Coalitional Strategies for Efficient Individual Prediction Explanation

Gabriel Ferrettini, Elodie Escriva, Julien Aligon, Jean-Baptiste Excoffier, Chantal Soulé-Dupuy

COALITIONAL STRATEGIES FOR EFFICIENT INDIVIDUAL PREDICTION EXPLANATION

A PREPRINT SUBMITTED TO INFORMATION SYSTEMS FRONTIERS

Gabriel Ferrettini
Université de Toulouse-Capitole,
IRIT, (CNRS/UMR 5505)
gabriel.ferrettini@irit.fr

Elodie Escriva
Kaduceo
elodie.escriva@kaduceo.com

Julien Aligon
Université de Toulouse-Capitole,
IRIT, (CNRS/UMR 5505)
julien.aligon@irit.fr

Jean-Baptiste Excoffier
Kaduceo
jbe@kaduceo.com

Chantal Soulé-Dupuy
Université de Toulouse-Capitole,
IRIT, (CNRS/UMR 5505)
chantal.soule-dupuy@irit.fr

April 5, 2021

ABSTRACT

As Machine Learning (ML) is now widely applied in many domains, in both research and industry, an understanding of what is happening *inside the black box* is becoming a growing demand, especially by non-experts of these models. Several approaches had thus been developed to provide clear insights of a model prediction for a particular observation but at the cost of long computation time or restrictive hypothesis that does not fully take into account interaction between attributes. This paper provides methods based on the detection of relevant groups of attributes -named *coalitions*- influencing a prediction and compares them with the literature. Our results show that these *coalitional* methods are more efficient than existing ones such as SHapley Additive exPlanation (*SHAP*). Computation time is shortened while preserving an acceptable accuracy of individual prediction explanations. Therefore, this enables wider practical use of explanation methods to increase trust between developed ML models, end-users, and whoever impacted by any decision where these models played a role.

***K*eywords** Data analysis · Machine learning · Interpretability · Explainable Artificial Intelligence (XAI) · Prediction explanation.

1 Introduction

The main deterrent to the comprehension of the majority of machine learning models is their ”black box” aspect. Once a model has been trained, it is often not possible to know the exact reasoning behind the classification performed. Of course, some models remain interpretable by nature, as they are simple enough to be understood when looked at code-wise. As an example, a decision tree is represented as a binary tree and thus will be interpretable for a human unless the number of branches becomes too large to be practical. Some other examples of an interpretable model class are linear methods such as linear, logistic, or Cox regression that are based on a simple linear equation that is easily understandable by a human. The ”black box” problem arises when more complex models are used. For those cases, the information necessary to understand directly the model becomes too large to be encompassed for a human. As an extreme example, we can always represent a neural network as a lattice of neurons, but the representation of thousands of nodes and paths would not be useful for a human observer.

Thus, we cannot directly access the information on the internal working of a complex model. Yet it is possible to observe the *effects* of this working, namely, the predictions done by those models. We can consider each prediction made by a model as an additional clue on the way the model functions internally. That the approach is used by a large part of the literature to understand how a model works [1, 2]. In those approaches, two main goals can be

seen: *explaining the model behavior globally*, in order to understand it in the general context, or *explaining the model behavior for a particular prediction*, aiming to highlight that particular decision process.

A problem arises when a domain expert user has to study the behavior of particular dataset instances over a predictive model. For example, a physician wanting to perform a cohort study of his patients may also need an explanation of prediction for every single patient, rather than just a global one. In this case, a global explanation is not enough to give the information needed by the study.

To fulfill that need, past researches studied the possibility of explaining single instance prediction of a model, e.g. [3] and [4]. However, these methods may be specifically designed for a single model type as in [5], which limits their use. Some methods are designed to be applicable for all types of models and are more and more used in the industry [1] but suffer from a lack of precision [6]. Yet overcoming this lack of precision can make their algorithms very long to apply [7]. This means the field of single instance prediction explanation needs to be developed further, so as to make it usable by everyone.

Our work fits this ambition to help a domain expert user get involved in data analysis operations, especially in learning tasks. Therefore, obtaining explanations for predictive models in an efficient manner, whether in terms of accuracy or computation time, is essential. In a previous work [8], we proved the feasibility of lowering the computation time of existing solutions with a limited loss of explanation accuracy. Finding more efficient approximations of these solutions, thanks to the detection of more relevant coalitions of attributes was the objective of our previous work in [9]. This paper extends these proposals by detailing and adding examples for each of the coalitional proposals. A new comparison with one of the most used method of the literature, SHAP [10], is presented. In particular, our experiments offer a complete view of the performances (in time and error scores) between all the methods as well as a better characterization of the groups of attributes generated. A real use case, regarding the SARS-COV2 data, illustrates the relevance of our coalitional explanations.

The paper is organized as follows. Section 2 explores previous works done in the domain of prediction explanation. In particular, the identification of attributes having a significant influence on a model is fundamental. To that end, the automated discovery of groups of linked attributes is an important challenge to overcome. For this purpose, we rely on attribute grouping methods from the literature inspired, notably by feature selection methods. Then, Section 3 describes the base methods used to generate prediction explanations. The extension of our work [8] is proposed in Section 4 to find faster explanation methods. This is achieved through new ways to find groups of attributes for the coalitional method described in Section 2.2. Experiments are presented in Section 5, showing the interest of our methods, compared to the literature, in terms of computation time and their limited impacts on accuracy loss, significantly improving the results of [8]. A particular focus is proposed about the characteristics of the groups of attributes generated by our methods. Then, a qualitative comparison between SHAP and one of our methods are challenged through a real use case and showing the consistency of our approach. Finally, Section 7 concludes the paper by discussing the perspectives of works including the new possibilities opened by our results.

2 Related works

2.1 How to explain a model by using its predictions

Explaining the influence of each attribute of a dataset on the output of a predictive model has been explored largely. A few of the works related to global attribute importance on a model can be seen in [11] [12]. The most recent methods are based on swapping the values of attributes in the dataset and analyzing which swaps affect the trained model predictions the most. The more modifying the values of the attributes affect the predictions, the more this attribute is considered important for the model, as a whole. These methods are often used during feature selection, allowing to opt-out attributes useless for the model. Another approach for understanding a model is described by Helenius et al. in [13]. The authors seek to understand which attributes of the datasets are "linked" to each other, according to the model. In particular, they proceed by randomizing the values of potential groups of attributes: when the predictions vary more than a predetermined threshold, the attributes are linked together.

The problem with a fixed global influence is that predictive models are often not consistent with the whole dataset on which they are trained. These global influences give an insight into the general working of the studied model, but there will often be particular regions of the dataset where the model deviates from this general influence. Thus, there also exists a need for a single instance prediction explanation, showing the user how a particular instance is classified, independently to the rest of the dataset. These methods aim to provide insight into the global influence of

[1] Clinical app that predicts an aggravation risk for a patient hospitalized with Covid-19. Attribute influences are computed with *SHAP*. `https://scorecovid.kaduceo.com/`

each attribute, rather than on their influence on a single prediction.

In the field of single prediction, these global influence methods have served as a basis to [4], using the same randomization technique. Given a single instance, the importance of each attribute is obtained by looking at the evolution of the prediction performance of the model on the instance when all of its values are swapped with other values of the dataset except the value of the attribute being studied. The more the prediction varies with the values swapping, the less the fixed attribute is important for the prediction. This method has the interest of relying on the same principle as the global technique which is largely recognized, but the main caveat is its computational cost needed for explaining the prediction of only one instance, as a large number of new predictions has to be generated for each single instance prediction to explain. Another caveat is this prediction explanation is realized from the point of view of model performance. Meaning that their metric shows which feature improves the performance of the model, rather than which feature the model consider as important for its prediction. If this line of reasoning is interesting for the model explanation field, it does not correspond to our scope, as we are aiming to help users understand how a model works, and not how to improve it.

Another approach can be found in [14], which inspired many of the functions of [15]. The principle here is to determine the smallest change needed in order to change the classification of an instance by the model. It has been integrated into the What-if tool. This tool allows the user to select a particular data point in the dataset, and visualize the nearest point classified differently. This is displayed along with the differences between the two points, thus highlighting what let the two points to be classified differently.

These methods give us an insight into how the model works, as they display the attributes which could put the instance in another class if their value was changed slightly. However, if these methods are interesting for analyzing the important points of a model, this kind of information would be far less useful to a domain expert, as they already require the user to understand the importance of this information and draw conclusions on it by himself.

One of the early explanation methods, found in [6], avoid the drawbacks mentioned above. In this explanation method, the weight of an attribute, on a particular prediction, is estimated by the difference of influence over the model with and without this attribute. This absence of an attribute is simulated by a weighted mean of the predictions of the model with all the possible values of the attribute, weighted by their probability of appearing in the dataset. This is faster than the method of [4] as only one value is randomized. Later, in [3], the possibility of retraining the model entirely without the considered attribute in the dataset is proposed, which consists of an interesting trade-off: the time needed for retraining a model for each attribute of a dataset can be considerable, but once this training has been done, the prediction comparison can become near-instantaneous. These methods are named *SHapley Additive exPlanations* by Lundberg et al. [10], who regrouped a large number of similar methods of explanation, and detailed in Section 3.

[10] highlights several interesting properties about these methods :

- Local precision: The system describes precisely the model in the close vicinity of the explained instance.
- "Missingness": If an attribute is missing for the prediction, the method does not give it a weight, or gives it a weight of zero.
- Consistency: If the explained model changes in a way that makes an attribute more important, or does not change its importance, its attributed weight is not diminished. This property is important, as some of the early prediction explanation methods could have an erratic behavior in some cases, as shown in an example of [10].

This type of prediction explanation is quite interesting, as we are aiming to facilitate the understanding of any machine learning models for users without particular knowledge on data analysis or machine learning. Thus, it is more relevant to focus on the works as [16] or [17], cited as *additive* methods, as they generate a simple set of importance weights for each attribute. This set of weights is easy to interpret, even for someone without expertise in machine learning. Yet, these methods have a major deterrent: their complexity makes them difficult to use for the average user. That is why [10] explored methods to generate explanations faster, but at the cost of very restricting hypotheses, as the independence of each attribute of the dataset, or the linearity of the model, which is not always the case.

LIME is another popular additive interpretability method [18]. However, it suffers from previously described restricting hypotheses as it relies on a surrogate linear model to locally approximate influence of each variable, through multiple random perturbations of the instance to explain. This random nature of *LIME* leads to an high instability, especially compared to *SHAP* [19]. Thus we will focus in this article on Shapley-based interpretability methods.

The ability to explain the prediction of any model thus appears to be a key point for allowing a broader public (non-expert) to access and use machine learning models.

This need led us to consider the diverse explanation systems, developed in the literature, as having a major interest in giving more autonomy to domain experts performing data analysis tasks. Yet, the computational load found in the most generic methods can be a hindrance to their use. In this paper, we seek to propose a prediction explanation method as generic as possible and try lowering its computing time without losing too much information.

2.2 Grouping attributes

In our work, we want to facilitate the generation of prediction explanation, without having to restrict to a given set of models. This paper is the continuity of [8] in which we already established possible methods of simplification and rely on the automatic detection of groups of attributes (especially the *K-complete* method, detailed in Section 3.2.).

For this work, we aim to identify and compare additional methods detecting groups, in order to compare their influence on the efficiency of the simplification method.

The selection of relevant attributes to be grouped can take inspiration from the works in the field of feature selection [20] [21]. In particular, the methods proposed in a dimensional reduction goal seem to reach our scope. Indeed, these methods have to automatically detect interactions between attributes for reducing a potential high dimensionality in a dataset. Thus, two main approaches, feature extraction (mainly the principal component analysis) and filter methods (which measure the relevance of features by their correlations) can be considered. The fact that the principal component analysis (PCA) and the filter methods rely only on information provided by a dataset (independent of the model used in analysis) is a great advantage for our work, in contrast with techniques such as SVM-RFE [22] or FS-P [23], based on a specific model. Indeed, different predictive models can classify differently the same instance. Thus, an explanation on this instance can be different from one model to another, and cannot depend on a selection of influence attributes made by a unique predictive model, such as SVM. The PCA is a largely recognized method to provide new features from sets of correlated attributes. The Correlation-based feature selection (CFS) methods [24] are promising candidates. In particular, the use of a multicollinearity measure by a variance inflation factor (VIF), can provide sets of attributes having linear correlations between them. This avoids calculating collinearity between pairs of attributes, using Pearson's measure, for example. However, the VIF measure is unable to compute non-linear correlations, on the contrary of the Spearman correlation factor. Even if this factor only works between pairs of attributes, the capacity to detect non-linear correlations makes it a good candidate.

3 The $Complete$ method and its approximations from the literature

This section introduces the $Complete$ method considered as the baseline of our work ([8]), computing all possible sub-groups of attribute influences. Then, two approximations of the literature, *K-complete* [8] and *SHAP* [10] are detailed. Their limits are finally discussed, opening a way for new proposals of coalitional methods in Section 4.

3.1 Complete explanation

To answer the problems of interaction between attributes, we propose to take inspiration from the work of [16]. The prediction task is a framework close to the situation called "coalitions", where groups of attributes can influence the prediction of the model. In this context, each attribute cannot be considered as independent, but in all possible attributes combinations. The influence of an attribute is measured according to its importance in each coalition. We can then refer to the coalition games as defined by Shapley in [25]: A coalitional game of N players is defined as a function mapping subsets of players to gains $g : 2^N \mapsto \mathbb{R}$. The parallel can easily be drawn with our situation, where we wish to assess the influence of a given attribute *in every possible coalition of attributes*. We then look at not only the influence of the attribute but also its use in all subsets of attributes. We thus define the *complete influence* of an attribute $a_i \in A$ on the classification of an instance x : given a dataset of instances described along the attributes of A, the *complete* influence of the attribute a_i on the classification of an instance x by the classifier confidence function f on the class C is dependant on the influence of all the possibles subgroups $A' \subseteq A$ which does not contain a_i. Thus, the *complete* influence of a_i is :

$$\mathcal{I}^C_{a_i}(x) = \sum_{A' \subseteq A \setminus a_i} p(A', A) * (inf^C_{f,(A' \cup a_i)}(x) - inf^C_{f,A'}(x)) \tag{1}$$

With $p(A', A)$ a penalty function accounting for the size of the subset A'. Indeed, if an attribute changes a lot the result of a classifier, in a large group of attributes, it can be considered as very influential compared to the others. On the opposite, an attribute changing the result of a classifier, whereas this classifier is based on a few attributes, cannot

be considered to have a decisive influence. The Shapley value [25] is a promising candidate and defines this penalty as:

$$p(A', A) = \frac{|A'|! * (|A| - |A'| - 1)!}{|A|!} \tag{2}$$

This *complete influence* of an attribute now takes into consideration its importance among all the possible attribute configurations, which is closer to the original intuition behind attributes' influence. However, computing the *complete influence* of a single instance is extremely computationally expensive, with complexity in $\bigcirc(2^n * l(n, x))$, with n the number of attributes, x the number of instances in the dataset, and $l(n, x)$ the complexity of training the model to be explained. It is then not practical to use the *complete influence*. Consequently, it becomes necessary to seek a more efficient way to explain predictions. Although the *complete influence* is too computationally heavy, it can be considered as an excellent baseline [16]. Thus, we can evaluate other explanation methods by studying their differences with the *complete influence*.

