: Shapley interactions generalize the Shapley value beyond individual effects up to **any-order** and capture **synergies** between features.

order 1:	up to order k:	up to order n:
Shapley Value	**Shapley Interactions**	**Möbius Interactions**

Faithfulness and **Complexity** →

Game Theory for General ML Applications

shapiq includes:

- **20** concepts (Shapley value and interactions, Banzhaf value and interactions, Faithful Shapley, Generalized values, Möbius, Core, ...)
- **14** state-of-the-art **approximators** and **exact computers**

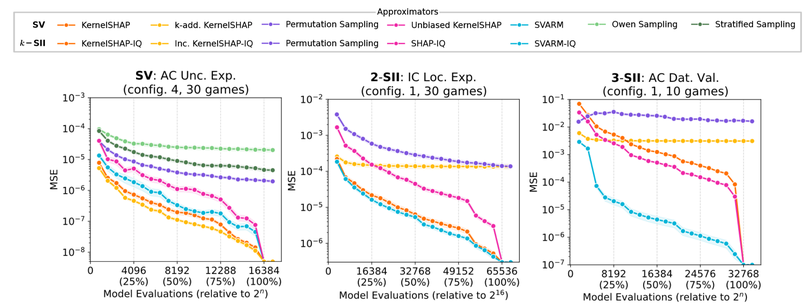
```python
import shapiq
class CountGame(shapiq.Game):
  def __init__(self, n_players): ...
  def value_function(coalitions):
    # define the worth of a coalition
    return np.sum(coalitions, axis=1)
game = CountGame(n_players=12)
# approximate SIs with KernelSHAP-IQ
approx = shapiq.KernelSHAPIQ(n=12)
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
```

Class	Shapley Interactions	Shapley Values
Approximator	KernelSHAP-IQ	KernelSHAP
	Inconsistent KernelSHAP-IQ	k_{ADD}-SHAP
	Faith-SHAP	Owen Sampling
	SHAP-IQ	Unbiased KernelSHAP
	SVARM-IQ	SVARM
	Permutation Sampling (SII)	Permutation Sampling (SV)
	Permutation Sampling (STII)	Stratified Sampling
Computer	Möbius Converter	
	Exact Computer	

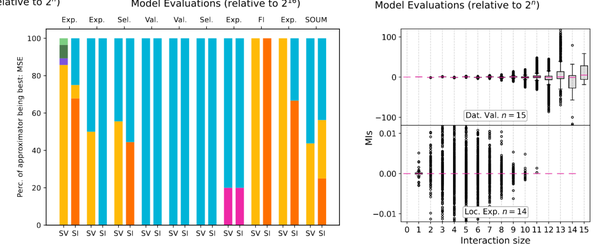
Benchmark of **11** ML domains (e.g., explanation, data valuation, uncertainty quantification, ...)

Games: 100 benchmark games with more than **2000** *pre-computed* configurations

Evaluation of Approximators on the Benchmark



- performance depends on the **domain**
- *kernel* and *stratification* based on top



1) LMU Munich
2) University of Warsaw and Warsaw University of Technology
3) Bielefeld University
4) Paderborn University

TRR 318
Funded by DFG Deutsche Forschungsgemeinschaft German Research Foundation

mmschlk/shapiq
PR Welcome!

```
pip install shapiq
```