applied sciences

MDPI

Article

C-SHAP: A Hybrid Method for Fast and Efficient Interpretability

Golshid Ranjbaran [1,*], Diego Reforgiato Recupero [2], Chanchal K. Roy [1] and Kevin A. Schneider [1]

[1] Department of Computer Science, University of Saskatchewan, Saskatoon, SK S7N 5A2, Canada; chanchal.roy@usask.ca (C.K.R.); kevin.schneider@usask.ca (K.A.S.)
[2] Department of Mathematics and Computer Science, University of Cagliari, 09124 Cagliari, Italy; diego.reforgiato@unica.it
* Correspondence: golshid.ranjbaran@usask.ca

Abstract: Model interpretability is essential in machine learning, particularly for applications in critical fields like healthcare, where understanding model decisions is paramount. While SHAP (SHapley Additive exPlanations) has proven to be a robust tool for explaining machine learning predictions, its high computational cost limits its practicality for real-time use. To address this, we introduce C-SHAP (Clustering-Boosted SHAP), a hybrid method that combines SHAP with K-means clustering to reduce execution times significantly while preserving interpretability. C-SHAP excels across various datasets and machine learning methods, matching SHAP's accuracy in selected features while maintaining an accuracy of 0.73 for Random Forest with substantially faster performance. Notably, in the Diabetes dataset collected by the National Institute of Diabetes and Digestive and Kidney Diseases, C-SHAP reduces the execution time from nearly 2000 s to just 0.21 s, underscoring its potential for scalable, efficient interpretability in time-sensitive applications. Such advancements in interpretability and efficiency may hold value for enhancing decision-making within software-intensive systems, aligning with evolving engineering approaches.

Keywords: interpretability in machine learning; interpretability; SHAP; LIME; C-SHAP; computational efficiency; K-means clustering; software engineering

Received: 15 November 2024
Revised: 28 December 2024
Accepted: 9 January 2025
Published: 11 January 2025

Citation: Ranjbaran, G.; Recupero, D.R.; Roy, C.K.; Schneider, K.A. C-SHAP: A Hybrid Method for Fast and Efficient Interpretability. *Appl. Sci.* **2025**, *15*, 672. https://doi.org/10.3390/app15020672

1. Introduction

Machine learning models are increasingly applied in high-stakes domains such as healthcare and finance, where decisions must be transparent and accountable. However, many models function as black boxes, making it difficult to understand their decision-making processes, which is problematic when trust and interpretability are essential [1–4]. In healthcare, AI-driven diagnostics must be interpretable for clinicians to ensure safe treatments [5,6], while in finance, regulatory transparency is critical [7–12].

For example, SHAP for Random Forest models in diabetes prediction required over 421 s on average, emphasizing the need for computationally efficient methods in time-critical domains like healthcare [13,14]. Furthermore, as machine learning advances into specialized fields such as software product-line engineering, the demand for models that are both interpretable and computationally efficient is more pressing than ever [15–17]. In this field, predictive models play a crucial role in tasks like defect prediction, feature selection, and system performance optimization, all of which require transparency and accountability to ensure reliable decision-making [18]. However, traditional interpretability methods often struggle with the high-dimensional and complex data structures typical in software engineering, where balancing execution speed and model clarity against the complexity of configurations is essential [19,20].

To address these issues, model interpretability methods like SHAP [21] and LIMEs (Local Interpretable Model-agnostic Explanations) [22] have been developed. SHAP has become a leading method for model interpretation by providing local and global feature attributions based on Shapley values from cooperative game theory [23]. In this context, Shapley values offer a way to fairly allocate a model's output among its features, akin to how they are used in cooperative game theory to distribute payouts to players based on their contributions to a coalition. This approach allows SHAP to explain the contribution of each feature to the prediction, making it a powerful tool for understanding both individual predictions and the model's overall behavior [24].

However, SHAP's computational complexity remains a significant obstacle; for instance, kernel SHAP for high-dimensional datasets can require thousands of perturbations, leading to execution times exceeding 32,000 s in certain cases [14].

However, SHAP's computational complexity presents a significant obstacle, especially for large-scale data and complex models like Random Forest and XGBoost (XGB), where the need to evaluate all possible feature combinations results in slow execution times [14,25–27]. This challenge is exacerbated in real-time applications, such as fraud detection or personalized healthcare, where quick decision-making is essential [28,29]. As a result, there is a pressing need for more efficient interpretability methods that balance computational cost and model accuracy.

LIMEs, while offering faster explanations by approximating models locally, compromise on global interpretability and accuracy [22,30,31]. Although LIMEs efficiently create local surrogate models, such as linear regressions, around individual predictions, their local approximations often fail to capture the global behavior of the model, leading to inconsistencies in feature attributions across different predictions [22,30,32,33]. Additionally, LIMEs can be unstable, with even slight changes in the input data potentially leading to different feature attributions, making them less reliable for domains like healthcare and finance, where both interpretability and accuracy are critical [31,34]. For example, LIMEs for text classification tasks can vary significantly, with execution times averaging 3 s per instance for 5000 perturbations, but with less stability compared to SHAP [22]

Clustering has long been used in feature selection approaches to enhance the efficiency of machine learning models by grouping similar data points, reducing dimensionality, and improving computational performance [35]. The clustering of K-means, in particular, helps identify and cluster similar instances, allowing for a more streamlined feature selection process by focusing on representative clusters rather than every individual data point [36]. This strategy has proven effective in reducing redundancy and computational overhead, especially in high-dimensional datasets where traditional feature selection techniques become infeasible [31,37].

Building on this concept, we propose C-SHAP, a novel approach that integrates K-means clustering to optimize SHAP's feature attribution process. SHAP, while robust and theoretically sound, becomes impractical for large datasets and complex models due to its high computational cost. C-SHAP overcomes this limitation by clustering similar data points, thereby reducing the number of calculations required for feature attributions. This not only significantly decreases the execution time but also preserves the interpretability and accuracy that make SHAP valuable. The motivation behind C-SHAP lies in its ability to provide scalable and fast explanations, making it suitable for real-time and large-scale applications and filling a critical gap in existing interpretability methods.

Furthermore, our approach, which combines SHAP's interpretability with K-means clustering, enables scalable, efficient interpretability tailored to large and intricate datasets found in software product lines. This work emphasizes the growing intersection between

machine learning interpretability and software engineering, reflecting broader trends in applying advanced analytics to improve software reliability and performance [38].

Therefore, this paper makes the following contributions:

1. We present C-SHAP, a novel extension of SHAP, which integrates K-means clustering to significantly reduce execution time across various machine learning models. For example, on the Diabetes dataset, which was collected by the National Institute of Diabetes and Digestive and Kidney Diseases https://www.kaggle.com/datasets/antoniofurioso/diabetes-dataset-sklearn (accessed on 14 November 2024), C-SHAP reduced the Random Forest execution time from 421.29 s to 0.39 s.
2. C-SHAP closely preserves select features of SHAP while significantly improving computational efficiency across various models. The Venn diagram analysis revealed that in models such as Random Forest, SVC (Support Vector Classifier), K-Nearest Neighbors, and Logistic Regression, the selected features between SHAP and C-SHAP were identical, with the only difference observed in the XGboost model.
3. C-SHAP was benchmarked against SHAP and LIME across multiple machine learning algorithms, demonstrating its versatility and efficiency improvements for both classification and regression tasks. The evaluations included datasets from domains such as healthcare and finance, confirming C-SHAP's scalability across diverse real-world applications.

This paper is structured as follows: It begins with a review of related works in Section 2, providing an overview of previous research and methods relevant to SHAP and feature interpretability. Next, Section 3 explains how SHAP interprets model decisions. Following this, Section 4 introduces C-SHAP, detailing how it enhances SHAP's performance by improving computational efficiency, supported by formulas and examples. The experimental evaluation of the proposed method across diverse datasets and models is presented in Section 5. In Section 6, the results are reported. The findings are then analyzed and compared with existing approaches in the discussion in Section 7. Limitations of the proposed approach are depicted in Section 8. Finally, the paper ends with a summary of contributions and proposed future research directions in Section 9.

2. Related Works

Feature selection is vital for improving the performance and interpretability of machine learning models by focusing on the most relevant variables, which simplifies models and reduces overfitting [39]. This process enhances efficiency and accuracy, particularly in critical fields like healthcare and finance, where model decisions must be transparent [40]. For example, in [41], feature selection methods such as LIME, SHAP, and CIU (Contextual Importance and Utility) were applied to datasets, including the Wisconsin Breast Cancer and SPECT (Single-Photon-Emission Computed Tomography) datasets, achieving execution times ranging from 116 s to 44,262 s.

Interpretability has become essential in AI [22], highlighting its role in building trust, especially in high-stakes applications. Methods that make models more understandable without sacrificing performance are crucial [7]. These works underscore the importance of both feature selection and interpretability in modern AI systems.