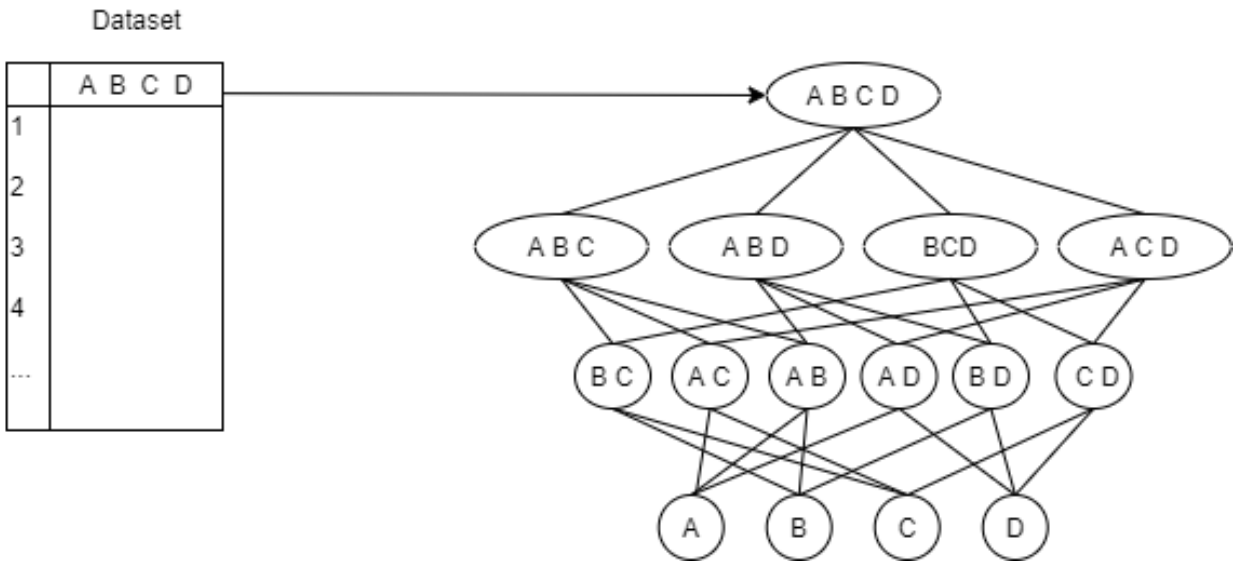

Figure 1: Depiction of the groups calculated by the complete method for a dataset with 4 attributes. Each possible combination of attributes is calculated to ensure an influence value as close to the reality as possible.

Example 1. As depicted in Figure 1, the influence of an attribute depends on its influence alone, but also on each possible groups of attributes containing it. As in the Figure 1, for a dataset with 4 attributes $A\ B\ C\ D$, the influence of the attribute A is composed of the influence of $\{A\}$ alone, along with the influences of the groups $\{A, B\}$, $\{A, C\}$, $\{A, D\}$, $\{A, B, C\}$, $\{A, B, D\}$, $\{A, C, D\}$ and $\{A, B, C, D\}$.

3.2 K-complete explanation

Another possible approach is proposed in our previous work [8]. This approximation, looking for a subset of all the subgroups of the $complete$ method, could be more practical in terms of complexity. This solution should produce explanation, a priori, more accurate than the basic consideration of independent attributes (*linear influence*). We consider then the *depth-k complete influence* defined as the complete influence, but ignoring the groups of attributes A' with a size superior to k :

$$\mathcal{I}_{a_i}^{C_k}(x) = \sum_{A' \subseteq A \setminus a_i,\ |A'| < k} p_k(A', A) * (inf_{f,(A' \cup a_i)}^{C}(x) - inf_{f,A'}^{C}(x)) \tag{3}$$

$$p_k(A', A) = \frac{|A'|! * (|A| - |A'| - 1)!}{k * (|A| - 1)!} \tag{4}$$

In particular, we can note that the *linear* influence is actually identical to the *depth*-1 *complete influence*. The intuition behind this approach is to eliminate the larger groups, which have a lesser impact on the Shapley value while being the most costly to calculate.

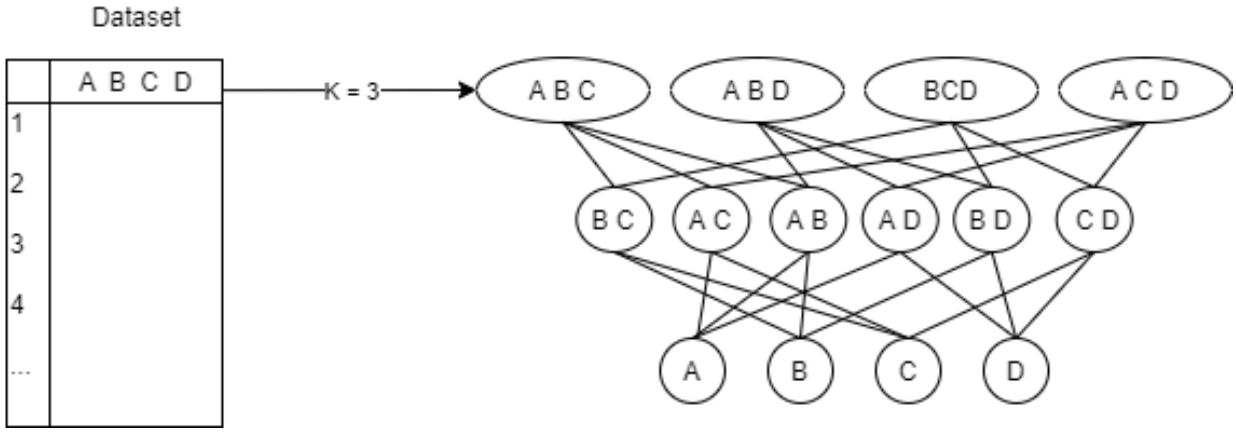

Figure 2: Depiction of the groups calculated by the k-complete method for a 4 attributes dataset. The group size is limited by the parameter k: here the groups maximum size is 3.

Example 2. As depicted in 2, for the same dataset with 4 attributes and a parameter $k = 3$, the total influence of the attribute A only depends on the influence of A alone and the groups of attributes containing A and with a maximum size of 3 : $\{A, B\}$, $\{A, C\}$, $\{A, D\}$, $\{A, B, C\}$, $\{A, B, D\}$ and $\{A, C, D\}$.

3.3 SHapley Additive exPlanation

As explained in Section 2.1, a major contribution in the literature about single prediction explanation can be found in [10]. This approach is one of the most used in the context of prediction explanation, especially in the biology and medical fields like in [26] and [27–29]. [10] theorizes a category of explanation methods, named *SHapley Additive exPlanations* methods, and produces an interesting review of the different methods developed in this category. The authors first propose to overcome a growing problem caused by various proposals of explanation methods: when is it preferable to use one of these methods rather than another one?

From this observation, the paper describes a unified approach for 6 existing methods [30], [31], [32], [33], [16] and [17]. They are summarized in [10] as methods giving for a particular prediction a weight to each attribute of the dataset.

This creates a very simple "predictive model", locally mimicking the original model's behavior. Thus, we have a simple interpretable linear model that gives information on the original model's inner working in a small vicinity of the predicted instance. The methods, from which these weights are affected to each attribute, vary between the different *additive* methods, but the end result is always this vector of weights.

In particular, this unified approach is based on a *SHAP* (SHapley Additive exPlanation) value giving the importance of a feature depending on the predictive model used (for a specific prediction): *LinearSHAP* for linear models, *TreeSHAP* for tree-based models and *DeepSHAP* for neural networks. A *KernekSHAP* is also presented and can be used for any type of predictive model but at the cost of a longer computation time.

Despite this interesting strategy, this approach is very time-consuming, as showed in [7].

The full implementation of this work can be found here: `https://github.com/slundberg/shap`

3.4 Limitations of these approaches

All of the three approaches described in this section have several limitations. *Complete* method has an exponential complexity with respect to the attribute number which makes it unusable in most practical cases. It can be approximated by *SHAP* methods -*KernelSHAP* and its variants- but at the cost of very restrictive hypothesis such as local linearity that does not fully account of dependence between attributes, thus biasing the explanation results. Moreover, the computation may take a very long time in such a configuration of high attribute interdependence, which is often the case in practice. The *k-complete* methods is another way to approximate the *complete* one that gives the ability to select complexity with the *k* parameter. The inconvenience of this set of methods is that these generated groups may include useless or redundant subgroups that greatly increases computation time without any significant gain in accuracy for the *complete*.

4 Proposals of coalition computing methods

Limitations of current approximation methods of the *complete* highlighted in the previous section indicate that potential interactions between attributes must be better taken into account. Combination of unrelated attributes should be avoided at maximum to minimize the complexity, thus computation time, while staying at high accuracy with respect to the *complete* method. In this end, we propose several grouping methods such as an existing algorithm from [34], and new algorithms based on *Principal Component Analysis* (PCA), *Spearman correlation factor* (Spearman) and *Variance Inflation Factor* (VIF). We also develop *Reverse* methods -based on either Spearman or VIF- that only gather uncorrelated attributes, since groups only formed of highly correlated attributes contain mostly redundant information. For each algorithm, a parameter controls the size of the generated subgroups. A higher value of this parameter generates larger groups whereas a smaller value produces smaller, thus less complex, groups. Explanations through influence for each attribute of the dataset is then computed using *coalitional* influence, which takes as parameter the list of groups generated by a grouping method.

4.1 Coalitional explanation methods

The coalitional explanations, presented in this Section 4, identify the attributes having an interaction between them. We can obtain a grouping such as $G = \{\{a_1, a_3\}, \{a_2, a_5, a_8\}, \{a_4\}...\}$. With such groupings of attributes, it becomes possible to consider only the attributes of a subgroup, without having to consider every possible attribute combination. It is important to note that the groups do not necessarily have to be exclusive, which mean an attribute a_i can be found in multiples groups of G. We then obtain a *coalitional influence* of an attribute a_i : Given G_{a_i}, the subset of G containing all the attributes groups $g \in G$ such as $a_i \in g$

$$simple\mathcal{I}^C_{a_i}(x) = \sum_{g' \subseteq g \setminus a_i,\ g \in G_{a_i}} p_c(g', g, G_{a_i}) * (inf^C_{f,(g' \cup a_i)}(x) - inf^C_{f,g'}(x)) \tag{5}$$

$$p_c(g', g, G_{a_i}) = \frac{|g'|! * (|g| - |g'| - 1)!}{\sum_{g \in G_{a_i}} |g|!} \tag{6}$$

Given the fact that we can set a maximum cardinal c for our subgroups, the complexity is now, in the worst case, $O(2^c * \frac{n}{c} * l(n, x)) \approx O(n * l(n, x))$. This method calculates less groups than the *depth-k complete influence*, but tries to make up for it by only grouping the attributes actually related to each other. In order to determine which attributes seem to be related, several types of coalition strategies are proposed below.

4.1.1 Model-based coalition

Regarding the *Model-based coalition* approach, the groups of attributes are created by using the model itself to detect interacting attributes. In this approach, no correlation is detected, but only interaction in the sense of the model usage of the attributes. This is done by randomizing the values of the dataset and studying the evolution of the model predictions. It consists of measuring the differences of predictions on the whole dataset before and after the randomization. When attributes are considered to be part of the same group, their values are swapped together with the values of another instance, classified by the model as the same class as the starting instance. Each attribute outside of the group has its value swapped completely randomly. Once this has been done, the new instances are classified by the model. The ratio of differences between the old and the new classification is called fidelity. A higher fidelity meaning a lower variation of the predictions. At each iteration, the attribute which removal lowers the less the fidelity is removed until it is not possible to keep the fidelity above a fixed threshold. Then the group is considered as fixed. This attribute grouping algorithm has been developed in [34] and is detailed in Algorithm 1.

Example 1. Given the same dataset with 4 attributes, we apply the algorithm from [34] on the dataset. It aims to build groups as small as possible such as when:

- The values of the grouped attributes are randomized **inside** of their original instance class ;
- The values of the non-grouped attributes are randomized completely ;
- This randomization is applied to the whole dataset.

The predictions of the model for the whole dataset do not vary more than a threshold percentage of δ.

In the first iteration, the algorithm finds that the smallest group of attributes that makes the predictions varies less than the threshold δ is $\{B, C\}$. Removing C or B makes the predictions vary more than the threshold and as such, the algorithm stores $\{B, C\}$ as a first group.

Algorithm 1 Model-based coalition extraction.

Input: Sensitivity parameter $\delta > 0$, the number of attributes m, and a fidelity function $fid()$. Two auxiliary functions $L(X) = \bigcup_{i \in X}\{\{i\}\}$ and $F(X) = L(\bigcup_{Y \in X} Y)$, which produces sets of singletons (e.g. $L(\{1,2,3\}) = F(\{\{1,2\},\{3\}\} = \{\{1\},\{2\},\{3\}\})$

Output: σ a coalition of attributes

$\sigma \leftarrow \{\}$
$R \leftarrow \{m\}$ ▷ R contains a group to test for
$A \leftarrow \{\}$ ▷ A contains the removed attributes
$\Delta \leftarrow fid(L([m])) + \delta$
while $R \neq \{\} or A \neq \{\}$ **do**
  if $A = \{\}$ and $fid(\{R\} \bigcup F(\sigma)) < \Delta$ **then**
    ▷ if we are already below Δ before removing any attribute assign the remaining attributes to singleton groups
    $\sigma \leftarrow \sigma \bigcup L(R)$
    $R \leftarrow \{\}$
    $A \leftarrow \{\}$
  else
    ▷ Find an attribute j whose removal from R decreases the fidelity least
    $j \leftarrow argmax_{j \in R} fid(\{\{R \backslash \{j\}\} \bigcup \{\{j\}\} \bigcup \{A\} \bigcup F(\sigma))$
    if $|R| = 1$ or $fid(\{\{R \backslash \{j\}\} \bigcup \{\{j\}\} \bigcup \{A\} \bigcup F(\sigma))$ **then**
      ▷ If the fidelity drops below Δ add the group of attributes to the results and look for the next group of attributes
      $\sigma \leftarrow \sigma \bigcup \{R\}$
      $R \leftarrow A$
      $A \leftarrow \{\}$
    else
      ▷ If the fidelity stays above Δ continue removing the grouping R
      $R \leftarrow R \backslash \{j\}$
      $A \leftarrow A \bigcup \{j\}$
    end if
  end if
end while
return σ

Now the algorithm tries to build another such group with the remaining non grouped attributes and find that $\{A, D\}$ makes the predictions vary too much. Since the biggest remaining possible group is already making the predictions vary more than the threshold, all the non grouped attributes are considered as singletons, resulting in the grouping $\{\{A\},\{B,C\},\{D\}\}$.

This grouping is then used to determine how each attribute has its influence calculated. As an example, the total influence of A only consists of the influence of the singleton $\{A\}$, while the total influence of B is composed of the influences of $\{B\}$ and $\{B,C\}$. Similarly, the total influence of C includes the influences of $\{C\}$ and $\{B,C\}$, and as D is in a singleton, its influence only takes into account the influence of $\{D\}$. Those groups are depicted in Figure 3.

4.2 *PCA* based coalition

The main principle of a Principal Component Analysis (PCA) is to reduce a dataset to its simplest expression in terms of attributes. In other words, if the dataset is considered a multidimensional matrix, the PCA aims to reduce its dimensionality as much as possible. To do that, the different attributes of the dataset are combined linearly, the result being a new set of attributes, each new attribute being a linear combination of the previous ones.

Our reasoning, for this approach, is to consider the set of combined attributes (summarized by the new attribute of the PCA) as a group of influence.