SHAP is a state-of-the-art method for feature attribution based on Shapley values from cooperative game theory, where Shapley values represent the fair contribution of each feature by quantifying its impact on the overall prediction based on all possible feature coalitions [42]. It provides local and global explanations, making it a powerful tool for understanding model behavior. SHAP's foundation in game theory ensures fairness by distributing the contribution of each feature to the final prediction [21]. This method has

been widely adopted due to its model-agnostic nature and ability to explain complex models [43].

One of SHAP's key strengths is its theoretical soundness, ensuring consistency in feature importance [14,21]. In comparison, LIME approximates model behavior by constructing interpretable models around individual predictions, offering local explanations for complex models [22]. By generating simpler models, such as linear regressions, to explain a single prediction, LIME is often faster than SHAP. For example, LIME provided explanations for a text classification dataset with an average execution time of ∼3 s per instance for 5000 perturbations [22], whereas Kernel SHAP required significantly more computations for models like MNIST [21].

However, unlike LIME [22], SHAP provides more reliable feature importance rankings by guaranteeing that higher importance scores accurately reflect their contribution to the model's output. This consistency has made SHAP a preferred method in critical domains such as healthcare and finance, where interpretability and trustworthiness are essential [44,45].

Despite its advantages in providing transparent and consistent feature attributions, SHAP is known for its computational inefficiencies, especially when dealing with large datasets or complex models like ensemble methods. The process of calculating Shapley values is inherently resource-intensive, as it requires evaluating all possible feature combinations to determine their contributions [42,46]. This complexity grows exponentially with the number of features, making SHAP computationally expensive, particularly in models such as Random Forest or XGBoost, where numerous decision paths must be analyzed for each prediction [21]. For instance, Marcilio et al. [14] observed that TreeSHAP's execution times ranged from 16.96 ms (Heart Disease) to 10,894.86 ms (Boston). Conversely, LIME achieved times as low as 0.23 s for simpler datasets [13].

Several studies have highlighted SHAP's performance limitations. For example, the work in [14] discusses how SHAP can become prohibitively slow for high-dimensional datasets due to its quadratic time complexity in model-agnostic cases. For example, kernel SHAP [23], the general purpose variant, is particularly slow compared to specialized versions like Tree SHAP, which is optimized for tree-based algorithms [43]. Benchmark studies have shown that the execution time for SHAP significantly increases with model depth and dataset size, particularly in XGBoost and Random Forest models [14].

Clustering techniques, such as K-means, have been extensively used in feature selection and optimization to improve the efficiency and accuracy of machine learning models. K-means helps group data points based on feature similarity, which not only reduces redundancy but also enhances the computational efficiency of feature attribution methods. For example, [37] demonstrated the use of kernel SHAP with the clustering of K-means to improve the performance of network anomaly detection models by selecting the most important features for anomaly detection. Similarly, [31] explored clustering to enhance SHAP's capability in the financial domain by clustering feature importance scores, improving both interpretability and execution speed.

Hybrid approaches that combine clustering with interpretability methods, like SHAP, have demonstrated significant potential in reducing computational overhead while maintaining interpretability [47,48]. For instance, ClusteredSHAP [30] used K-means clustering to reduce the number of gradient evaluations in deep learning models. ClusteredSHAP's focus is primarily on minimizing gradient evaluations for deep learning model explanations, with a particular emphasis on enhancing the efficiency of explanations for complex models, such as a 12-lead ECG classification model.

Building on SHAP's computational limitations and inspired by the clustering approach, we were motivated to introduce a novel method called C-SHAP. Unlike [30], which

targets gradient evaluations in deep learning, C-SHAP broadens the clustering concept to enhance the efficiency of SHAP computations in more general models and datasets, and this is basically a paper that introduces a method concept and method that can be applied to any dataset and domain. C-SHAP clusters similar data points using K-means and computes feature attributions for representative clusters. This approach minimizes redundant calculations while ensuring interpretability and accuracy in feature attributions. By focusing on optimizing SHAP computations more generally, C-SHAP aims to provide a comprehensive solution applicable to various domains, including software engineering, and its performance can be compared with alternative interpretable methods like LIME. In Table 1, a summary of the interpretability techniques, their associated models, and their execution times across various datasets is provided to complement the insights discussed.

Table 1. Interpretability techniques and their execution times across various datasets and models.

Ref.	Dataset	Machine Learning Model	Interpretability Techniques	Execution Times of Interpretability Techniques
[41]	Wisconsin Breast Cancer (BC, BCD), Diabetes, Parkinson's, SPECT, Single-Proton-Emission Computed Tomography (SPECTF)	Multi-layer Perceptron (MLP), Support Vector Machine (SVM), Random Forest, XGB, Naive Bayes (NB)	LIME, SHAP, Model-Agnostic Supervised Local Explanations (MAPLEs), Local Rule-Based Explanations (LOREs), CIU	LIME: 789–9211 s, SHAP: 741–32,566 s, MAPLE: 120–8333 s, LORE: 9411–44,262 s, CIU: 116–1062 s.
[49]	French Royalty KG, Lung Cancer KG	Random Forest, Decision Trees, AdaBoost Classifier	LIME	LIME: 58.09 s (French Royalty KG), 20.18 s (Lung Cancer KG).
[50]	Book Categorization, IMDb Sentiment	L2-regularized Logistic Regression (unigram Bag-of-Words (BOW) with Term Frequency-Inverse Document Frequency (TF-IDF) normalization)	LIME, Covariance (COVAR)	LIME: 0.750 ± 0.111 s per instance; COVAR: 0.014 ± 0.008 s per instance.
[51]	Medical Survey Dataset (Predicting Diabetes)	Random Forest	LIME, SHAP, Explain Like I'm 5 (ELI5)	LIME: execution time not explicitly stated but faster than SHAP; SHAP: computationally intensive.
[13]	Mortality, Diabetes, Drug Review, Side Effects	Random Forest	LIME, SHAP, Anchors, LORE, Influence-Based LIME (ILIME), MAPLE	LIME: 0.23–1.13 s; SHAP: 0.23–0.8 s; Anchors: 9.1–10.3 s; LORE: 1639–1642 s; ILIME: 2.1–2.35 s; MAPLE: 375–20,135 s.
[21]	MNIST	Decision Trees, CNNs	SHAP (Kernel, Deep), LIME	Kernel SHAP: computationally efficient; exact time not provided. Deep SHAP: faster than Kernel SHAP. LIME: slower with 50,000 perturbations.
[22]	Text Classification Dataset	SVM (RBF Kernel), Logistic Regression, Random Forest, Decision Trees, Nearest Neighbors	LIME, Submodular Pick LIME (SP-LIME), Parzen window method, Gradient-based methods	LIME: $\sim$3 s per instance for 5000 perturbations.
[14]	Indian Liver Disease, Heart Disease, Wine, Vertebral Column, Breast Cancer, Boston, Diabetes, NHANESI	XGBoost	SHAP (Tree SHAP), Recursive Feature Elimination (RFE), Mutual Information, Analysis of Variance (ANOVA)	TreeSHAP: 16.96–10,894.86 ms; RFE: 1118.04–251,591.22 ms; Mutual Information: 9.95–403.95 ms; ANOVA: 0.0–4.96 ms.
[30]	CPSC2018 (12-lead ECG classification)	ResNet34	SHAP (GradientExplainer, ClusteredSHAP, RandomGrad Explainer, DeepExplainer)	ClusteredSHAP: 10.63 s/sample; RandomGrad Explainer: 10.97 s/sample; GradientExplainer: 24.47 s/sample; DeepExplainer: 177.35 s/sample.
[43]	Mortality, Chronic Kidney Disease, Hospital Procedure Duration	Gradient-Boosted Trees, Random Forests, Decision Trees	TreeExplainer (SHAP values), Kernel SHAP, LIME, MAPLE	Kernel SHAP: exponential complexity $O(2^M)$; slower and less scalable for datasets with many features or samples.
[47]	Mobile Price Classification, MCCB Electrical Dataset	Random Forest Classifier, MLP, Random Forest Regressor	SHAP (Kernel Explainer, Tree Explainer), Local Interpretable Model-Agnostic SHAP Explanations (LIMASEs)	Kernel Explainer: 6.9–391.5 s (MLP); LIMASE: 0.08–6.83 s; Tree Explainer: exact times not specified but similar results with faster computations.

3. How SHAP Interprets Model Decisions

SHAP [21,52] interprets model decisions by assigning importance values to each feature based on its contribution to a prediction, which is crucial for understanding the role of different features in machine learning models. Interpretability, the ability to explain or understand the reasoning behind a model's predictions, is key to building trust in machine learning systems. It refers to a model's capacity to provide human-understandable insights into its decision-making process, making it easier to understand how and why certain predictions are made.

Grounded in cooperative game theory, SHAP uses Shapley values to fairly and consistently distribute credit among features, ensuring a balanced representation of their importance in the decision-making process. The Shapley value for a feature i is given by

$$\phi_i = \sum_{S \subseteq N \setminus \{i\}} \frac{|S|!(|N| - |S| - 1)!}{|N|!} (v(S \cup \{i\}) - v(S))$$

Here,

- ϕ_i is the Shapley value for feature i.
- N is the set of all features.
- S is a subset of features that does not include feature i.
- $v(S)$ is the prediction based only on the subset S of features.
- $v(S \cup \{i\})$ is the prediction based on the subset S plus the feature i.
- The fraction is the weighting factor, representing how to fairly distribute the marginal contributions across all possible subsets.

However, while SHAP provides highly accurate and consistent feature attributions, it is computationally expensive, especially when applied to larger datasets and complex algorithms like Random Forest and SVC. This results in slow execution times in real-world applications, limiting its practicality in scenarios where both speed and interpretability are crucial, such as medical diagnostics or financial modeling [30]. Even with optimizations like TreeSHAP, SHAP's computational burden can hinder its use in time-sensitive environments [25,26].

In summary, SHAP is a powerful tool for explaining model decisions, offering both fairness and accuracy in feature attribution. However, its computational cost remains a significant challenge, particularly in large-scale applications [22,28,53,54].

4. How C-SHAP Improves SHAP's Performance

In this study, we introduce C-SHAP, a hybrid approach designed to improve SHAP's execution time while maintaining its interpretability. C-SHAP enhances SHAP by preserving its core interpretability features while making the process more efficient. Specifically, C-SHAP leverages K-means clustering to group similar data points into clusters, allowing SHAP to evaluate representative clusters rather than each data point individually. The pseudocode corresponding to the implemented code is presented below (Algorithm 1).

To show the connection between K-means clustering and SHAP in C-SHAP, we introduce a formula that reflects how data points are clustered first using K-means, and then SHAP values are computed for each cluster's centroid rather than for individual data points.

Let

- X be the dataset with n samples and m features.
- C_k represent the k-th cluster centroid derived from K-means clustering, where $k \in \{1, 2, \ldots, K\}$ is the number of clusters.

Algorithm 1: Pseudocode for C-SHAP

Input: Dataset X, y, Machine Learning Model M, Number of Clusters K
Output: Top-ranked features
1 **Train the Model:** Train M using the training dataset $X_{\text{train}}, y_{\text{train}}$.
2 **Apply K-Means Clustering:** Cluster the training data X_{train} into K clusters and compute the centroids of the K clusters.
3 **Compute SHAP Values:** Use a SHAP explainer to calculate SHAP values for the K centroids.
4 **Rank Features:** Aggregate SHAP values for each centroid and rank features based on their importance.

The SHAP values for C-SHAP, $\phi_i(C_k)$, can be computed by applying SHAP to the centroid of each cluster, rather than individual data points. The process involves

$$\phi_i(C_k) = \sum_{S \subseteq N \setminus \{i\}} \frac{|S|!(|N| - |S| - 1)!}{|N|!} (v(S \cup \{i\}, C_k) - v(S, C_k))$$

Here,

- C_k is the centroid of the k-th cluster found by K-means.
- $v(S, C_k)$ represents the model's output when using the feature subset S for the centroid C_k.
- $\phi_i(C_k)$ is the SHAP value for feature i in the cluster C_k.

To illustrate the effectiveness of C-SHAP in a regression context, four critical factors in predicting diabetes outcomes, Body Mass Index (BMI), age, sex, and blood pressure (bp), were selected for analysis. In Figure 1, the feature importance rankings are displayed for SHAP, C-SHAP, and LIME. The feature ranked as number one, highlighted in blue, is identified as the most important according to SHAP values. In contrast, the feature ranked as number four, marked in red, is considered the least important based on SHAP values. To simplify the analysis in this toy example, four out of the ten available features in the Diabetes dataset were selected solely for illustrating feature importance rankings. While the full regression model utilized all features for predicting diabetes outcomes, these four were specifically chosen to demonstrate how feature importance rankings are assigned by SHAP, C-SHAP, and LIME. The order of importance for SHAP and C-SHAP is identical, while LIME shows a slight variation in the feature rankings. In Figure 2, the SHAP values for each feature are shown for both the SHAP and C-SHAP methods. The similarity in these values indicates that both methods produce consistent feature importance rankings. This alignment confirms that C-SHAP can reliably replicate the insights provided by SHAP while maintaining accuracy in assessing the significance of each feature.

The images in Figures 3 and 4 represent a small slice of decision trees generated using SHAP and C-SHAP, respectively. Each node in these trees corresponds to a decision point based on the bmi and bp features. Each node in the decision trees provides key details about the decision-making process, represented in four rows:

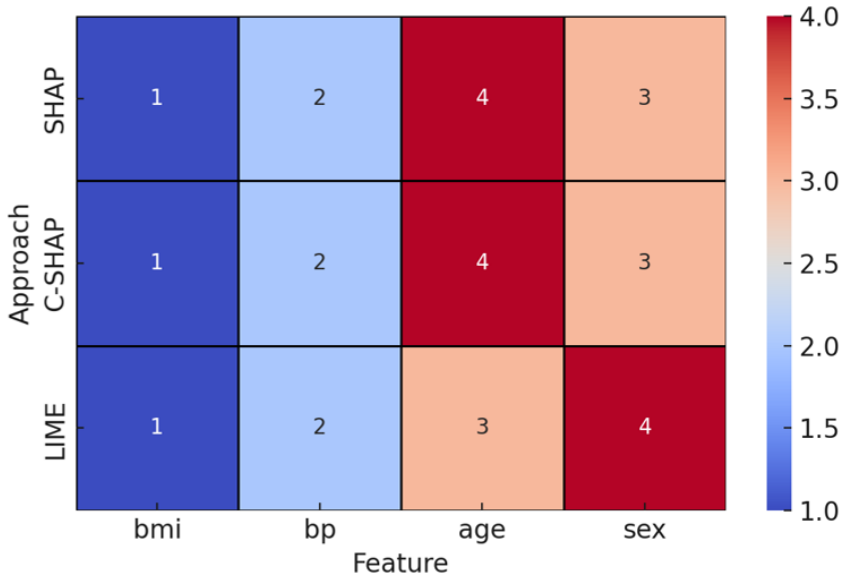

Figure 1. Feature importance rankings in SHAP, C-SHAP, and LIME.

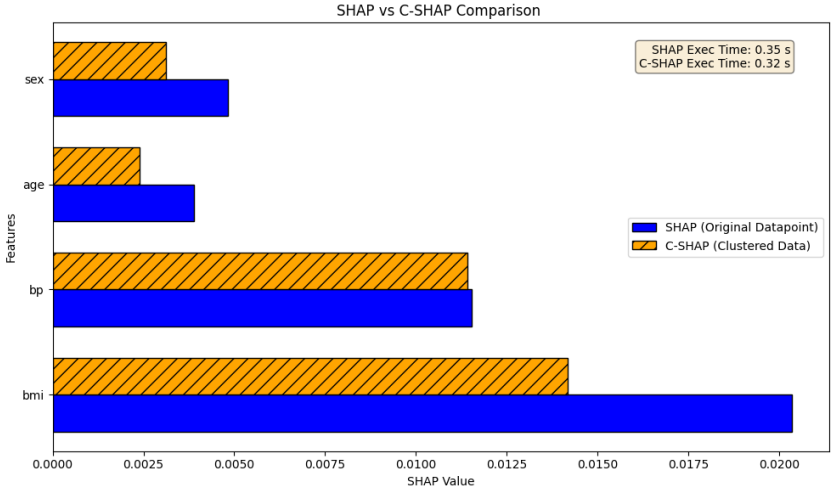

Figure 2. SHAP values for the SHAP and C-SHAP approaches are compared across four features.

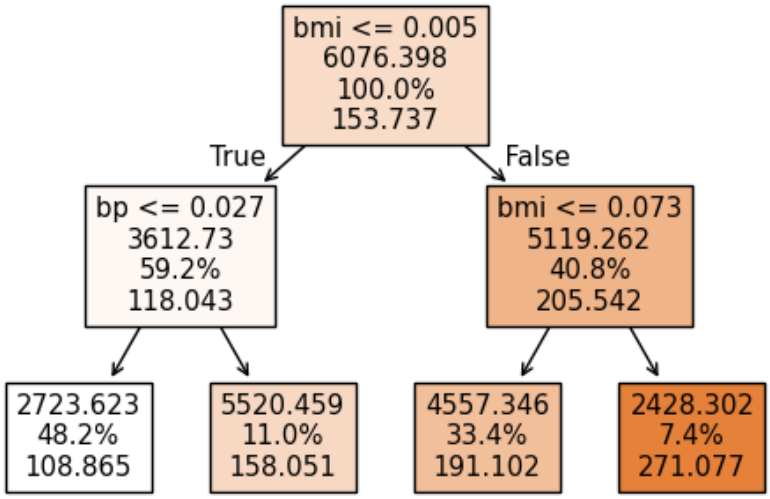

Figure 3. Decision tree of SHAP approach.