Given a dataset $D = (A, X)$ composed of a set of n attributes $A = \{a_1, ..., a_n\}$, and a set of instances X where $x \in X, x = \{x_1, ..., x_n\} \forall i \in [1..n], x_i \in a_i$.

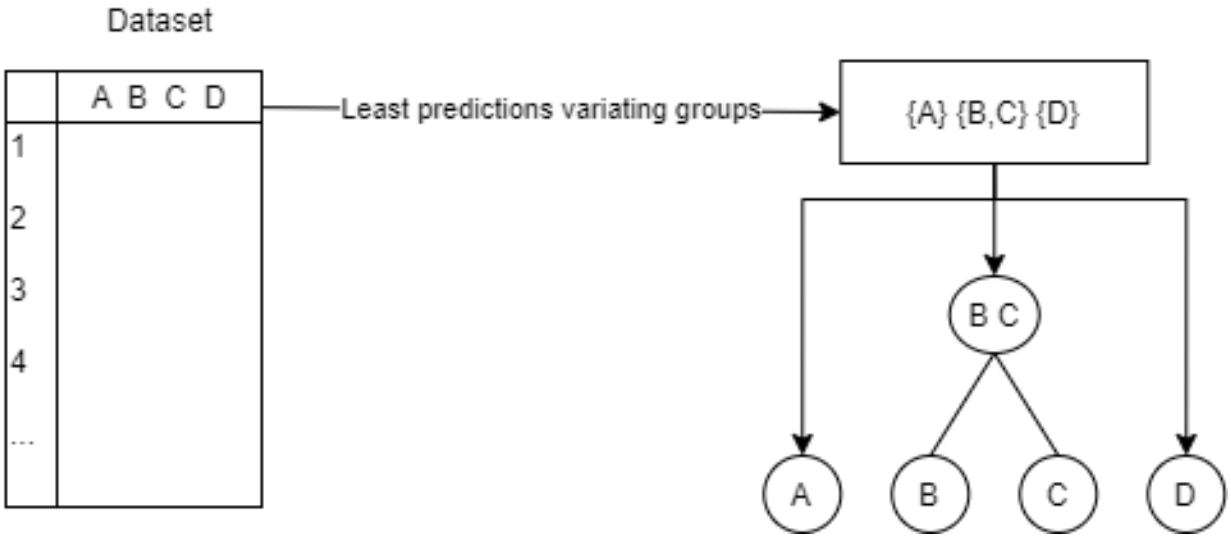

Figure 3: Depiction of the groups calculated by the model-based coalition method for a 4 attributes dataset.

We can apply a principal component analysis which produces a new dataset $D' = (A', X')$ such as $A' = \{a'_1, ..., a'_m\}$ with each new attribute being a linear composition of the previous attributes : $\forall i, a'_i \in A', \exists \{\alpha_1, ..., \alpha_n\} \in R^n, a'_i = \alpha_1 * a_1 + ... + \alpha_n + a_n$.

Each new instance is associated with an instance of the previous dataset. $\forall x' = \{x'_1, ..., x'_m\} \in X', \exists!\ x \in X, \forall i \in [1, .., m] \exists\ \alpha_1, ..., \alpha_n \in R^n, x'_i = \alpha_1 * x_1 + ... + \alpha_n + x_n$.

Given this set of factors $\alpha_1, ..., \alpha_n$, for each attribute, we consider each factor as an evaluation of the importance of the attributes in the group. We can then constitute a coalition of attributes by exploiting the groups formed by the most important factors. This gives us the algorithm 2. For the sake of simplicity, we consider each $a' \in A'$ as a vector of its α_i factors.

Algorithm 2 PCA-based coalition extraction.

Input: a threshold t and the set of attributes A' of the PCA
Output: σ a coalition of attributes

```
σ ← {}
for all a' ∈ A' do                          ▷ for each attribute generated by the PCA
    g ← {}                                  ▷ g, a new possible group
    αmax ← max(a' = α_1, ..., α_n)          ▷ find the most important factor
    for all α_i ∈ a' do
        if α_i ≥ α max * (1 − t) then
            add a_i to g                    ▷ the attribute is included in the group if close to the max
        end if
    end for
    add g to σ
end for
return σ
```

Example 2. Given our previous dataset of 4 attributes, we run a PCA on it. The new attributes generated are as in Figure 4. We have two principal components : $A' = 0.65D - 0.4B + 0.25C + 0.1A$ and $B' = 0.5A + 0.3B + 0.1C - 0.1D$. Here, we consider each attribute with the highest associated coefficients as part of a group. So, here, we have two groups: one for A', $\{D, B, C\}$ and one for B', $\{A, B\}$. We then calculate the total influence of each attribute as the combined influences of each attribute alone and each subgroup of the two generated groups containing the attribute. Thus, the total influence of A is composed of the influence of $\{A\}$ and $\{A, B\}$ as no other group of subgroup generated contains A. For the total influence of B, we use the influence of $\{B\}, \{A, B\}, \{B, C\}, \{B, D\}$ and $\{B, C, D\}$. The total influence of C depends on the influences of $\{C\}, \{B, C\}, \{C, D\}$ and $\{B, C, D\}$. Finally, the influence of D is constituted by the influences of $\{D\}, \{B, D\}, \{C, D\}$ and $\{B, C, D\}$.

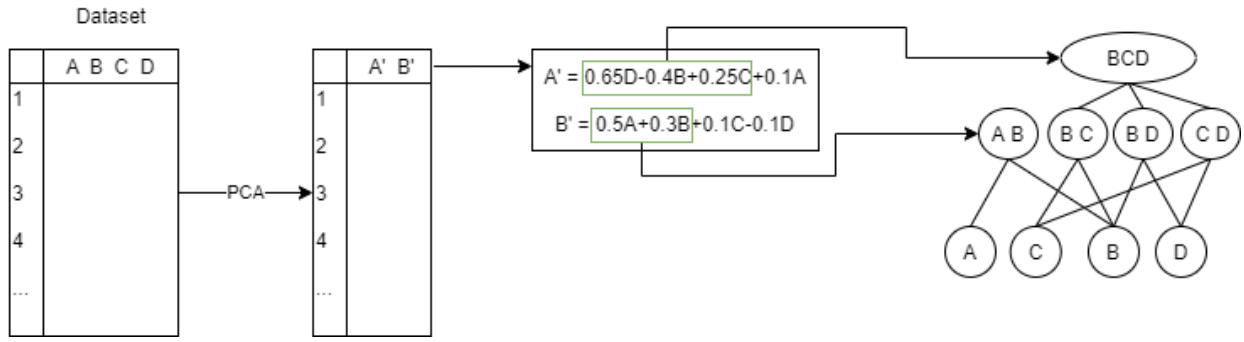

Figure 4: Depiction of the groups calculated by the PCA based coalition method for a dataset of 4 attributes. The new attributes formed by the PCA are a combination of the previous attributes. The attributes with the highest coefficient for each new attribute are considered as part of a group to be calculated.

4.3 *VIF* and *revVIF* based coalition

The variance inflation factor (VIF) is an estimation of the multicollinearity of the attributes of the dataset regarding a given target attribute.

Given a dataset $D = (A, X)$, the VIF value of $a \in A$ is calculated by running a standard linear regression with a as the target for the prediction. Then, given R the coefficient of determination of the linear regression, we have:

$$VIF(a) = \frac{1}{1 - R^2} \tag{7}$$

It is commonly accepted that a variance inflation factor superior to 10 indicates strong multicollinearity of the attribute with other attributes of the dataset. This threshold of 10 is arbitrary but considered as a standard in numerous publications (e.g. [35]). Moreover, when an attribute is removed from the dataset, the VIF of the attributes multicollinear with it decreases. Then, we can automatically detect groups of attributes by calculating the VIF of each attribute (considered as a target) of the dataset, and then comparing them with a new VIF calculation with an attribute removed. For this purpose, we consider two possible approaches:

- Considering as a priority the calculation of strongly multicollinear groups of attributes: Those are groups of attributes with a dependency on one another. In the context of this approach, attributes whose VIF varies strongly when an attribute is removed from the dataset is considered as part of the group.
- Considering as a priority the calculation of weakly or non-multicollinear groups of attributes: Given the fact that correlated attributes tend to bring the same information to the model, it may be preferable to prioritize groups for which the addition or removal of an attribute changes greatly the information brought by the group.

These two approaches are named *VIF coalition* and *reverse VIF coalition*, respectively. This gives us the algorithm 3, for the *VIF coalition*. The *reverse VIF coalition* can be obtained simply by replacing the condition for adding an attribute to a group by $if\ newvifs(a') > oldvifs(a') * (1 - t * 0.05)$. This supplementary ratio of 0.05 has been obtained by preliminary experiments, which showed that just keeping the $1 - t$ factor led to a generation of all the possible subgroups, which defeat the principle of an approximation.

Example 3. Given our dataset of 4 attributes, we calculate the VIFs of each attribute. Then, we calculate the VIFs again each time with one of the attributes removed. The results are depicted in Figure 5. In the case where A is removed, we see that the VIFs of B and C vary greatly. Thus, our first group is $\{A, B, C\}$. Then, when B is removed, only the VIF of A varies a lot. We then have a second group: $\{A, B\}$. Finally, we can see that removing C and D do not make the other VIFs vary in a significant way. Because of that, the attributes C and D are considered as singletons. As the group $\{A, B\}$ is contained by $\{A, B, C\}$, our final coalition is $\{\{A, B, C\}\{D\}\}$. The complete influence of A is constituted of the influences of $\{A\}$, $\{A, B\}$, $\{A, C\}$ and $\{A, B, C\}$. the complete influence of B includes $\{B\}$, $\{A, B\}$, $\{B, C\}$, $\{A, B, C\}$. The complete influence of C contains $\{C\}$, $\{A, C\}$, $\{B, C\}$, $\{A, B, C\}$. Finally, the complete influence of D only contains $\{D\}$.

Algorithm 3 VIF-based coalition extraction.

Input: a threshold t, the set of attributes of the dataset A and a function $VIF(A)$ calculating the array of all the VIF of all the subsets of a set of attributes
Output: σ a coalition of attributes

```
σ ← {}
oldvifs ← VIF(A)                          ▷ calculating the initial VIFs of the attributes
for all a ∈ A do
    g ← {}
    add a to g
    newvifs ← VIF(A/a)
    for all a' ∈ A do
        if newvifs(a') < oldvifs(a') * (0.4 + t) then
            add a' to g
        end if
    end for
    add g to σ
end for
return σ
```

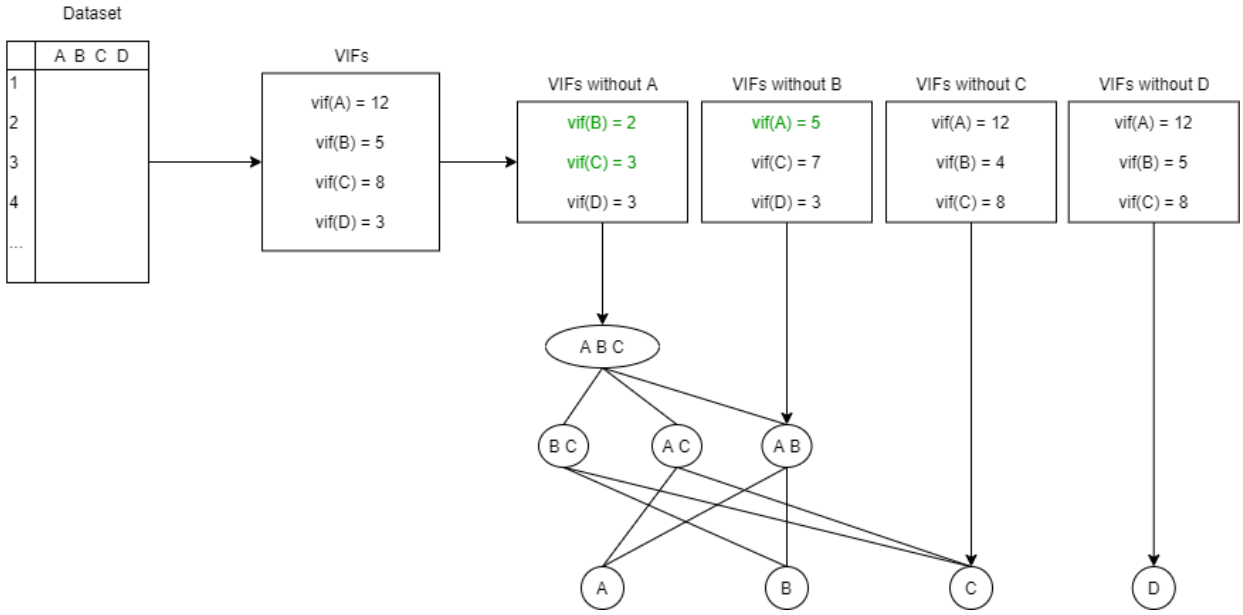

Figure 5: Depiction of the groups calculated by the VIF based coalition method for a 4 attributes dataset. the VIFs are calculated for each attribute and are recalculated with an attribute absent from the dataset. The attributes whose VIFs varies the most are considered as grouped with the removed attribute. If no VIF is changed, the removed attribute is considered as a singleton.

4.4 Spearman correlation based coalition

A limit of the variance inflation factor is the sole consideration of multicollinearity, while a correlation between attributes might not be linear. This problem is addressed through the Spearman correlation coefficient, which takes into account non-linear correlations. Spearman being not multicollinear, the calculation of the correlation between attributes has to be done by pairs. Thus, the method consists of generating the matrix of all the correlations of each pair and then deciding which attributes are part of a group. For this method, we have the same two possibilities as for the *VIF* method: we can either prioritize the calculation of strongly correlated attributes or on the contrary, prioritize groups of non-correlated attributes. These two approaches are named respectively *Spearman coalition* and *reverse Spearman coalition*.

Given a dataset $D = (A, X)$, with $A = \{a_1, ..., a_n\}$ the correlation matrix C is obtained by computing the Spearman correlation coefficient of each attribute couple : $C(1, 2) = corr(a_1, a_2)$. Thus C is symmetrical and have 1 as the

value of its whole diagonal. For each line i of the matrix C, we consider as grouped with a_i the attributes strongly (or weakly) correlated with a_i, for the *Spearman coalition* (or the *reverse Spearman coalition*).

Algorithm 4 Spearman-based coalition extraction.

Input: a threshold t, the set of attributes of the dataset A, and a function $spearman(A)$ calculating the matrix of all the absolute Spearman correlation coefficient of all the subsets of a set of attributes. a max and min functions which returns the maximum and minimum of a matrix line.
Output: σ a coalition of attributes

```
σ ← {}
corrmat ← spearman(A)                                   ▷ calculating the correlation matrix
for all a ∈ A do
    g ← {}
    for all a ∈ A do
        if corrmat(a, a') > max(corrmat(a)) * (1 − t) and max(corrmat(a)) > 0.1 then
                        ▷ If the most correlated attribute have a coefficient less than 0.1, we consider a as a singleton
            add a' to g
        end if
    end for
    add g to σ
end for
return σ
```

The algorithm 4 details the *Spearman coalition* method. The *reverse Spearman coalition* method can be obtained by replacing the condition for adding an attribute to a group by $corrmat(a, a') < min(corrmat(a)) + max(corrmat(a)) * t$ and $min(corrmat(a)) < 0.5$. This adds the least correlated attributes up to a threshold : if the attribute least correlated to a have its Spearman correlation to a superior to 0.5, we consider the attribute a as a singleton.