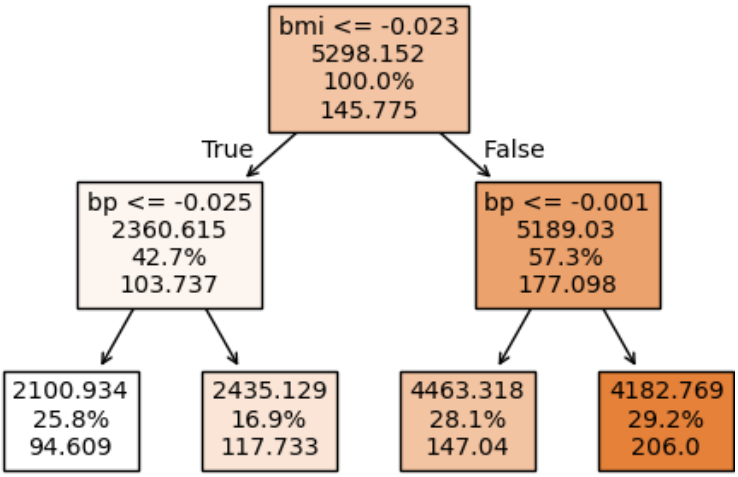

Figure 4. Decision Tree of C-SHAP approach. Note: the BMI threshold $bmi \leq -0.023$ observed in the decision tree reflects the scaled value of the BMI after preprocessing. The original BMI values were normalized using z-score normalization to improve model performance, and the thresholds should be interpreted in this context.

- Condition (top row): This specifies the splitting criterion at the node, based on a feature. For example, in Figure 3, "bmi $\leq$ 0.005" is a decision point that splits data based on the BMI feature. In Figure 4, the decision point for the root node is "bmi ≤ -0.023."
- SHAP value (second row): This value represents the predicted output or cumulative SHAP contribution at the node. For example, the root node in Figure 3 displays a SHAP value of 6076.398, while the corresponding node in Figure 4 shows 5298.152, indicating differences in the values processed due to clustering in C-SHAP.
- Dataset coverage (third row): This percentage indicates the proportion of the total dataset reaching the node. For instance, both root nodes in Figures 3 and 4 begin with 100% coverage, showing the full dataset before any splits.
- Sample count (fourth row): This denotes the volume of data or number of samples processed at the node. In the root node of Figure 3, the count is 153.737, compared to 145.775 in Figure 4. This difference highlights how C-SHAP clusters data points, potentially altering sample sizes across nodes.

Thus, Figure 3 illustrates the original decision thresholds and sample distributions between the nodes that SHAP creates. Here, the decision path dives into finer-grained details as it does not group similar data points, leading to a greater number of samples per node.

Conversely, Figure 4 shows how the model structure changes with clustering. By clustering similar data points, C-SHAP reduces the sample size at each node, which leads to a more streamlined tree with fewer decision points while maintaining comparable error margins. So, in terms of performance, C-SHAP demonstrates advantages over SHAP by requiring fewer computations due to clustering, which can result in faster execution times and reduced error margins.

In conclusion, by clustering data points, C-SHAP retains SHAP's ability to provide both local (individual predictions) and global (overall model behavior) explanations, ensuring that the transparency of the model's decision-making process is not compromised. The method maintains the high-quality interpretability offered by SHAP while addressing its execution time limitations, making it suitable for real-time applications that demand both speed and interpretability.

5. Experimental Evaluation

In this paper, we used various datasets with different machine learning algorithms and applied multiple metrics to evaluate C-SHAP performance.

5.1. Dataset

The paper [14], which was the first to leverage SHAP for feature explanation, utilized a set of benchmark datasets. We used the same datasets to ensure comparability between our methods. These datasets encompass both classification and regression tasks, providing a comprehensive evaluation scope. The classification datasets include Indian Liver Disease, https://www.kaggle.com/datasets/uciml/indian-liver-patient-records (accessed on 14 November 2024); Heart Disease, https://www.kaggle.com/datasets/johnsmith88/heart-disease-dataset (accessed on 14 November 2024); Wine https://www.kaggle.com/datasets/yasserh/wine-quality-dataset (accessed on 14 November 2024), Vertebral Column, https://www.kaggle.com/datasets/caesarlupum/vertebralcolumndataset (accessed on 14 November 2024); and Breast Cancer, https://www.kaggle.com/datasets/uciml/breast-cancer-wisconsin-data (accessed on 14 November 2024). For regression tasks, we used the Diabetes dataset we mentioned earlier, https://www.kaggle.com/datasets/mathchi/diabetes-data-set (accessed on 14 November 2024). Table 2 illustrates the mentioned datasets. These datasets provided a comprehensive evaluation across various tasks.

Table 2. Datasets used in the experiments.

Dataset Name	Data Points	Number of Features	Type
Indian Liver Disease	583	10	Classification
Heart Disease	303	13	Classification
Wine	178	13	Classification
Vertebral Column	310	6	Classification
Breast Cancer	569	30	Classification
Diabetes	442	10	Regression

5.2. Evaluation Metrics

To evaluate the performance of C-SHAP, we implemented SHAP and LIME using various machine learning algorithms, including Random Forest (RF), SVC, XGBoost, Logistic Regression (LR), and K-Nearest Neighbors (K-NNs). These models were applied to the different datasets mentioned in Table 2. The experiments were conducted using the free version of Google Colab. To conduct the experiments in this study, we used the sklearn, https://scikit-learn.org/stable/ (accessed on 14 November 2024); Pandas, https://pandas.pydata.org/ (accessed on 14 November 2024); SHAP, https://pypi.org/project/shap/ (accessed on 14 November 2024); and matplotlib_venn, https://python-graph-gallery.com/venn-diagram/ (accessed on 14 November 2024) libraries. We reported execution time, accuracy, and visualized feature selection similarities using Venn diagrams [55,56].

Accuracy is reported in the results to demonstrate the performance of the selected algorithm on the dataset. While accuracy measures performance rather than feature interpretability, it highlights the importance of feature selection in achieving optimal results and improving both classification and regression processes.

In addition, Venn diagrams are a graphical tool used to show the overlap between sets. In this context, they illustrate how the important features identified by C-SHAP and SHAP overlap, highlighting mutual features between the two methods.

5.3. Assessing Scalability

As machine learning models are increasingly applied to large-scale datasets, the scalability of interpretability methods becomes a critical factor in their practical adoption. Methods like SHAP, while robust and widely used, often exhibit computational inefficiencies as datasets' size increases due to their quadratic complexity. This section evaluates the scalability of C-SHAP compared to SHAP by analyzing their execution times across datasets of varying sizes.

To assess scalability, we selected the Diabetes dataset and created subsets with sizes of 10%, 25%, 50%, 75%, and 100% of the original dataset. For each subset, we applied SHAP and C-SHAP to compute feature attributions for Random Forest models. The execution times were recorded for generating explanations for the entire dataset subset. The scalability of each method was evaluated based on the growth in execution time as the dataset size increased.

The results, summarized in Table 3 and visualized in Figure 5, demonstrate the computational advantage of C-SHAP over SHAP. As dataset size increased, SHAP exhibited exponential growth in execution time, while C-SHAP achieved linear scalability, significantly reducing execution times compared to SHAP. This highlights C-SHAP's suitability for large-scale applications without compromising interpretability.

Table 3. Execution times of SHAP and C-SHAP on the Diabetes dataset for different dataset sizes.

Dataset Size (%)	SHAP Execution Time (Sec)	C-SHAP Execution Time (Sec)
10%	43.12	0.15
25%	126.45	0.28
50%	269.87	0.40
75%	382.43	0.55
100%	439.15	0.29

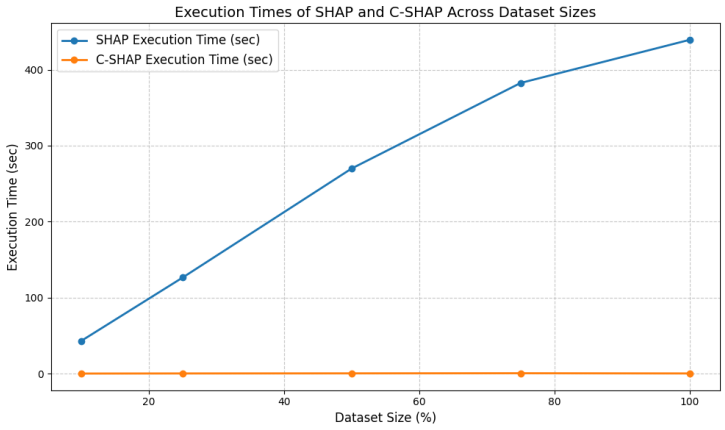

Figure 5. Comparison of execution times for SHAP and C-SHAP across dataset sizes on the diabetes dataset.

6. Results

In the experiments, we utilized the datasets introduced in Section 5.1 to evaluate the performance of the proposed C-SHAP method in comparison to standard SHAP and LIME. The metrics used to assess performance include accuracy and execution time, presented in tables along with Venn diagrams, to demonstrate the feature selection efficiency of SHAP and C-SHAP. In the following Venn diagrams, we illustrate the number of features

identified as the most important by each of the two methods considered, highlighting the overlap and differences in selected features between SHAP and C-SHAP.

The results include accuracy to showcase how effectively the selected algorithm performs on the dataset. While accuracy assesses overall model performance rather than feature interpretability, it is not directly affected by methods like LIME, SHAP, or C-SHAP. These methods focus on explaining and interpreting feature importance rather than influencing accuracy. However, accuracy does underscore the importance of selecting relevant features, as effective feature selection contributes to optimal outcomes and enhances the accuracy of classification and regression tasks, ultimately leading to more robust and reliable predictive models.

The Venn diagrams confirm that C-SHAP not only reduces execution time but also maintains interpretability by selecting relevant features, highlighting the efficiency gained through clustering in the C-SHAP method. The results demonstrate that C-SHAP significantly improves execution time while delivering performance comparable to SHAP. In the Venn diagrams, three colors represent feature selection: red indicates features chosen by SHAP, blue represents features chosen by C-SHAP, and green shows the mutual features selected by both SHAP and C-SHAP in each algorithm and dataset. The numbers within each section indicate the count of selected features in each category.

In this experiment, we utilized the default settings of the SHAP library in Python, https://shap.readthedocs.io/en/latest/ (accessed on 14 November 2024). SHAP selects features by calculating SHAP values, which measure each feature's contribution to the prediction. Features with higher absolute SHAP values are considered more influential, allowing for feature ranking based on their impact on the model's decisions.

In the Diabetes dataset (Table 4), C-SHAP significantly reduces the execution time for SVC, bringing it down from 1968.09 s to just 0.21 s, underscoring its efficacy in regression tasks. As illustrated in Figure 6, both SHAP and C-SHAP consistently identified the same two out of ten features as the most influential.

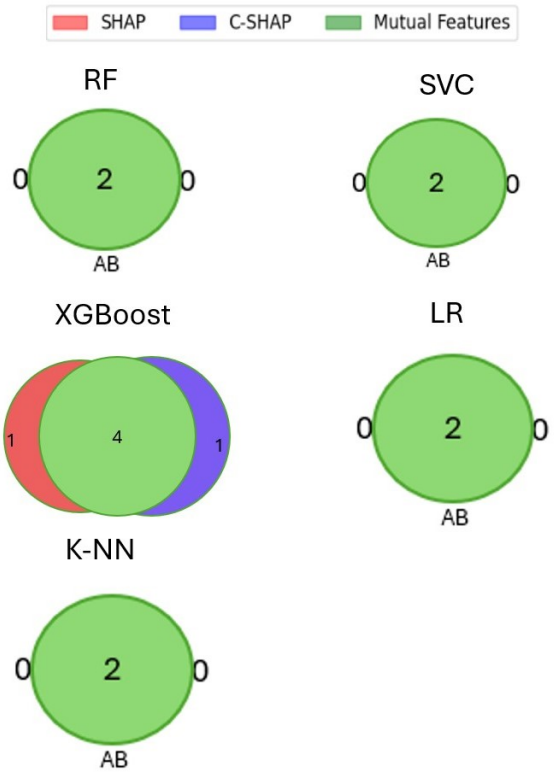

Figure 6. The number of features in the Diabetes dataset (total number of features is 10) extracted by SHAP and C-SHAP and their mutual features.

Table 4. Diabetes dataset performance, presenting accuracy metrics and execution times (seconds) for LIME, SHAP, and C-SHAP.

Model	Accuracy	LIME (Sec)	SHAP (Sec)	C-SHAP (Sec)
Random Forest	0.73	0.14	421.29	0.39
SVC	0.73	0.19	1968.09	0.21
XGBoost	0.72	0.12	0.06	0.05
LR	0.73	0.06	165.72	0.05
K-NNs	0.68	0.09	350.84	0.07

Similarly, for the Heart Disease dataset, in a classification task (Table 5), C-SHAP decreased the Random Forest execution time from 439.15 s to 0.29 s. Figure 7 confirms that SHAP and C-SHAP converge on the selection of the same two most significant features out of thirteen.

Table 5. Heart Disease dataset performance, presenting accuracy metrics and execution times (seconds) for LIME, SHAP, and C-SHAP.

Model	Accuracy	LIME (Sec)	SHAP (Sec)	C-SHAP (Sec)
Random Forest	0.736	0.09	439.15	0.29
SVC	0.73	0.20	1984.44	0.22
XGBoost	0.72	0.11	0.06	0.05
LR	0.73	0.07	176.95	0.07
K-NNs	0.68	0.08	363.78	0.07

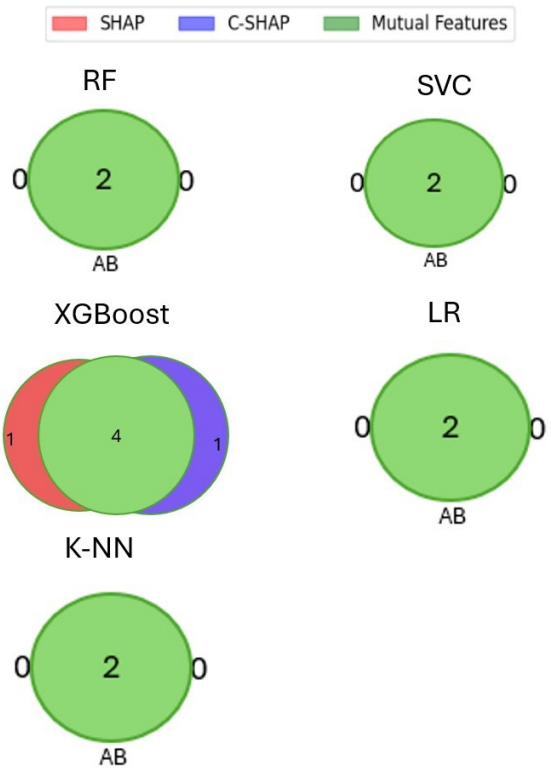

Figure 7. The number of features in the Heart Disease dataset (total number of features is 13) extracted by SHAP and C-SHAP and their mutual features.

In another classification task involving the Indian Liver Disease dataset (Table 6), C-SHAP reduced the execution time for Random Forest from 953.18 s to 0.39 s. Figure 8 shows the consistent selection of the same two critical features out of ten by both SHAP and C-SHAP.

Table 6. Indian Liver Disease dataset performance, presenting accuracy metrics and execution times (seconds) for LIME, SHAP, and C-SHAP.

Model	Accuracy	LIME (Sec)	SHAP (Sec)	C-SHAP (Sec)
Random Forest	0.74	0.12	953.18	0.39
SVC	0.73	0.20	3954.12	0.72
XGBoost	0.70	0.14	0.05	0.04
LR	0.72	0.12	368.42	0.17
K-NNs	0.69	0.11	844.85	0.20

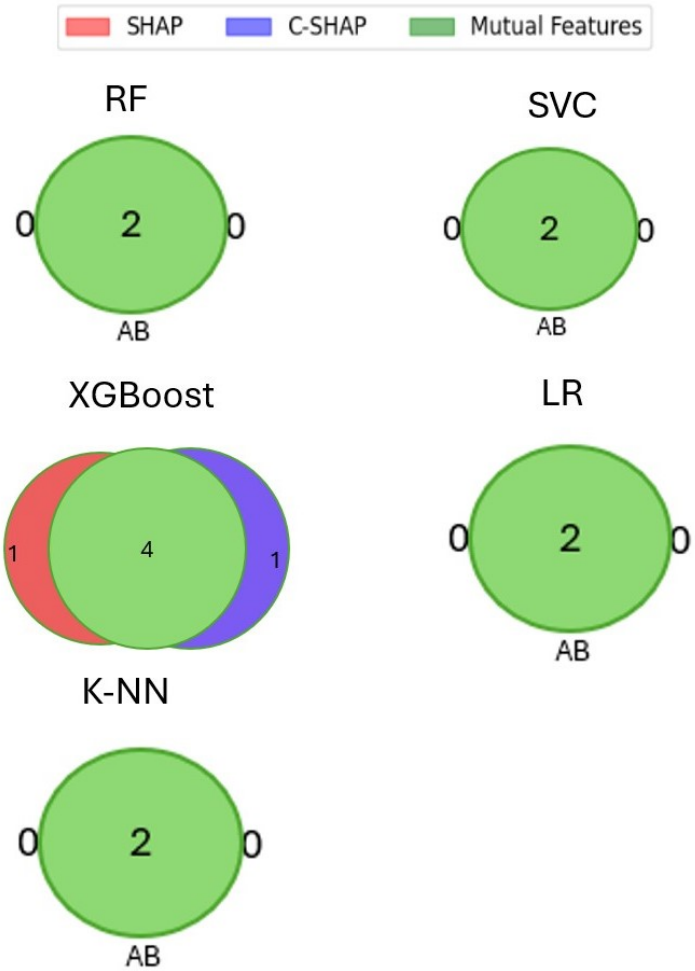

Figure 8. The number of features in the Indian Liver Disease dataset (total number of features is 10) extracted by SHAP and C-SHAP and their mutual features.

For the Breast Cancer dataset (Table 7), which was also used for a classification task, C-SHAP cut down the execution time from 525.33 to 0.43 s for Random Forest and from 1044.86 to 0.15 s for SVC. Both methods identified the same two key features out of thirty, as depicted in Figure 9.