Example 4. Given our previous dataset of 4 attributes, we calculate the matrix of the spearman correlation coefficients as depicted in Figure 6. In this matrix, we iterate on each row of the matrix, in order to create groups based on the most correlated attributes. In the first line, we see that the attribute most correlated to A is B, and the two other attributes are very weakly correlated to A. Thus, we have a first group: $\{A, B\}$. The second line tells us that A and C are both strongly correlated to B. So, we have a second group: $\{A, B, C\}$. Similarly, the third line indicates that B and D are correlated to C, and so we add a third group: $\{B, C, D\}$. Finally, by looking at the last line we learn that only C is strongly correlated to D and so our last group is $\{C, D\}$. As the two groups of cardinal 2 are contained by the two groups of cardinal 3, we have our final coalitions: {{A,B,C},{B,C,D}}. With this coalition, the complete influence of the attribute A is composed of $\{A\}$, $\{A, B\}$, $\{A, C\}$ and $\{A, B, C\}$. B is composed of $\{B\}$, $\{B, D\}$, $\{A, B\}$, $\{B, C\}$, $\{A, B, C\}$ and $\{B, C, D\}$. Finally, the complete influence of D is composed of $\{D\}$, $\{C, D\}$, $\{B, D\}$ and $\{B, C, D\}$.

5 Evaluation of the additive methods

In this section we aim to evaluate the performances of each coalition calculation method, considering their precision when compared to the *complete* influence, and their computational time. We also give an overview of the group characterization for each coalition method.

5.1 Experimental protocol

Our experiments are run on an AMD Ryzen 3700 processor with 8 x 3.6 GHz cores and 32 GB of RAM. Our tests are realized from the data available on the Openml platform [36]. We select the biggest collection of datasets [2] on which classification tasks have been run. We also consider two classification tasks: Random Forest and Support Vector Machine (SVM) with the non-linear Radial-Basis-Function (RBF) kernel. Experiments are conducted using Python 3.7.9 with the Scikit-Learn package on both models [3]. Due to the heavy computational cost of the complete influence - considered as the reference of our experiments- we select the datasets having at most nine attributes. Thus, a collection of 243 datasets is obtained. Considering the two types of workflows, we have a total of 486 runs. For each of those

[2] Available in `https://www.openml.org/s/107/tasks`

[3] `https://scikit-learn.org/stable/`

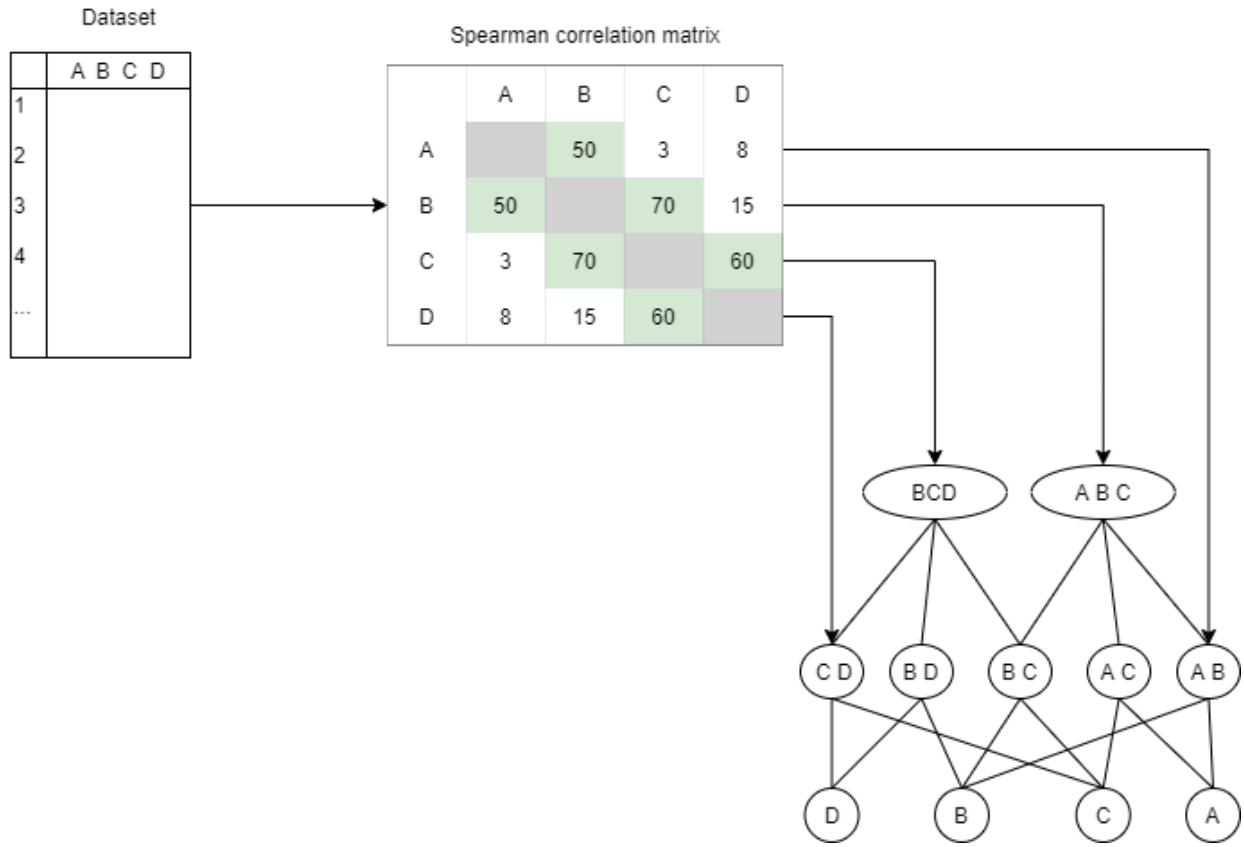

Figure 6: Depiction of the groups calculated by the spearman based coalition method for a 4 attributes dataset. The spearman correlation matrix is calculated. For each line, the attributes most correlated with the line's attribute are considered as part of a group.

runs, we generate each type of influence described in this paper, for each instance of the 243 datasets: the *complete* influence for the baseline, along with the *coalitional* influences, *k-depth* influence, and the Kernel and Tree SHAP influences. The *coalitional* influences are generated using the different group generation methods described in Section 4, which are based on an $\alpha \in]0, 0.5[$ parameter (small values of α resulting in smaller subgroups, and high values in bigger ones). We generate the possible subgroups with 5 different values of α to study the influence of subgroup size. To compare the different explanation methods, we consider the explanation results as a vector of attribute influences noted $\mathcal{I}(x) = [i_1, ..., i_n]$ with n the number of attributes in the dataset. Thus, each of the attributes a_k is given an influence $i_k \in [0, 1]$ by the method $\mathcal{I} : \forall k \in [1..n], i_k = \mathcal{I}_{a_i}(x)$, with x an instance of the dataset. We then define a difference between two vectors of influences i, j as the normalised Euclidean distance:

$$d(i, j) = \frac{1}{2\sqrt{n}} \sum_{k=1}^{n} \sqrt{(i_k - j_k)^2} \tag{8}$$

Considering this formula, we define an error score based on the difference between an explanation method and the *complete* influence method. Given an instance x, an explanation method $\mathcal{I}(x)$, and the *complete influence* method $\mathcal{I}^C(x)$:

$$err(\mathcal{I}, x) = d(\mathcal{I}(x), \mathcal{I}^C(x)) \tag{9}$$

For each instance of each dataset, we generate the error score of every method, allowing us to compare their performances across the different collected datasets. Each error score is the distance of one of the coalitional methods from the *complete* method. Thus, lesser error is indicative of a more precise estimation of the *complete* method.

To compare methods, we also consider the time needed to explain a set of data, called computation time. This includes the time to determine the subgroups of interest and to compute the influences of all instances in the *coalitional* and *k-depth* methods. For the *SHAP* methods, the calculation time is the one taken to calculate the influences of all instances.

Number of attributes	1	2	3	4	5	6	7	8	9
Number of datasets	3	21	44	25	38	26	34	28	24
Mean number of instances	724	736	1688	560	843	600	456	750	479

Table 1: Dataset stats with a given number of attributes

Table 1 details the number of datasets and mean number of instances for each number of attributes. Since the number of instances impacts the total computation time for a dataset, each computational time is normalized by dividing by the number of instances in the dataset to compare times per instance.

5.2 Evaluation of the literature methods

In this section, we focus on the *k-depth* and *SHAP* methods and evaluate them with the protocol described previously. Figure 7 indicates the mean error with respect to the *complete* method, and with the distinction of the datasets based on their number of attributes. For both models, the *linear* (or 1-depth) method gives the worst results, particularly as the number of attributes grows. A larger *k* in *k-depth* methods results in more precise influence attribution. This is fully expected since a higher k generates larger groups closer to the $Complete$ method. Interestingly, *SHAP* methods -*KernelSHAP* for both models and *TreeSHAP* only for the Random Forest- give rather accurate results, without really losing accuracy as the number of attributes grows.

Figure 8 represents the mean computation time per instance for all methods. The y-axis is log scaled in order to have a clear view of all results, even for datasets with a low number of attributes, as the growth is exponential. The *k-depth* methods take more and more time as the *k* grows -from *linear* to *complete*. The *TreeSHAP* method, only usable with the Random Forest, gives rather good computation time, being between those of *linear* and *2-depth* methods. On the contrary, the *KernelSHAP* methods give poor results, being slower than the complete, especially with the SVM model. This is a major inconvenience since it suggests that interpretability with models that are not tree-based would often be intractable in practice.

We then present, in the next section, the evaluations of our coalitional methods, including the first comparison with literature approaches.

5.3 Evaluation of the coalitional methods

The error rate, with respect to the $complete$ method, of the coalitional methods (for several α-thresholds), along with the *linear* method, is shown in Figure 9. All methods give better results with a higher α-threshold since a larger one generates bigger subgroups, thus leading to higher complexity. In particular, $RevVIF$ approach seems to give better results compared to the other ones, for a majority of the α-thresholds. A possible reason is the ability of $RevVIF$ to generates groups of "less correlated" attributes. We can suppose these types of groups better represent the diversity of possible explanations.

Figure 10 shows computation time per instance for each coalitional grouping method and the complete one. All methods except for *Model-based* are notably faster than the *complete* method, especially for *PCA*, *Spearman* or *VIF*. On the contrary, *Model-based* takes mostly more time than the *complete* method. Indeed, the time taken by the *Model-based* method to generate a set of groups is particularly long compared to other coalition-based methods, thus canceling out the effects of group selection. This automatically disqualifies this method for practical use. In particular, the computation time for $RevVIF$ generally seems higher than the other methods. A possible reason is that it should generate larger groups, closer to the $Complete$ one.

To have a clearer view of the methods' performances, we average the error with respect to the $Complete$ method and the computation time globally, thus independently of the number of attributes in the datasets. Therefore, it gives us a single representation, called *Performance Map* (Figures 11 and 12), with the computation time normalized by one of the complete on the horizontal axis, and the error for the complete on the vertical axis. The complete method is thus placed on the point with coordinates $(1, 0)$. All the methods are thus placed above the complete since they can not have a null error with respect to the complete, and methods placed at the left of the complete have lower computation time than the complete, while those placed at the complete right are slower to compute.

We also retained only the most promising grouping methods from previous results -*PCA*, *Spearman*, *Reverse Spearman* and *Reverse VIF*- for $\alpha \in [0.2, 0.4]$ so as to avoid a figure with too much noisy information. The *k-depth* and $SHAP$ methods are also shown.

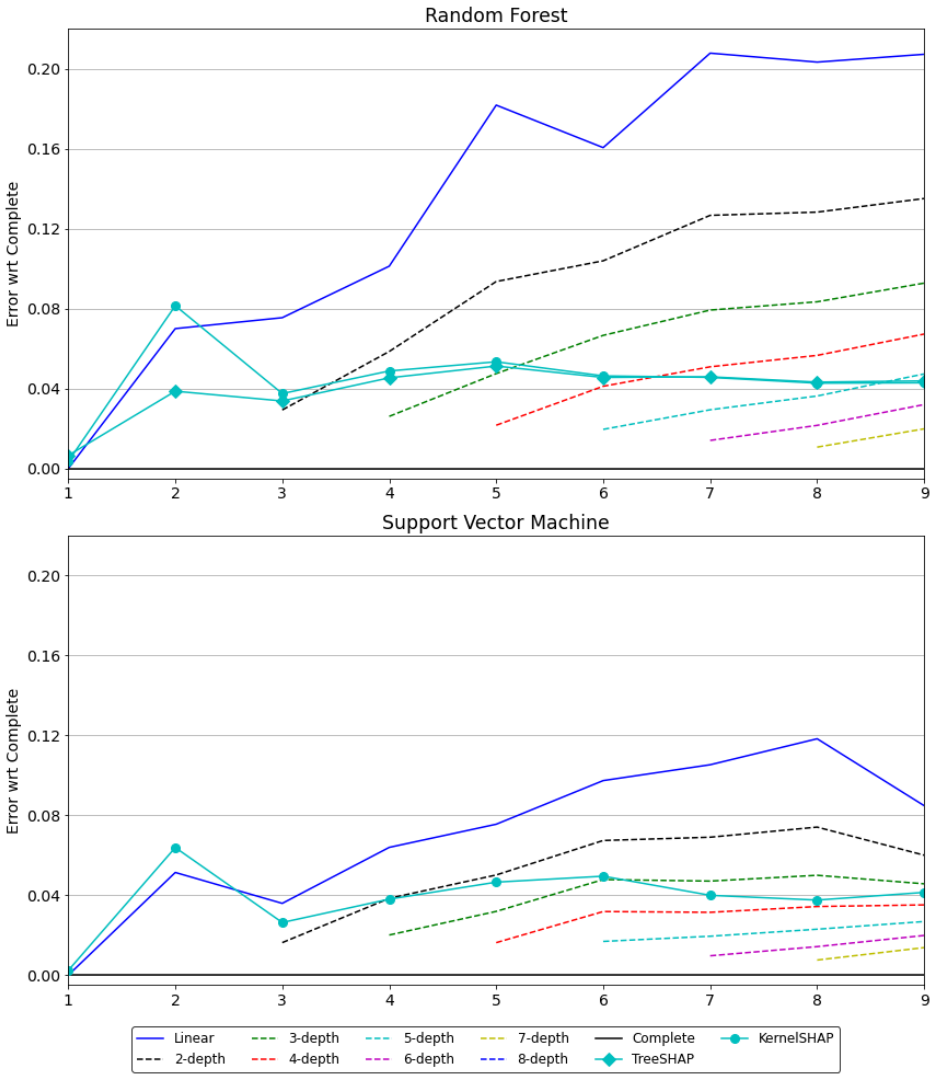

Figure 7: Error score between each explanation method and the *complete influence* depending on the number of attributes in the dataset.

Figure 11 shows the *Performance Map* for the Random Forest model. *KernelSHAP* is not included since its computation time is too high, thus flattening the rest of the graph. Nevertheless, *TreeSHAP* method is still on average slower to compute than the complete, despite being specially designed for tree-based models. As for the coalitional methods, *PCA* is the fastest, but not the most accurate, while *Reverse VIF* is the most accurate but also the slowest method between all coalitional methods. *Spearman* and *Reverse Spearman* seems to be the best balance between precision and computation time.

Figure 12 shows the *Performance Map* for the SVM model. The subfigure on the left includes the *KernelSHAP* method, but as this method is on average 40 times slower than the $Complete$ method for our datasets, it flattens the other ones. Thus the subfigure on the right does not include *KernelSHAP* method. In a similar fashion to Random Forest, *PCA* is the fastest coalitional method but the least accurate, while *Reverse VIF* is the most accurate. Again, *Spearman*-based grouping methods seem to be the most balanced ones.