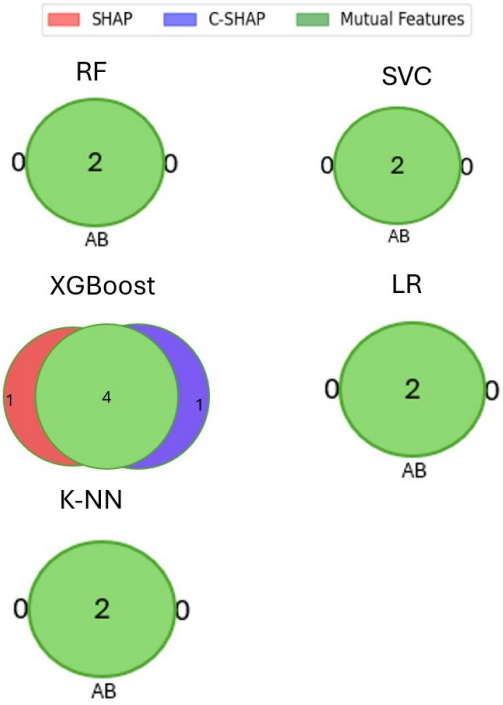

Figure 9. The number of features in the Breast Cancer dataset (total number of features is 30) extracted by SHAP and C-SHAP and their mutual features.

Table 7. Breast Cancer dataset performance, presenting accuracy metrics and execution times (seconds) for LIME, SHAP, and C-SHAP.

Model	Accuracy	LIME (Sec)	SHAP (Sec)	C-SHAP (Sec)
Random Forest	0.98	0.12	525.33	0.43
SVC	0.97	0.10	1044.86	0.15
XGBoost	0.95	0.13	0.03	0.04
LR	0.95	0.09	245.18	0.07
K-NNs	0.97	0.10	894.58	0.15

In the Wine dataset (Table 8), which is frequently used for classification, C-SHAP reduced the Random Forest execution time from 132.42 s to 0.58 s and the SVC execution time from 300.64 s to 0.69 s. Figure 10 illustrates the consistent selection of three influential features out of thirteen by both SHAP and C-SHAP.

Table 8. Wine dataset performance, presenting accuracy metrics and execution times (seconds) for LIME, SHAP, and C-SHAP.

Model	Accuracy	LIME (Sec)	SHAP (Sec)	C-SHAP (Sec)
Random Forest	1.0	0.13	132.42	0.58
SVC	0.76	0.16	300.64	0.69
XGBoost	0.96	0.19	0.08	0.08
LR	1.0	0.12	63.48	0.29
K-NNs	0.74	0.13	103.10	0.34

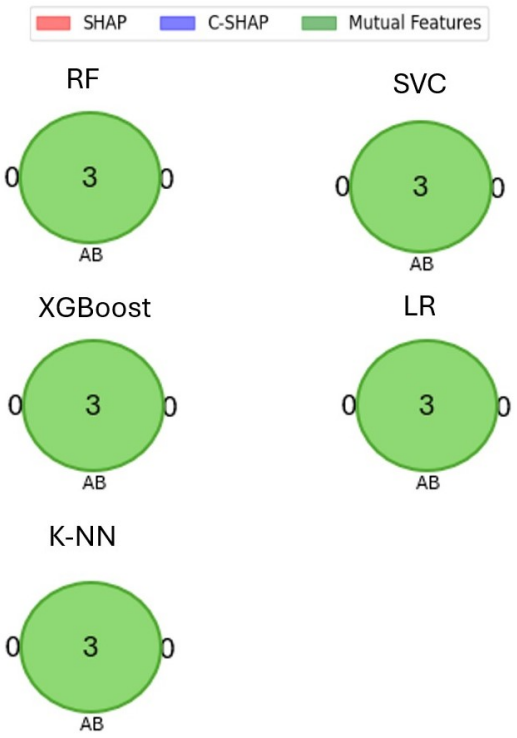

Figure 10. The number of features in the Wine dataset (total number of features is 13) extracted by SHAP and C-SHAP and their mutual features.

Lastly, for the Vertebral Column dataset (Table 9), C-SHAP decreased the Random Forest execution time from 19.07 s to 0.36 s and the SVC execution time from 41.53 s to 0.11 s. Both methods consistently identified three critical features out of six, as seen in Figure 11.

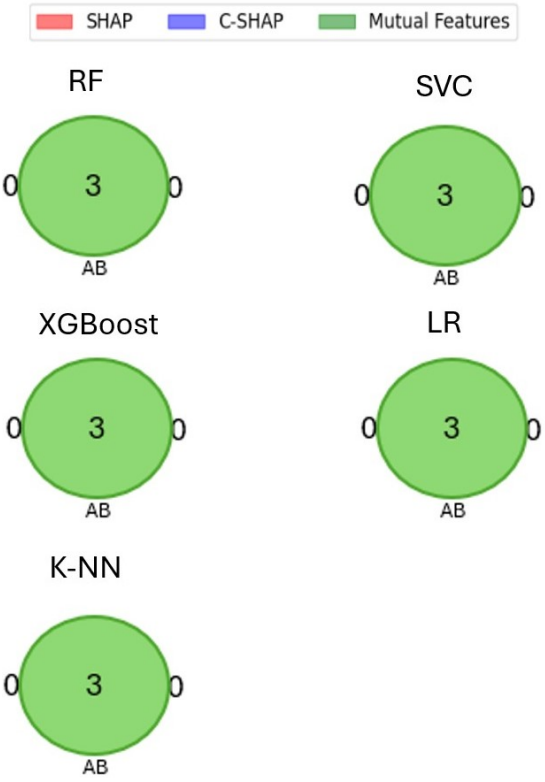

Figure 11. The number of features in the Vertebral Column dataset (total number of features is 6) extracted by SHAP and C-SHAP and their mutual features.

Table 9. Vertebral Column dataset performance, presenting accuracy metrics and execution times (seconds) for LIME, SHAP, and C-SHAP.

Model	Accuracy	LIME (Sec)	SHAP (Sec)	C-SHAP (Sec)
Random Forest	0.84	0.09	19.07	0.36
SVC	0.80	0.09	41.53	0.11
XGBoost	0.83	0.08	0.09	0.09
LR	0.83	0.06	11.49	0.03
K-NNs	0.78	0.07	16.61	0.03

In addition, as shown in the results, in both the Vertebral Column and Wine datasets, the selected features were consistent across all machine learning methods. Similarly, for the remaining datasets, the selected features were the same across all the methods, with only minor differences observed for the XGBoost approach.

Overall, C-SHAP delivers both speed and consistency in feature selection across datasets, matching SHAP's accuracy in feature selection while drastically reducing execution times.

7. Discussion

This paper showcases the effectiveness of the proposed approach, C-SHAP, in enhancing computational efficiency while maintaining interpretability. C-SHAP addresses one of SHAP's primary limitations, execution time, by leveraging K-means clustering to reduce the number of calculations required for feature attribution. This approach effectively balances accuracy in feature selection and computational demands. In a regression context, for example, C-SHAP reduced SVC's execution time for the Diabetes dataset from nearly 2000 s to just 0.21 s, underscoring its potential for real-time applications in fields like healthcare, where swift and accurate interpretations are essential.

Furthermore, the findings across classification datasets, such as the Heart Disease and Indian Liver Disease datasets, reinforce C-SHAP's ability to accelerate interpretability processes. In the Heart Disease dataset, C-SHAP reduced the Random Forest execution time from 439.15 s to 0.29 s, while in the Indian Liver Disease dataset, it lowered Random Forest's runtime from 953.18 to 0.39 s. These results highlight C-SHAP's versatility across various machine learning tasks and models, as it consistently identifies the same key features as SHAP while drastically reducing execution times. This capacity to enhance performance without sacrificing feature selection accuracy positions C-SHAP as a valuable tool for scalable machine learning interpretability.

In addition, as indicated in the results, SHAP and C-SHAP show similar results in feature selection for most algorithms because they handle linear and non-linear relationships in a similar way. However, XGBoost, being more sensitive to feature interactions, may lead C-SHAP to group data points differently due to clustering, which could cause slight variations in feature importance selection compared to SHAP. This explains the difference observed specifically with XGBoost.

Adaptive Clustering in C-SHAP

In this study, we employed a fixed number of clusters for K-means clustering based on the observation that the datasets used are of similar size and complexity. While this approach simplifies implementation and ensures consistency across experiments, it may not be optimal for larger or more complex datasets, where the fixed number of clusters could limit the method's scalability and adaptability.

To address this limitation, adaptive clustering strategies can be incorporated into C-SHAP. Adaptive clustering dynamically determines the number of clusters based on the characteristics of the dataset, such as its size, dimensionality, or intrinsic structure. This allows the method to adjust to varying dataset complexities, improving both interpretability and computational efficiency. For instance, techniques like the elbow method, silhouette analysis, or gap statistics can be used to determine the optimal number of clusters by evaluating clustering performance metrics [57,58].

However, implementing adaptive clustering introduces its own challenges. Dynamically determining the optimal number of clusters can increase computational overhead, potentially offsetting the efficiency gains achieved by clustering. Furthermore, the choice of clustering metrics (e.g., silhouette score or Davies–Bouldin index) may affect the results, leading to variability in feature attributions. These trade-offs highlight the need for further exploration of adaptive clustering strategies to balance scalability with interpretability.

In the context of C-SHAP, adaptive clustering aligns well with the method's objectives. By dynamically adjusting the number of clusters, it ensures that the clustering process captures the underlying patterns in the data while minimizing redundancy in the computation of SHAP values. This would enhance the method's practical applicability, especially in scenarios involving highly diverse or large-scale datasets.

8. Limitations

One limitation of this study is the use of a fixed number of clusters across all datasets, regardless of their size. While the datasets used for evaluation are comparable in size, it is evident that applying the same number of clusters to datasets with varying sizes may not yield optimal results. In future work, it would be beneficial to develop a dynamic method for determining the appropriate number of clusters based on the specific size and characteristics of each dataset.

Another limitation of C-SHAP lies in its reliance on clustering, which, while improving computational efficiency, may obscure fine-grained feature interactions that are critical for certain datasets. For instance, in datasets with highly overlapping clusters, feature attribution might lose precision due to the aggregation process. Future studies could employ metrics like clustering fidelity or feature ranking stability to quantify these trade-offs and identify areas where interpretability is affected.

Additionally, although C-SHAP was evaluated on a diverse set of well-known benchmark datasets, the size of these datasets remains relatively moderate. This raises concerns about the scalability of C-SHAP when applied to much larger datasets or those with a vast number of features. Also, the quality of the data and clusters can alter the performance of the proposed method. By taking data points from each cluster, there is a possibility that the effect of missing data may impact the accuracy of the method. Moreover, real-world datasets often contain noise, imbalances, or missing data, which can affect both clustering performance and feature attribution. Future evaluations should include such datasets to assess C-SHAP's robustness under diverse conditions and validate its generalizability across different domains.

9. Conclusions and Future Works

In conclusion, C-SHAP offers a significant advancement in feature interpretability for machine learning models by addressing the computational limitations associated with traditional SHAP. By leveraging K-means clustering, C-SHAP consistently reduces execution times without sacrificing the accuracy of feature selection. For instance, in the Heart Disease dataset, C-SHAP decreased the Random Forest execution time from over 400 s to just 0.29 s while maintaining consistent results with SHAP in identifying influential

features. This improvement highlights C-SHAP's potential as an efficient interpretability tool across a variety of datasets and machine learning tasks, making it a viable solution for real-time applications.

This study also underscores the need for scalable interpretability approaches in fields with large, complex datasets, aligning with the goals of emerging research at the intersection of machine learning and specialized domains, such as software product-line engineering. While our study focuses on SHAP and C-SHAP, incorporating widespread strategies like saliency maps into future work could provide comparative insights and extend interpretability to new domains.

In future work, we aim to implement C-SHAP specifically in the software engineering domain, exploring its effectiveness in analyzing software engineering datasets and interpreting model decisions within this specialized field. This direction could further validate C-SHAP's adaptability and expand its practical applications.

Future work will address the current limitations of C-SHAP by exploring dynamic clustering methods that adapt to dataset complexity and size. Techniques such as hierarchical clustering or density-based methods (e.g., DBSCAN) could offer greater flexibility and improve clustering fidelity in high-dimensional datasets. Moreover, metrics like clustering stability, feature attribution robustness, and fidelity in aggregated SHAP values will be incorporated to evaluate trade-offs between computational efficiency and interpretability. These improvements aim to enhance the adaptability and scalability of C-SHAP across diverse real-world applications.

Author Contributions: Conceptualization, G.R.; methodology, G.R. and D.R.R.; software, G.R.; validation, G.R. and D.R.R.; formal analysis, G.R. and D.R.R.; investigation, G.R.; resources, G.R.; data curation, G.R.; writing—original draft preparation, G.R.; writing—review and editing, D.R.R. and C.K.R.; visualization, G.R. and D.R.R.; supervision, D.R.R. and C.K.R.; project administration, C.K.R.; funding acquisition, C.K.R. and K.A.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data will be made available on request.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Burrell, J. How the machine 'thinks': Understanding opacity in machine learning algorithms. *Big Data Soc.* **2016**. [CrossRef]
2. Castelvecchi, D. Can we open the black box of AI? *Nat. News* **2016**, *538*, 20. [CrossRef] [PubMed]
3. Villamizar, H.; Kalinowski, M.; Lopes, H.; Mendez, D. Identifying concerns when specifying machine learning-enabled systems: A perspective-based approach. *J. Syst. Softw.* **2024**, *213*, 112053. [CrossRef]
4. Le, T.T.H.; Kim, H.; Kang, H.; Kim, H. Classification and explanation for intrusion detection system based on ensemble trees and SHAP method. *Sensors* **2022**, *22*, 1154. [CrossRef]
5. Luo, Y.; Tseng, H.H.; Cui, S.; Wei, L.; Ten Haken, R.K.; El Naqa, I. Balancing accuracy and interpretability of machine learning approaches for radiation treatment outcomes modeling. *BJR Open* **2019**, *1*, 20190021. [CrossRef]
6. Stiglic, G.; Kocbek, P.; Fijacko, N.; Zitnik, M.; Verbert, K.; Cilar, L. Interpretability of machine learning-based prediction models in healthcare. *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.* **2020**, *10*, e1379. [CrossRef]
7. Doshi-Velez, F.; Kim, B. Towards a rigorous science of interpretable machine learning. *arXiv* **2017**, arXiv:1702.08608.
8. Adadi, A.; Berrada, M. Peeking inside the black-box: A survey on explainable artificial intelligence (XAI). *IEEE Access* **2018**, *6*, 52138–52160. [CrossRef]

9. Carvalho, D.V.; Pereira, E.M.; Cardoso, J.S. Machine learning interpretability: A survey on methods and metrics. *Electronics* **2019**, *8*, 832. [CrossRef]
10. Lipton, Z.C. The mythos of model interpretability: In machine learning, the concept of interpretability is both important and slippery. *Queue* **2018**, *16*, 31–57. [CrossRef]
11. Mukhtar, A.; Hofer, B.; Jannach, D.; Wotawa, F. Explaining software fault predictions to spreadsheet users. *J. Syst. Softw.* **2023**, *201*, 111676. [CrossRef]
12. Debjit, K.; Islam, M.S.; Rahman, M.A.; Pinki, F.T.; Nath, R.D.; Al-Ahmadi, S.; Hossain, M.S.; Mumenin, K.M.; Awal, M.A. An improved machine-learning approach for COVID-19 prediction using Harris Hawks optimization and feature analysis using SHAP. *Diagnostics* **2022**, *12*, 1023. [CrossRef] [PubMed]
13. ElShawi, R.; Sherif, Y.; Al-Mallah, M.; Sakr, S. Interpretability in healthcare: A comparative study of local machine learning interpretability techniques. *Comput. Intell.* **2021**, *37*, 1633–1650. [CrossRef]
14. Marcílio, W.E.; Eler, D.M. From explanations to feature selection: Assessing SHAP values as feature selection mechanism. In Proceedings of the 2020 33rd SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI), Porto de Galinhas, Brazil, 7–10 November 2020; pp. 340–347.
15. Tüzün, E.; Tekinerdogan, B.; Kalender, M.E.; Bilgen, S. Empirical evaluation of a decision support model for adopting software product line engineering. *Inf. Softw. Technol.* **2015**, *60*, 77–101. [CrossRef]
16. Samir, M.; Sherief, N.; Abdelmoez, W. Improving bug assignment and developer allocation in software engineering through interpretable machine learning models. *Computers* **2023**, *12*, 128. [CrossRef]
17. Wang, S.; Huang, L.; Gao, A.; Ge, J.; Zhang, T.; Feng, H.; Satyarth, I.; Li, M.; Zhang, H.; Ng, V. Machine/deep learning for software engineering: A systematic literature review. *IEEE Trans. Softw. Eng.* **2022**, *49*, 1188–1231. [CrossRef]
18. Kamal, M.S.; Nimmy, S.F.; Dey, N. Interpretable Code Summarization. *IEEE Trans. Reliab.* 2024, *early access*. [CrossRef]
19. Dam, H.K.; Tran, T.; Ghose, A. Explainable software analytics. In Proceedings of the 40th International Conference on Software Engineering: New Ideas and Emerging Results, Gothenburg, Sweden, 27 May–3 June 2018; pp. 53–56. [CrossRef]
20. Tantithamthavorn, C.K.; Jiarpakdee, J. Explainable ai for software engineering. In Proceedings of the 2021 36th IEEE/ACM International Conference on Automated Software Engineering (ASE), Melbourne, Australia, 15–19 November 2021; pp. 1–2. [CrossRef]
21. Lundberg, S. A unified approach to interpreting model predictions. *arXiv* **2017**, arXiv:1705.07874.
22. Ribeiro, M.T.; Singh, S.; Guestrin, C. "Why should i trust you?" Explaining the predictions of any classifier. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 13–17 August 2016; pp. 1135–1144. [CrossRef]
23. Molnar, C.; Casalicchio, G.; Bischl, B. Interpretable machine learning—A brief history, state-of-the-art and challenges. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*; Springer: Berlin/Heidelberg, Germany, 2020; pp. 417–431.
24. Merrick, L.; Taly, A. The explanation game: Explaining machine learning models using shapley values. In Proceedings of the Machine Learning and Knowledge Extraction: 4th IFIP TC 5, TC 12, WG 8.4, WG 8.9, WG 12.9 International Cross-Domain Conference, CD-MAKE 2020, Dublin, Ireland, 25–28 August 2020; Proceedings 4; Springer: Berlin/Heidelberg, Germany, 2020; pp. 17–38.
25. Zhang, Y.; Tiňo, P.; Leonardis, A.; Tang, K. A survey on neural network interpretability. *IEEE Trans. Emerg. Top. Comput. Intell.* **2021**, *5*, 726–742. [CrossRef]
26. Sundararajan, M.; Najmi, A. The many Shapley values for model explanation. In Proceedings of the International Conference on Machine Learning, PMLR, Online, 13–18 July 2020; pp. 9269–9278.
27. Meddage, P.; Ekanayake, I.; Perera, U.S.; Azamathulla, H.M.; Md Said, M.A.; Rathnayake, U. Interpretation of machine-learning-based (black-box) wind pressure predictions for low-rise gable-roofed buildings using Shapley additive explanations (SHAP). *Buildings* **2022**, *12*, 734. [CrossRef]
28. Arrieta, A.B.; Díaz-Rodríguez, N.; Del Ser, J.; Bennetot, A.; Tabik, S.; Barbado, A.; García, S.; Gil-López, S.; Molina, D.; Benjamins, R.; et al. Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Inf. Fusion* **2020**, *58*, 82–115. [CrossRef]
29. Rudin, C. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nat. Mach. Intell.* **2019**, *1*, 206–215. [CrossRef] [PubMed]
30. Mo, B.Y.; Nuannimnoi, S.; Baskoro, A.; Khan, A.; Ariesta Dwi Pratiwi, J.; Huang, C.Y. ClusteredSHAP: Faster GradientExplainer based on K-means Clustering and Selections of Gradients in Explaining 12-Lead ECG Classification Model. In Proceedings of the 13th International Conference on Advances in Information Technology, Bangkok, Thailand, 6–9 December 2023; pp. 1–8. [CrossRef]