The only advantage of *SHAP* methods is that one does not need to retrain any model. *SHAP* simulates missing attributes through a heavy number of perturbations of the to-be-explained instance, implying a substantial cost. While our methods, only retraining once the model for every coalitional group, seems to be beneficial over *SHAP*. That is

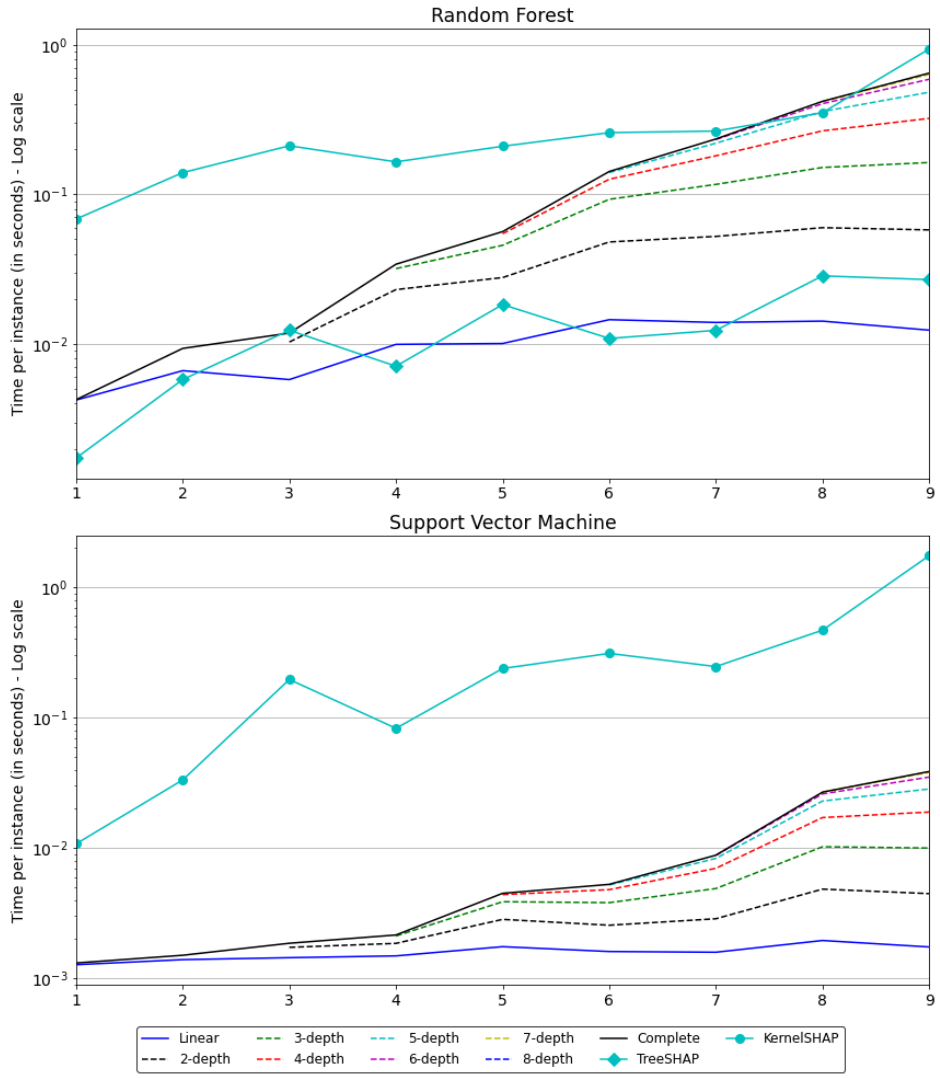

Figure 8: Time per instance (in seconds and log scaled) taken by each method depending on the number of attributes in the dataset.

why the *complete* method, which has the highest number of coalitional groups, is often faster to compute than *SHAP* methods.

The next section gives an overview of the characteristics of the groups generated by our coalitional methods. It justifies the very interesting results obtained by the $Spearman$-based methods, in particular.

5.4 Group characterization of coalitional methods

Figure 13 and 14 compare the average number and average size of the groups of attributes generated by coalitional methods for several α-thresholds. We see that the grouping method and the α-threshold parameter have an important impact on these metrics, thus on the complexity of the groups. *Reverse VIF* generates the highest group sizes on average, which can explain the fact that this method gives the lowest error with respect to the complete, but also the longest computation time as discussed in Section 5.3. On the contrary, *PCA* gives the highest number of groups along with the lowest size of groups explaining the fact that this method is the fastest since it generates a list of groups

closer to the *linear* method than the *complete*.

Figure 15 shows the average complexity of the generated groups with respect to the complexity of the *complete* method which is equal to $2^{nb\,attributes} - 1$. The complexity of a list of groups generated by a grouping method is equal to the number of distinct subgroups. The group complexity is then divided by the *complete* one to get a value between 0 and 100%, the latter indicating that the generated group list is similar to the *complete*, while a low value indicates that the group list is closer to the *linear*, thus less complex and faster to compute. As expected, the *PCA* method generates on average the least complex groups, while the *Reverse* methods, whether based on *Spearman* or *VIF*, generate the most complex ones (particularly true for $RevVIF$) and explaining a higher computation time.

The previous results on the group characterization for coalitional methods show that the groups differ greatly in function of the α-threshold and the grouping method. Figure 16 displays the evolution of the complexity of the groups generated by four coalitional methods for a particular dataset with 7 attributes. The *linear* complexity is thus equal to 7 while the *complete* complexity is $2^7 - 1 = 127$. There is a clear difference between the evolution of the grouping methods confirming that the α-threshold can not be set at the same value for all methods. In this example, if one wants a complexity which equals to 25% of the *complete*, the α-threshold for the *Reverse VIF* method would be equal to about 0.08, whereas for *Spearman* it would be about 0.42.

To have a more appropriate and fair comparison of the several coalitional methods, we decide not to rely only on the α-threshold but rather on the proportion of the *complete* complexity needed as shown in Figure 16. If one does want a short computation time they can set the proportion to 10%, while 50% can be set if the computation time is not a problem and that more accurate results are needed. A well-used bisection method is applied to find the α-threshold that matches the selected proportion of the *complete* complexity.

In the following Section 5.5, we propose a set of tests considering a complexity proportion to 10%, 25% and 50% for each coalitional method. The time taken by the bisection method is, thus, included in the computation time. It is also worth to note that for all the coalitional methods -except for the *Model-based*- the group generation relies only on the α-threshold and not on the model used. Thus one does not need to generate again the groups if we switch to another model, while any *SHAP*-based method needs to be retrained for any new model, even if it only differs from the previous one from a small change in one hyper-parameter.

5.5 Performance between literature and coalitional methods over different types of datasets

So as to have a clearer view of the impact of all methods in one figure, *Performance Map* are displayed for each model. But as the number of attributes and instances in a dataset has a strong impact on the performances, we split the datasets into two parts. The first one includes datasets that have relatively few instances (strictly less than 500) or attributes (strictly less than 6), whereas the second part only includes datasets with a higher number of instances (at least 500) and attributes (at least 6). There are 213 datasets in the first set and 30 in the second one.

Figure 17 shows the results for the Random Forest model for the coalitional methods -*PCA*, *Spearman*, *Reverse Spearman* and *Reverse VIF*- for three complexity proportion -10%, 25% and 50%-, the *k-depth* methods -from *linear* to *complete*- and *TreeSHAP*. The left subgraph shows the results for the first set containing datasets with few attributes or instances and the subgraph on the right indicates results for more complex datasets as described previously.

TreeSHAP has on average a higher computation time than the *complete* method with an error between the *4-depth* and *5-depth* for both sets of datasets. Coalitional methods show strong results, *PCA - 50%* has a computation time equivalent to a *3-depth* but has an error closer to the *5-depth* for both sets. Similarly, all coalitional methods with a threshold of 25% are in terms of computation time closer to the *2-depth* while being more accurate than *3-depth*. This is explained by the fact that coalitional methods generates smarter groups with less useless or redundant information than *k-depth*, thus being more efficient.

Figure 18 shows the *Performance Maps* for the SVM model for both sets of datasets. *SHAP* are not displayed since *TreeSHAP* is not usable with SVM and *KernelSHAP* has a really high computational time as previously shown in Figure 12, hence flattening the figure if displayed.
Unlike Random Forest, there is a clear difference between the results for the two sets. For smaller datasets -either in terms of an attribute or instance number- some coalitional methods are longer to compute than the *complete* one. This is because the time taken to find the appropriate α-threshold with the bisection method is too large relative to the global computation time, which also includes training models and influence computation for each attribute.

Nevertheless, for larger datasets, the performances of coalitional methods are satisfying. Indeed, in a similar way to Random Forest results, coalitional methods with a 50%-threshold are faster to compute than *4-depth* method for the

same error with respect to the *complete* -especially *PCA* which is really efficient. *Spearman* and *PCA* with a 25% threshold are very efficient as well, with a computation time between those of *2-depth* and *3-depth* while being more accurate than *3-depth*.

5.6 How to interpret the coalitional explanations

In this section, we show an example of the use of our coalitional methods on a real use case dataset and for specific instances from this dataset. For this experiment, we only consider the *Spearman* method, since it produces one of the best results as discussed in the previous sections. We also include the *Kernel SHAP* method for a comparison with the explanations found with the *Spearman* approach.

The quality of an interpretability method is a subjective concept and it would be difficult to theorize measures to assess what constitutes good interpretability. Nevertheless, some criteria exist in the literature to evaluate individual explanations [37]. Properties such as fidelity and comprehensibility can help non-experts to evaluate and compare individual explanations, thus explanations methods. Fidelity represents the ability of an explanation to approximate the prediction of the "black box" model and comprehensibility evaluates the ability of users to understand the explanations.

The use case dataset concerns the SARS-COV2 - also called Covid-19 - epidemic outbreak in France during spring 2020. Data collection complied with the European GDPR rules and consists of anonymized medical information of 409 patients with Covid-19 virus hospitalized at the Centre Hospitalier Intercommunal de Créteil [4] between March and May 2020. The primary binary outcome consists of the deterioration of the patient's state of health during their stay, also called aggravation. Deterioration was defined as the requirement for mechanical ventilation, presence of septic shock, acute respiratory distress syndrome, a requirement for resuscitation maneuvers during hospitalization or hospital mortality. Out of the total number of patients, 176 of them had a deterioration in their health state, i.e. 43% of the data set. Each patient profile is established at the patient's arrival at the hospital. Available information consists of 10 attributes such as basic characteristics (age and gender), exam results of Chest Computed-Tomography (CT) scan severity, and comorbidities like cancer, type-2 diabetes, obesity, intellectual disability, cardiovascular disease. For this use case, a *Random Forest* model and the *Spearman* coalitional method with a complexity threshold of 25% are used. The model has an accuracy of 74% with an 80% precision and a 69% recall.

Figure 19 and 20 give the average absolute influence of each attribute, with or without taking into account the class predicted by the model, for the *Spearman 25%* and *Kernel SHAP* method respectively. Age and chest Chest scan severity are the two most important attributes for both methods, with Chest scan severity having a greater impact on aggravation class. This shows a coherence between the medical reality and both explanation methods. Indeed, a high Chest scan severity is strongly associated with an aggravation of the health state as shown in [38]. Both methods also have different results for other attributes, such as cardio-vascular disease, cancer and mental disability that have on average almost no impact with $KernelSHAP$ and all attributes have on average a higher influence with the $Spearman$ method. Taking into account classes, the average influences for both classes are relatively similar using $KernelSHAP$, except for the age and severity of the chest CT scan. With the $Spearman$ method, the average influences of ageusia anosmia, diabetes and insulin treatment are dissimilar. For older patients with high chest scanner severity, type-2 diabetes, insulin treatment, or ageusia anosmia, the model is likely to predict a higher risk of deterioration with $Spearman$ since the average absolute influence of these attributes is higher for the aggravation class.

All these behaviors from our model are coherent with the clinical literature about Covid-19 [39]. In contrast, with the $KernelSHAP$ method, the near-zero average influences for some attributes are inconsistent with known risk factors.

Another important point of the explanations is the fidelity and ease of understanding and interpreting them. Although very subjective, these parameters are essential to take into account in the medical field, since a lack of fidelity to the model and understanding of the explanations can lead to wrong decision making and consequences for the health of patients. To evaluate this, one instance of each class from the Covid-19 dataset was randomly drawn to describe and evaluate the explanation of the $KernelSHAP$ and the $Spearman$ coalitional method. Figures 21 and 22 show the influence of each attribute for these patients, whose descriptions are given below. Patient A is a 54-year-old obese man with no clinic sign of infection in his chest CT scan. He also has insulin treatment and signs of ageusia or anosmia. The two methods find that the value of Chest CT scan severity and age for this patient contributes the most to the prediction of non-aggravation while his gender, his symptoms of anosmia and ageusia, his obesity, and his insulin treatment goes against the prediction. The explanations allow us to understand that this patient has many risk factors and that the non-aggravation prediction comes mainly from the absence of severity of the chest CT scan and the patient's age. However, for the *Kernel SHAP* method, the absence of cardiovascular disease goes against non-aggravation prediction while it contributes to the prediction for the *Spearman* method. This seems contrary to medical knowledge about

[4] `https://www.chicreteil.fr/`

Covid-19 [39], since cardiovascular disease is a risk factor. The absence of disease should therefore be in favour of a non-aggravation of the patient's state of health.

Patient B is a 76-year-old man with type-2 diabetes, insulin treatment, ageusia anosmia. The severity of his chest CT scan is 4 out of 4 which is a critical value. For *Kernel SHAP* method, the chest scan severity is way more important than other attributes in the prediction. For *Spearman* method, even if the severity of the chest CT scan is significant, the presence of insulin treatment, the patient's gender and age are important. The absence of cancer, cardiovascular disease, intellectual disability, and obesity goes against the prediction, while there are no impact with *Kernel SHAP* method. *Spearman*'s explanations are slightly more contrasted than *Kernel SHAP*'s ones.

For this use case, the two methods are easy to understand as there are based on the same additive strategy. For both methods and both examples, influences approximate closely model predictions, and therefore have a high fidelity. However, this fidelity is only local, as methods only explain individual data instances. Moreover, and based on the clinical literature about Covid-19 [39], the explanations from *Spearman* method seem more consistent for comorbidities. Finally, the complete dataset was computed in 51 seconds with the *Spearman* method when it took more than 18 minutes for the *Kernel SHAP* method, for similar results.

6 Source code

The full implementation of our proposals (including all the methods proposed in Section 4 as well as the *K-complete* and *Complete* methods) is available here: `https://github.com/kaduceo/coalitional_explanation_methods`.
The source code will evolve considering future works.