31. Gramegna, A.; Giudici, P. SHAP and LIME: An evaluation of discriminative power in credit risk. *Front. Artif. Intell.* **2021**, *4*, 752558. [CrossRef] [PubMed]
32. Garreau, D.; Luxburg, U. Explaining the explainer: A first theoretical analysis of LIME. In Proceedings of the International Conference on Artificial Intelligence and Statistics, PMLR, Online, 26–28 August 2020; pp. 1287–1296.
33. Aldughayfiq, B.; Ashfaq, F.; Jhanjhi, N.; Humayun, M. Explainable AI for retinoblastoma diagnosis: Interpreting deep learning models with LIME and SHAP. *Diagnostics* **2023**, *13*, 1932. [CrossRef] [PubMed]
34. Slack, D.; Hilgard, S.; Jia, E.; Singh, S.; Lakkaraju, H. Fooling lime and shap: Adversarial attacks on post hoc explanation methods. In Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society, New York, NY, USA, 7–8 February 2020; pp. 180–186. [CrossRef]
35. Chen, R.C.; Dewi, C.; Huang, S.W.; Caraka, R.E. Selecting critical features for data classification based on machine learning methods. *J. Big Data* **2020**, *7*, 52. [CrossRef]
36. Ismi, D.P.; Panchoo, S.; Murinto, M. K-means clustering based filter feature selection on high dimensional data. *Int. J. Adv. Intell. Inform.* **2016**, *2*, 38–45. [CrossRef]
37. Roshan, K.; Zafar, A. Using kernel shap xai method to optimize the network anomaly detection model. In Proceedings of the 2022 9th International Conference on Computing for Sustainable Global Development (INDIACom), New Delhi, India, 23–25 March 2022; pp. 74–80. [CrossRef]
38. Zheng, W.; Shen, T.; Chen, X.; Deng, P. Interpretability application of the Just-in-Time software defect prediction model. *J. Syst. Softw.* **2022**, *188*, 111245. [CrossRef]
39. Guyon, I.; Elisseeff, A. An introduction to variable and feature selection. *J. Mach. Learn. Res.* **2003**, *3*, 1157–1182.
40. John, G.H.; Kohavi, R.; Pfleger, K. Irrelevant features and the subset selection problem. In *Machine Learning Proceedings 1994*; Elsevier: Amsterdam, The Netherlands, 1994; pp. 121–129. [CrossRef]
41. Hakkoum, H.; Idri, A.; Abnane, I. Global and local interpretability techniques of supervised machine learning black box models for numerical medical data. *Eng. Appl. Artif. Intell.* **2024**, *131*, 107829. [CrossRef]
42. Shapley, L.S. Stochastic games. *Proc. Natl. Acad. Sci. USA* **1953**, *39*, 1095–1100. [CrossRef]
43. Lundberg, S.M.; Erion, G.; Chen, H.; DeGrave, A.; Prutkin, J.M.; Nair, B.; Katz, R.; Himmelfarb, J.; Bansal, N.; Lee, S.I. From local explanations to global understanding with explainable AI for trees. *Nat. Mach. Intell.* **2020**, *2*, 56–67. [CrossRef]
44. Zhang, Q.s.; Zhu, S.C. Visual interpretability for deep learning: A survey. *Front. Inf. Technol. Electron. Eng.* **2018**, *19*, 27–39. [CrossRef]
45. Ghosh, A.; Kandasamy, D. Interpretable artificial intelligence: Why and when. *Am. J. Roentgenol.* **2020**, *214*, 1137–1138. [CrossRef] [PubMed]
46. Shapley, L.S. A value for n-person games. *Contrib. Theory Games* **1953**, *2*. [CrossRef]
47. Parisineni, S.R.A.; Pal, M. Enhancing trust and interpretability of complex machine learning models using local interpretable model agnostic shap explanations. *Int. J. Data Sci. Anal.* **2023**, 18, 457–466. [CrossRef]
48. Liu, S.; Wu, K.; Jiang, C.; Huang, B.; Ma, D. Financial time-series forecasting: Towards synergizing performance and interpretability within a hybrid machine learning approach. *arXiv* **2023**, arXiv:2401.00534.
49. Chudasama, Y.; Purohit, D.; Rohde, P.; Gercke, J.; Vidal, M. InterpretME: A tool for interpretations of machine learning models over knowledge graphs. *Semant. Web* **2023**, 1–21. [CrossRef]
50. Schmidt, P.; Biessmann, F. Quantifying interpretability and trust in machine learning systems. *arXiv* **2019**, arXiv:1901.08558.
51. Vishwarupe, V.; Joshi, P.M.; Mathias, N.; Maheshwari, S.; Mhaisalkar, S.; Pawar, V. Explainable AI and interpretable machine learning: A case study in perspective. *Procedia Comput. Sci.* **2022**, *204*, 869–876. [CrossRef]
52. Covert, I.; Lundberg, S.M.; Lee, S.I. Understanding global feature contributions with additive importance measures. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 17212–17223.
53. Amoukou, S.I. Trustworthy Machine Learning: Explainability and Distribution-Free Uncertainty Quantification. Ph.D. Thesis, Université Paris-Saclay, Paris, France, 2023.
54. Bhatt, U.; Xiang, A.; Sharma, S.; Weller, A.; Taly, A.; Jia, Y.; Ghosh, J.; Puri, R.; Moura, J.M.; Eckersley, P. Explainable machine learning in deployment. In Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency, Barcelona, Spain, 27–30 January 2020; pp. 648–657. [CrossRef]
55. Liu, C.; Yin, X.; Huang, D.; Xu, Y.; Li, S.; Yu, C.; Zhang, Y.; Deng, Y. An Interpretable Prediction Model for the Risk of Retinopathy of Prematurity Development Based on Machine Learning and SHapley Additive exPlanations (SHAP) PREPRINT (Version 1) Available at Research Square 2023. Available online: https://doi.org/10.21203/rs.3.rs-3569382/v1 (accessed on 14 November 2024).
56. Bizimana, P.C.; Zhang, Z.; Hounye, A.H.; Asim, M.; Hammad, M.; El-Latif, A.A.A. Automated heart disease prediction using improved explainable learning-based technique. In *Neural Computing and Applications*; Springer: Berlin/Heidelberg, Germany, 2024; pp. 1–30.

57. Zhong, Q.M.; Chen, S.Z.; Sun, Z.; Tian, L.C. Fully automatic operational modal analysis method based on statistical rule enhanced adaptive clustering method. *Eng. Struct.* **2023**, *274*, 115216. [CrossRef]
58. Zheng, M.; Gao, P.; Zhang, R.; Li, K.; Wang, X.; Li, H.; Dong, H. End-to-end object detection with adaptive clustering transformer. *arXiv* **2020**, arXiv:2011.09315.