7 Conclusion and perspectives

This paper explored several approximations of the additive explanation method based on Shapley values, named *complete* method in the paper since this method is computationally exponential with respect to attribute number, thus intractable in most practical cases. We compared existing approximation methods such as *SHAP* or *k-depth* that are both limited in their inclusion of attribute interdependence which has a clear impact on their performances either in their accuracy, with respect to the *complete* method, or computation time. In order to take into account the interaction between attributes more efficiently, we developed several methods, called *coalitional* methods, based on smarter grouping attribute procedures that only retain relevant groups of attributes, thus lowering complexity and computation time while maintaining an acceptable precision. Tests were conducted on 243 datasets with a different number of attributes and instances. *Coalitional* methods show the most promising results, especially those based on *PCA*, *Spearman* and *Reverse VIF*. These new methods notably outperform existing ones, whether *k-depth* or *SHAP*-based, for more complex datasets where attribute interdependence is more likely to be present. Although our results open up encouraging perspectives of practical application of individual prediction interpretability, either in terms of accuracy or in computation time, the main problem is that computing the *complete* influences, which is the comparison baseline for our study, becomes near impossible with larger attribute numbers. Thus, it is very difficult to monitor the performance of our different methods with this baseline. A possible way to address this problem could be first to run a global attribute importance study for large datasets using methods such as *Permutation Importance* that is model agnostic or *Gini Importance* for tree-based models. Then use this information to compute influences only for the most important attributes during the individual explanation generation.

A longer-term perspective is also to take into account the context where the predictive analysis is conducted. Indeed, the explanations provided for particular instances cannot be totally satisfactory for any users in any situations. The degree of user expertise in the data seems very important to consider: an expert user will certainly be more interested in very precise explanations than a novice one. Moreover, the analysis process may have an impact on the type of explanations to consider. The explanations should not be analysed in the same way depending on whether the analysis is carried out in an exploratory or confirmatory manner.

8 Acknowledgement

The use-case dataset was acquired in collaboration with the Centre Hospitalier Intercommunal de Créteil. Therefore, we greatly acknowledge the managers and physicians involved in this project.

References

[1] Amina Adadi and Mohammed Berrada. Peeking inside the black-box: A survey on explainable artificial intelligence (xai). *IEEE Access*, 6:52138–52160, 2018.

[2] Diogo V Carvalho, Eduardo M Pereira, and Jaime S Cardoso. Machine learning interpretability: A survey on methods and metrics. *Electronics*, 8(8):832, 2019.

[3] Erik Strumbelj and Igor Kononenko. An Efficient Explanation of Individual Classifications Using Game Theory. *J. Mach. Learn. Res.*, 11:1–18, March 2010. Publisher: JMLR.org.

[4] Giuseppe Casalicchio, Christoph Molnar, and Bernd Bischl. Visualizing the feature importance for black box models. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 655–670. Springer, 2018.

[5] Scott M Lundberg and Su-In Lee. Consistent feature attribution for tree ensembles. *arXiv preprint arXiv:1706.06060*, 2017.

[6] Erik Štrumbelj and Igor Kononenko. Towards a model independent method for explaining classification for individual instances. In *International Conference on Data Warehousing and Knowledge Discovery*, pages 273–282. Springer, 2008.

[7] Guy Van den Broeck, Anton Lykov, Maximilian Schleich, and Dan Suciu. On the tractability of shap explanations, 2020.

[8] Gabriel Ferrettini, Julien Aligon, and Chantal Soulé-Dupuy. Explaining single predictions: A faster method. In Alexander Chatzigeorgiou, Riccardo Dondi, Herodotos Herodotou, Christos Kapoutsis, Yannis Manolopoulos, George A. Papadopoulos, and Florian Sikora, editors, *SOFSEM 2020: Theory and Practice of Computer Science*, pages 313–324, Cham, 2020. Springer International Publishing.

[9] Gabriel Ferrettini, Julien Aligon, and Chantal Soulé-Dupuy. Improving on coalitional prediction explanation. In Jérôme Darmont, Boris Novikov, and Robert Wrembel, editors, *Advances in Databases and Information Systems - 24th European Conference, ADBIS 2020, Lyon, France, August 25-27, 2020, Proceedings*, volume 12245 of *Lecture Notes in Computer Science*, pages 122–135. Springer, 2020.

[10] Scott M Lundberg and Su-In Lee. A Unified Approach to Interpreting Model Predictions. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 4765–4774. Curran Associates, Inc., 2017.

[11] André Altmann, Laura Toloşi, Oliver Sander, and Thomas Lengauer. Permutation importance: a corrected feature importance measure. *Bioinformatics*, 26(10):1340–1347, 2010.

[12] Kenji Kira and Larry A Rendell. A practical approach to feature selection. In *Machine Learning Proceedings 1992*, pages 249–256. Elsevier, 1992.

[13] Andreas Henelius, Kai Puolamäki, and Antti Ukkonen. Interpreting classifiers through attribute interactions in datasets. *arXiv preprint arXiv:1707.07576*, 2017.

[14] Sandra Wachter, Brent Mittelstadt, and Chris Russell. Counterfactual explanations without opening the black box: Automated decisions and the gdpr. *Harv. JL & Tech.*, 31:841, 2017.

[15] James Wexler, Mahima Pushkarna, Tolga Bolukbasi, Martin Wattenberg, Fernanda Viégas, and Jimbo Wilson. The what-if tool: Interactive probing of machine learning models. *IEEE transactions on visualization and computer graphics*, 26(1):56–65, 2019.

[16] E. Strumbelj and I. Kononenko. Explaining prediction models and individual predictions with feature contributions. *Knowledge and Information Systems*, 41:647–665, 2013.

[17] A. Datta, S. Sen, and Y. Zick. Algorithmic transparency via quantitative input influence: Theory and experiments with learning systems. In *2016 IEEE Symposium on Security and Privacy (SP)*, pages 598–617, May 2016.

[18] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. " 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*, pages 1135–1144, 2016.

[19] Radwa ElShawi, Youssef Sherif, Mouaz Al-Mallah, and Sherif Sakr. Interpretability in healthcare: A comparative study of local machine learning interpretability techniques. *Computational Intelligence*, 2020.

[20] Verónica Bolón-Canedo, Noelia Sánchez-Maroño, and Amparo Alonso-Betanzos. A review of feature selection methods on synthetic data. *Knowledge and Information Systems*, 34(3):483–519, 2013.

[21] Lei Yu and Huan Liu. Efficient feature selection via analysis of relevance and redundancy. *J. Mach. Learn. Res.*, 5:1205–1224, December 2004.

[22] Alain Rakotomamonjy. Variable selection using svm based criteria. *J. Mach. Learn. Res.*, 3(null):1357–1370, March 2003.

[23] M Mejía-Lavalle, E Sucar, and G Arroyo. Variable selection using svm based criteria. In *International workshop on feature selection for data mining*, page 131–1350, 2006.

[24] Mark A. Hall. *Correlation-based Feature Selection for Machine Learning*. PhD thesis, 1999.

[25] L. S. Shapley. A value for n-person games. *Contributions to the Theory of Games*, (28):307–317, 1953.

[26] Simon Eitzinger, Amina Asif, Kyle E Watters, Anthony T Iavarone, Gavin J Knott, Jennifer A Doudna, and Fayyaz ul Amir Afsar Minhas. Machine learning predicts new anti-CRISPR proteins. *Nucleic Acids Research*, 48(9):4698–4708, 04 2020.

[27] E. Tjoa and C. Guan. A survey on explainable artificial intelligence (xai): Toward medical xai. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1–21, 2020.

[28] Jean-Emmanuel Bibault, Daniel Chang, and Lei Xing. Development and validation of a model to predict survival in colorectal cancer using a gradient-boosted machine. *Gut*, 09 2020.

[29] Simon Meyer Lauritsen, Mads Kristensen, Mathias Vassard Olsen, Morten Skaarup Larsen, Katrine Meyer Lauritsen, Marianne Johansson Jørgensen, Jeppe Lange, and Bo Thiesson. Explainable artificial intelligence model to predict acute critical illness from electronic health records. *Nature communications*, 11(1):1–11, 2020.

[30] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "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*, KDD '16, pages 1135–1144, New York, NY, USA, 2016. ACM.

[31] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. Learning Important Features Through Propagating Activation Differences. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, pages 3145–3153, 2017. event-place: Sydney, NSW, Australia.

[32] Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PLOS ONE*, 10(7):1–46, 07 2015.

[33] Stan Lipovetsky and Michael Conklin. Analysis of regression in game theory approach. *Applied Stochastic Models in Business and Industry*, 17:319 – 330, 10 2001.

[34] Andreas Henelius, Kai Puolamaki, Henrik Boström, Lars Asker, and Panagiotis Papapetrou. A peek into the black box : exploring classifiers by randomization. *Data mining and knowledge discovery*, 28(5-6):1503–1529, 2014. QC 20180119.

[35] Sara Makki. *An Efficient Classification Model for Analyzing Skewed Data to Detect Frauds in the Financial Sector*. PhD thesis, Université de Lyon; Université libanaise, 2019.

[36] Joaquin Vanschoren, Jan N. van Rijn, Bernd Bischl, and Luis Torgo. Openml: Networked science in machine learning. *SIGKDD Explorations*, 15(2):49–60, 2013.

[37] Marko Robnik-Sikonja and Marko Bohanec. Perturbation-Based Explanations of Prediction Models. In *Human and Machine Learning*, pages 159–175. June 2018.

[38] Marco Francone, Franco Iafrate, Giorgio Maria Masci, Simona Coco, Francesco Cilia, Lucia Manganaro, Valeria Panebianco, Chiara Andreoli, Maria Chiara Colaiacomo, Maria Antonella Zingaropoli, et al. Chest ct score in covid-19 patients: correlation with disease severity and short-term prognosis. *European radiology*, 30(12):6808–6817, 2020.

[39] Zhaohai Zheng, Fang Peng, Buyun Xu, Jingjing Zhao, Huahua Liu, Jiahao Peng, Qingsong Li, Chongfu Jiang, Yan Zhou, Shuqing Liu, et al. Risk factors of critical & mortal covid-19 cases: A systematic literature review and meta-analysis. *Journal of Infection*, 2020.

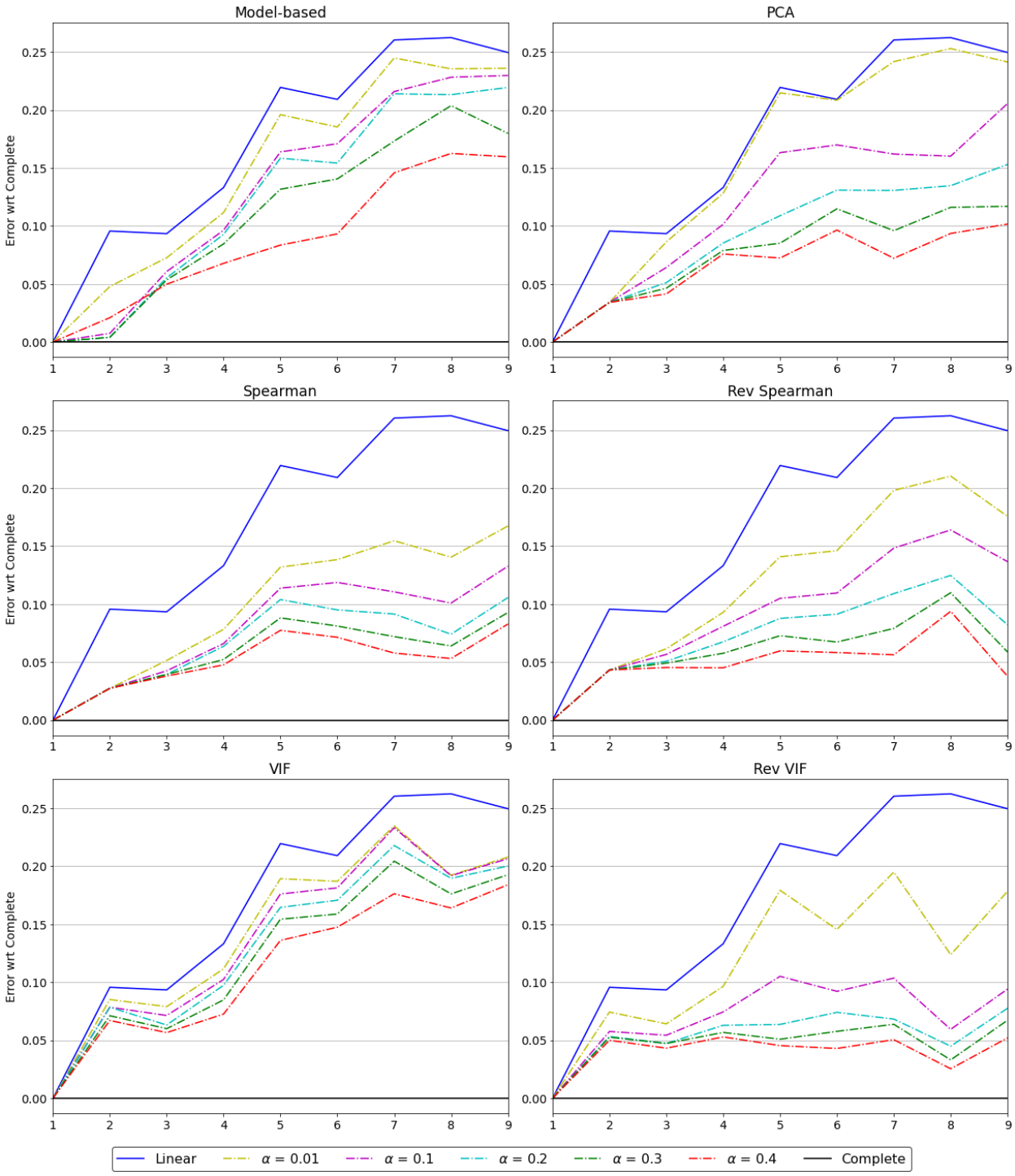

Figure 9: Error score between each coalitional method and the complete influence, versus the number of attributes in the dataset averaging over the two models

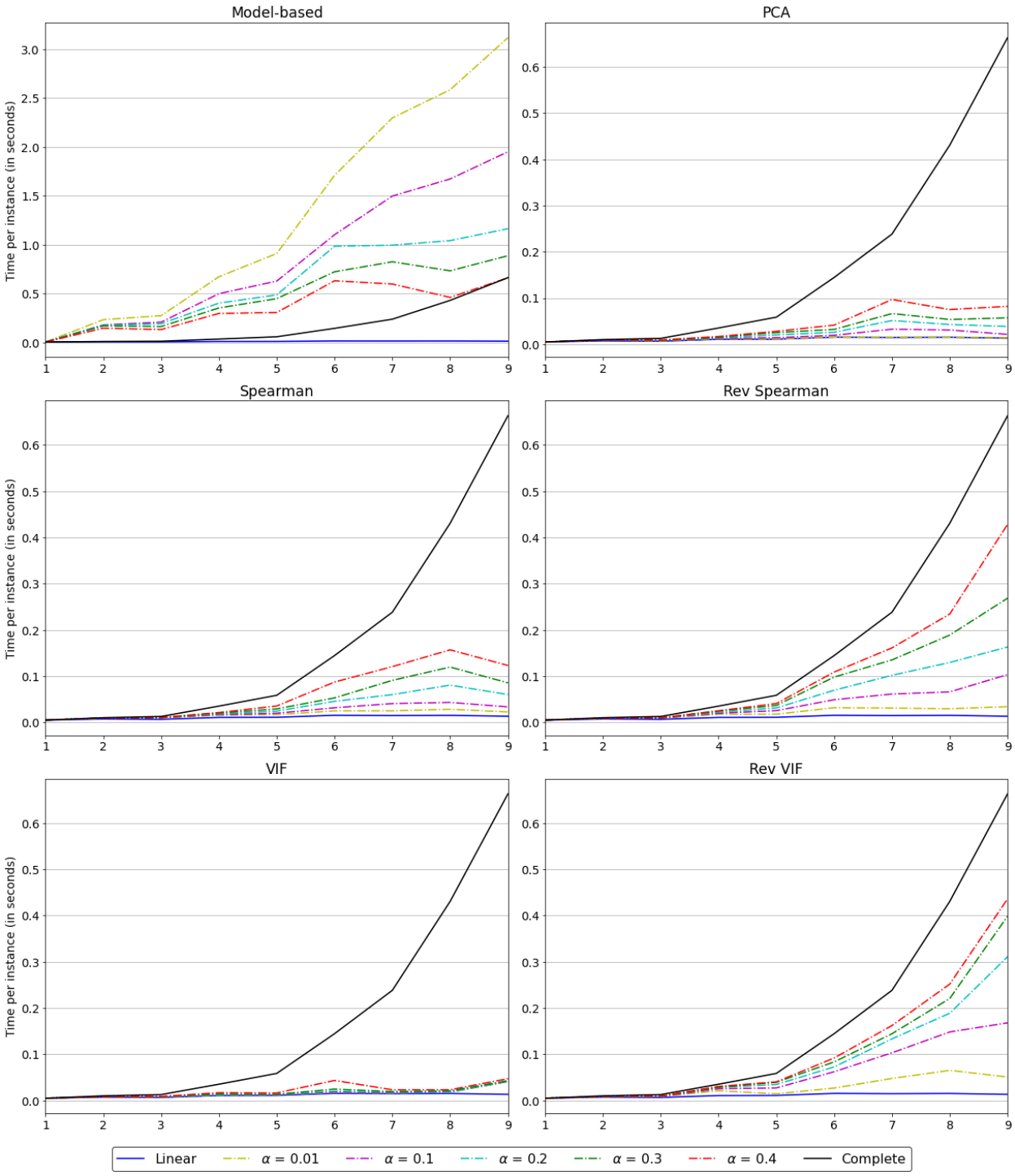

Figure 10: Time per instance of each coalitional method versus the number of attributes in the dataset, averaging over the two models

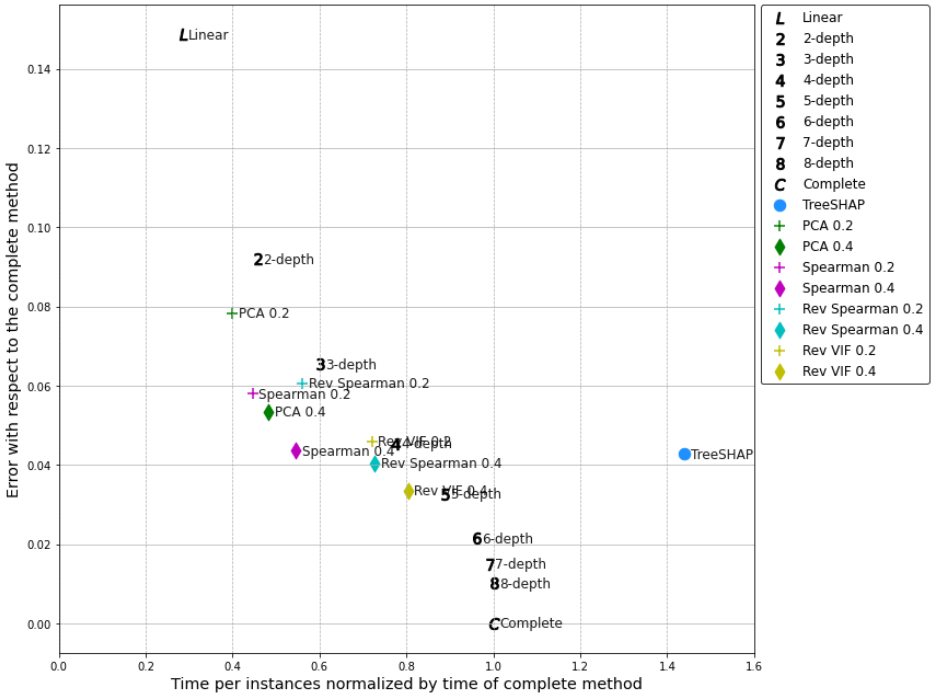

Figure 11: Performance map over all datasets for k-complete, SHAP and coalitional methods for Random Forest

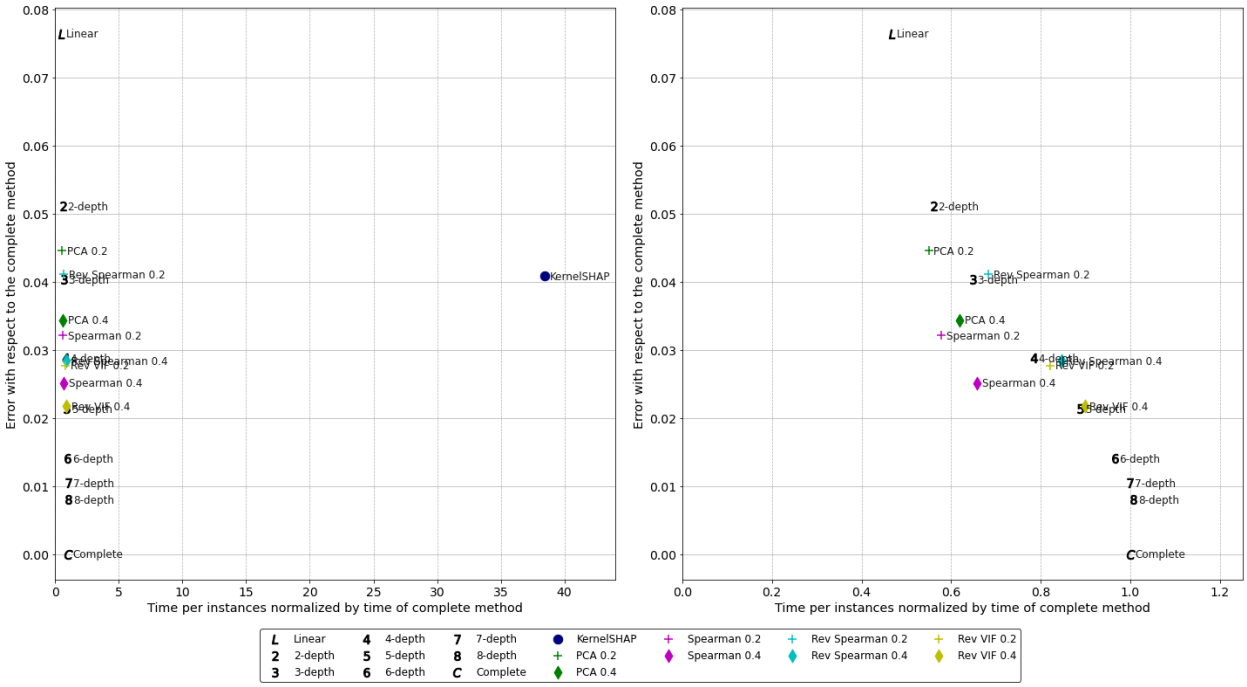

Figure 12: Performance map over all datasets for k-complete, SHAP and coalitional methods for SVM

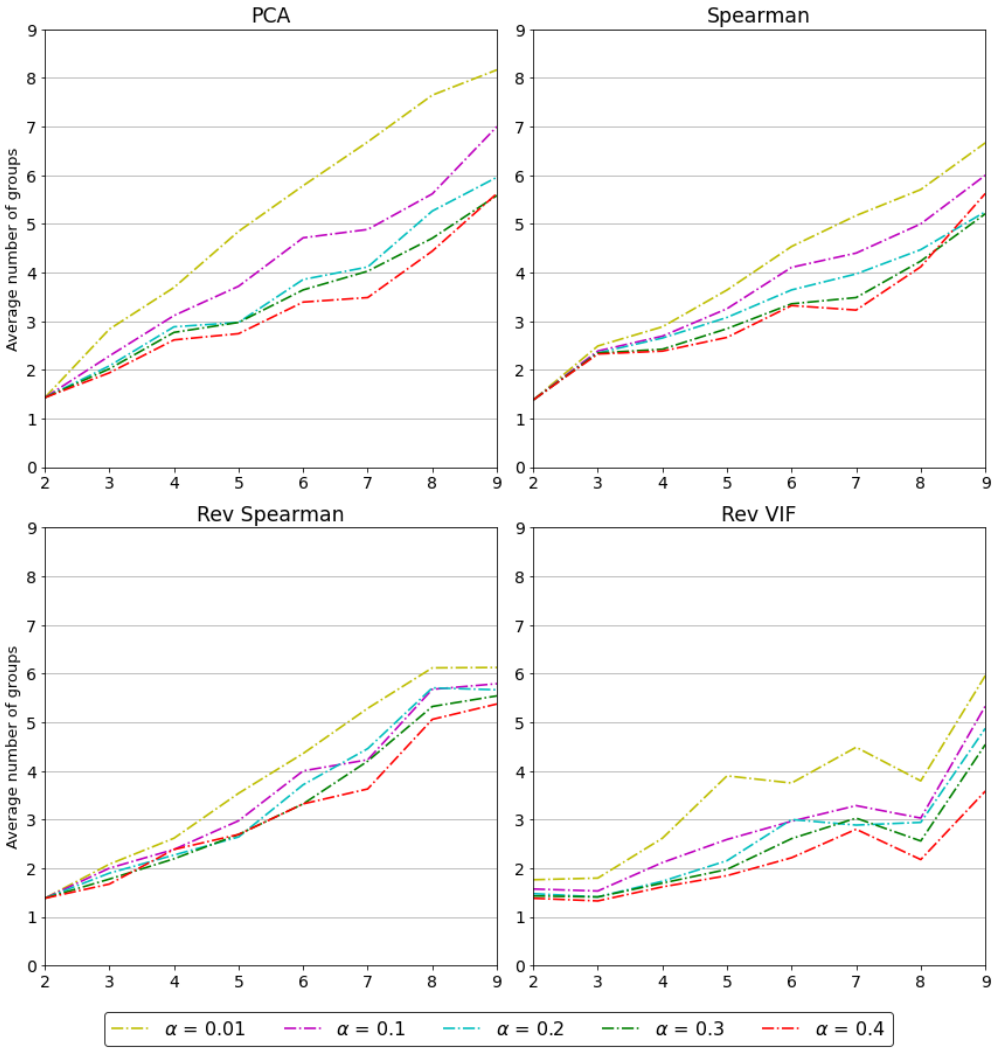

Figure 13: Mean number of groups for coalitional methods depending on α-threshold and number of attributes

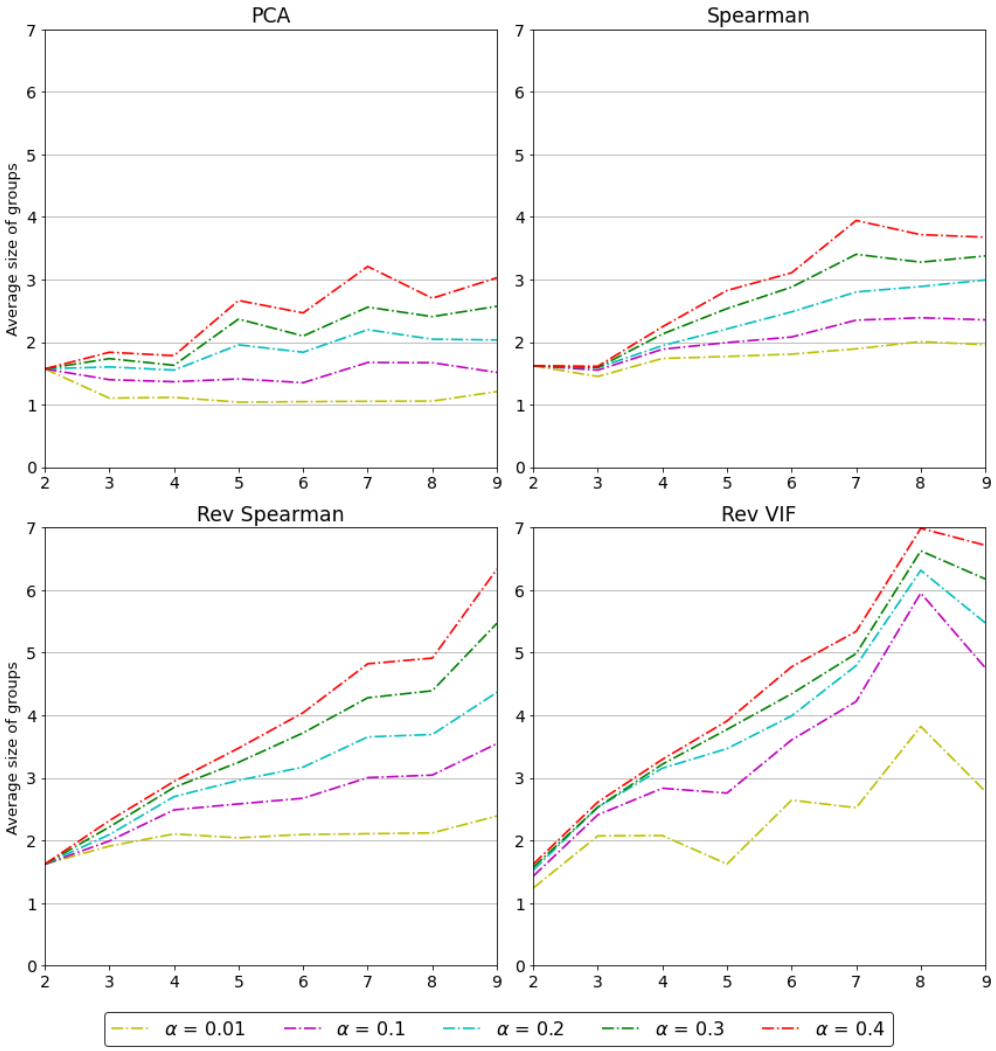

Figure 14: Mean size of groups for coalitional methods depending on α-threshold and number of attributes

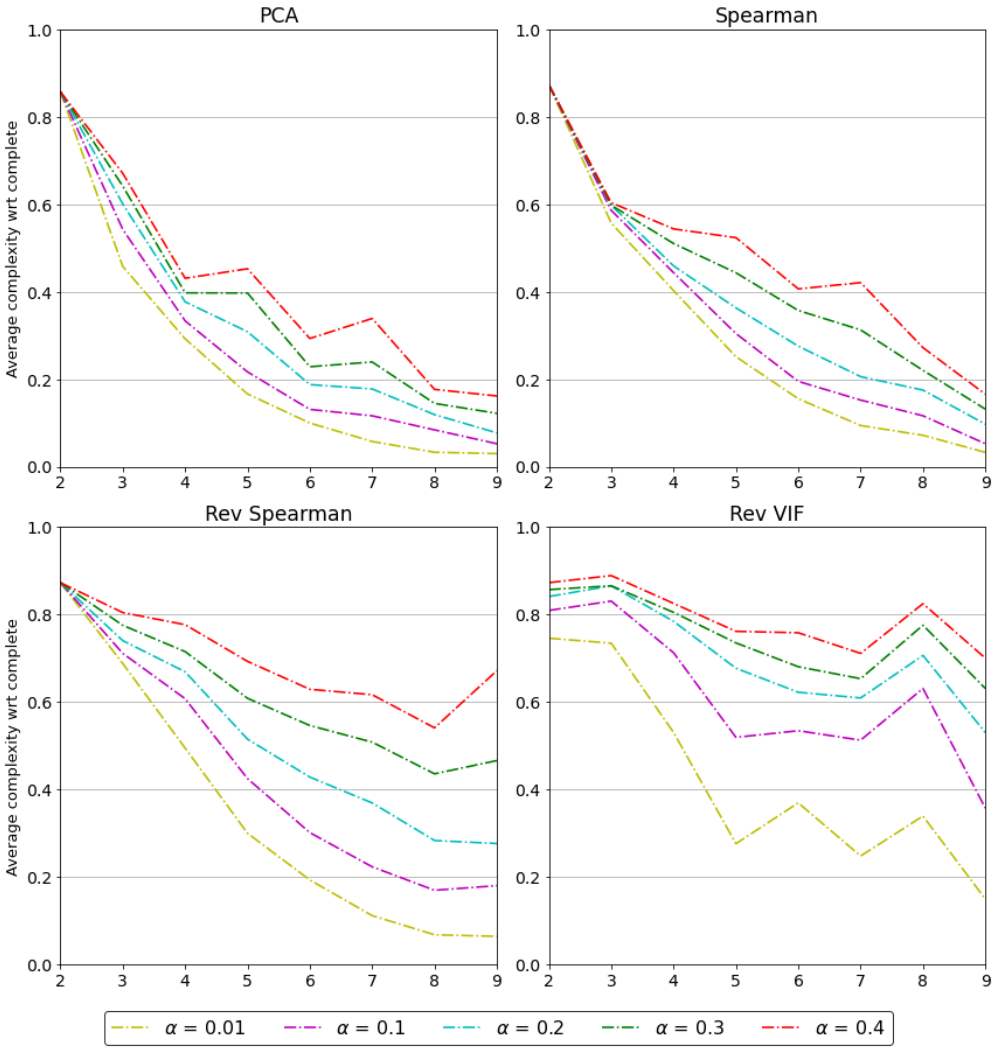

Figure 15: Mean complexity proportion with respect to complete complexity for coalitional methods depending on α-threshold and number of attributes

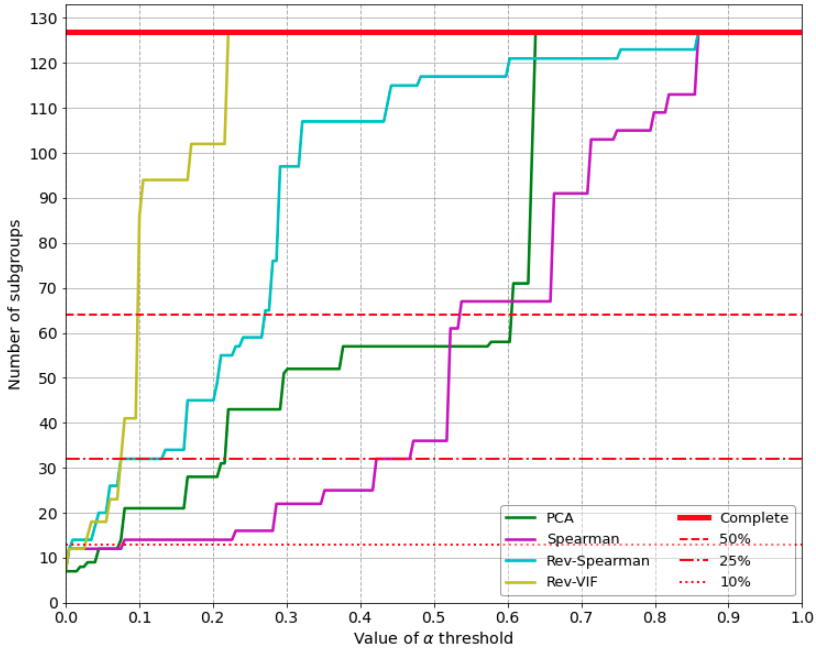

Figure 16: Evolution of complexity of coalitional methods depending on α-threshold

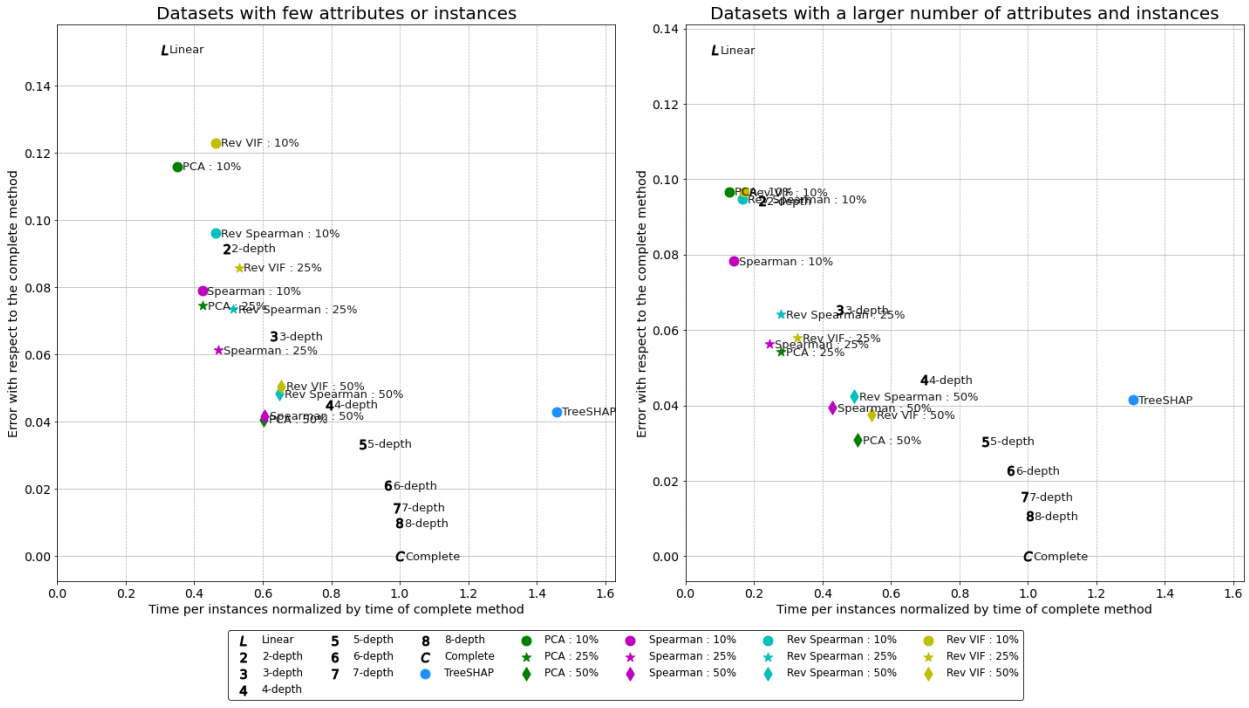

Figure 17: Performance maps for two sets of datasets for coalitional methods for Random Forest

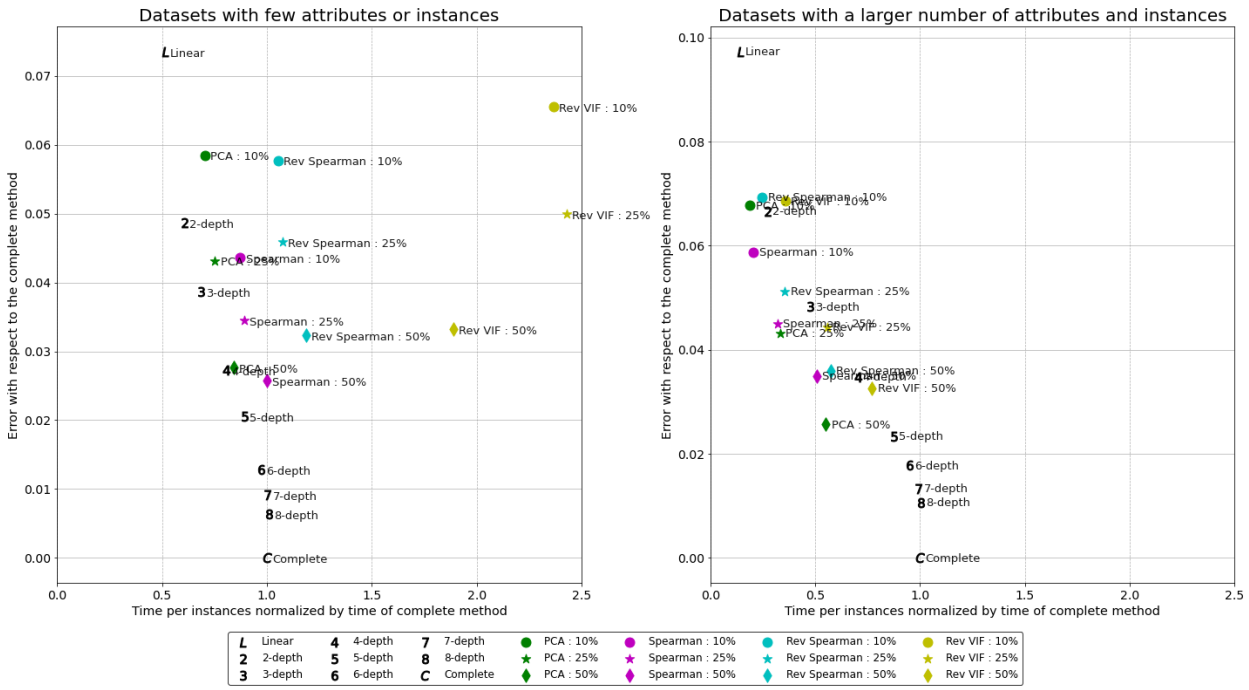

Figure 18: Performance maps for two sets of datasets for coalitional methods for SVM

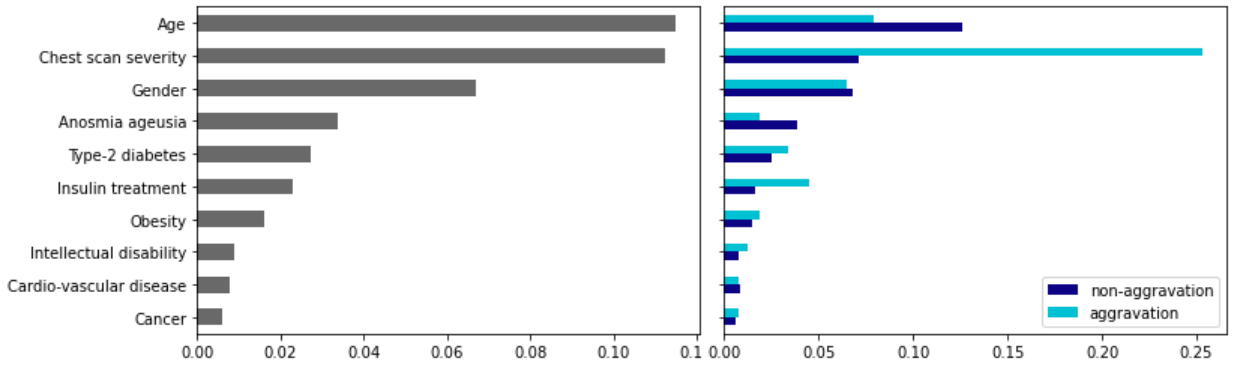

Figure 19: Mean absolute influence for each attribute with Spearman 25% method

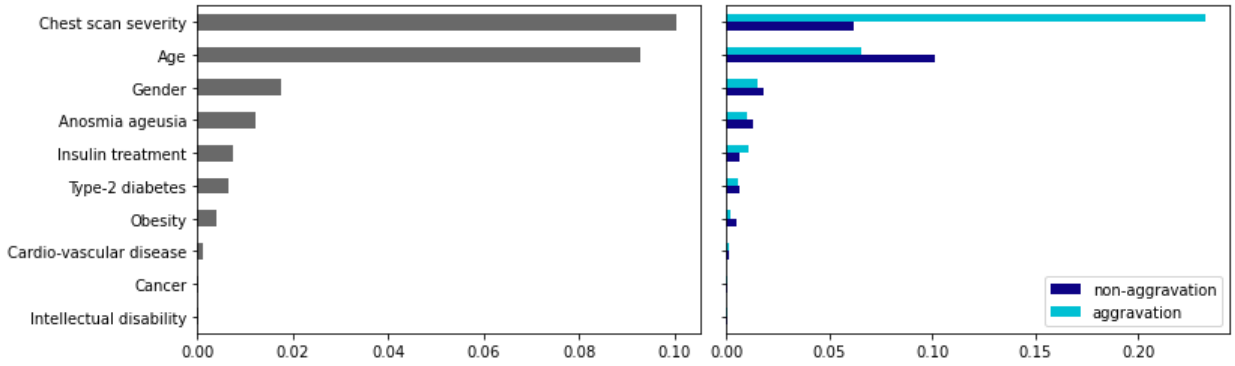

Figure 20: Mean absolute influence for each attribute with Kernel SHAP method

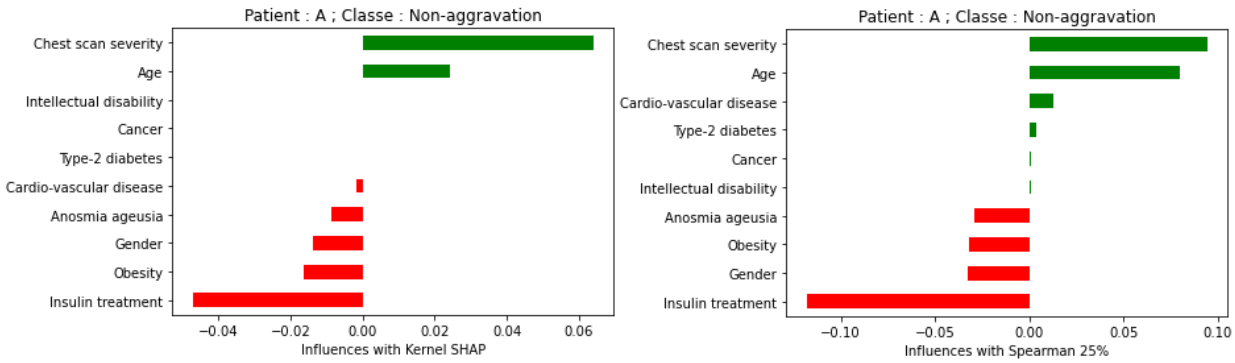

Figure 21: Influences of patient A with Kernal SHAP and Spearman 25%

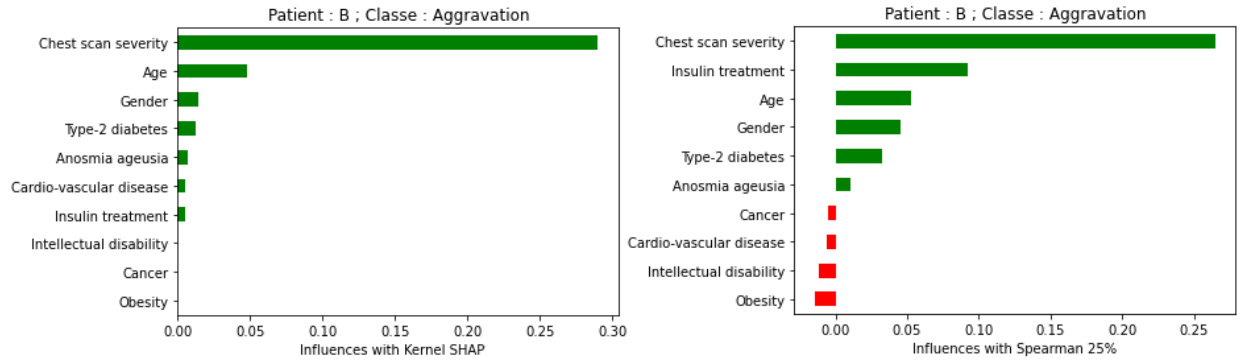

Figure 22: Influences of patient B with Kernal SHAP and Spearman 25%