Visualization of feature importance dependencies in machine learning

G.S. Koehorst
University of Twente
P.O. Box 217, 7500AE Enschede
The Netherlands
g.s.koehorst@student.utwente.nl

ABSTRACT

There is a growing need for methods that can help to explain machine learning models. Feature importance analysis is often used to determine the relationship between a feature and the target of the model. However, this analysis does not take into account the dependencies between features. This research defines three model-agnostic methods to visualize feature importance dependencies. The defined methods are evaluated using synthetic data. The proposed methods are a feature importance dependency bar chart, holding out on correlation pie-charts, and a feature importance dependency tree. All three methods were able to effectively visualize feature importance dependencies when the number of features was six or lower. When the number of features was eight, the tree method became less effective. However, we conclude that all three methods were able to visualize feature importance *dependencies*. So, the contribution of this paper is that we define and evaluate three approaches to visualize feature importance *dependencies*, which can help us in showing how dependencies affect feature importances, in order to improve the explainability of machine learning.

Keywords

Feature importance analysis, feature dependencies, visualization, machine learning

1. INTRODUCTION

Today, machine learning is being used everywhere to make predictions and aid in decision-making in various industries. Recommendations of a machine learning algorithm can be critical in making decisions and so, inaccurate predictions can have detrimental effects. Hence, complex machine learning models are being adopted to increase accuracy. The increase in accuracy often comes at the cost of interpretability, because the complexity of the model causes it to be more like a black box. This makes it hard to interpret the results and to reason why the model gives a certain outcome. Hence, the call for explainable machine learning has grown.

To have a better understanding of the relation between a feature and the target of a machine learning model, feature importance analysis is often performed. This analysis aims to lead to a score for each feature as to how important they are for the predictions of the model. The importance of a feature can, as originally described by Breiman[1], be determined by randomly permuting the value of the feature to see what the effect is on the accuracy of the model. Although the method is based on random forest [1], and has different variations that are specific for random forests[8], the simple concept allows the method to be applied to basically any machine learning model. It has for example been used to estimate feature importance for deep learning models [2], logistic regression and support-vector machines [3]. Another often-used approach is the hold-out method: instead of permuting it, the value is completely withheld. However, these methods do not take into account the dependencies between features. If two features are highly correlated, permuting or holding out one of the two features will not necessarily decrease the performance of the model, because the other feature has a similar predictive value. However, this does not mean that the feature is not important, it could simply be as important as the related feature. The contribution of this paper is that we define and evaluate three methods for visualizing feature importance *dependencies*, which can help us in showing how dependencies affect feature importances, in order to improve the explainability of machine learning.

28^{th} *Twente Student Conference on IT* Febr. 2^{nd}, 2018, Enschede, The Netherlands.

2. PROBLEM STATEMENT

Feature importance analysis is often used to determine the feature importances of machine learning models. However, this analysis does not directly take into account dependencies between the features. So, an associated effective visualization is desirable, such that feature importance analysis becomes less misleading in order to improve decision making.

So this problem leads to the following **research question**:

How can we visualize feature dependencies when performing feature importance analysis for the purpose of improving the explainability of machine learning models?

The following sub-questions will be used to answer the main research question:

SQ1: What are possible methods for visualizing feature importance dependencies?

SQ2: Which of the visualizations defined for **SQ1** are the most effective in visualizing feature importance dependencies?

3. RELATED WORK

Here we will go through some related work in feature importance analysis, interpretable machine learning, and visualization of feature dependencies.

In 2008 Xu et al. [10] developed an approach where they decoupled correlated and uncorrelated contributions of a feature to the output of a linear model. Later Xu [9] proposed a more general method that can be applied to various different models where the dependence between the features also does not have to be linear. Feature importance measures based on this decoupling are also recommended by Wei et al. [8] when dealing with correlated input features.

Ribeiro et al. [7] defined the Local Interpretable Model-Agnostic Explanations (LIME) algorithm. LIME aims to explain the predictions of any machine learning model. So, for a given prediction, it shows how much a certain feature contributed to that prediction. It tries to achieve this by approximating the neighborhood of the feature values of a certain prediction with an interpretable model, for example, a linear model. They also defined a procedure that picks individual predictions that will cover most of the different explanations that LIME produces. This method looks at the interpretability of individual predictions, rather than the importance of certain features for the model predictions as a whole.

Furthermore, there are tools that have been developed to observe relationships between features. Nason et al. [5] proposed CARTscans as a tool to visualize complex models. It builds on the concepts of CT scans and aims to display the relationships between features. It can visualize up to four dimensions. So in order to determine which features to visualize, they actually perform permutation importance[1] and pick the most important features according to that measure.

There are algorithms that try to select features to reduce the number of needed features while keeping the accuracy of a model high. Some examples are [4][6]. However, these algorithms are more aimed towards removing redundant variables while keeping accuracy high and do not necessarily care about the relations between features and what effect those dependencies have.

4. VISUALIZATION METHODS

In this section, we propose three methods for visualizing feature importance dependencies.

4.1 Importance dependency bar-chart

This method is a bar-chart that, for every feature, aims to visualize how the importance of that feature changes when the other features are excluded. For every feature, the feature importance is determined if one of the other features is excluded from training. This is repeated for every feature. An example of a resulting visualization with five features could look like the chart in figure 1. An obvious drawback of this method is the fact that it only takes into account dependencies between two features at a time. However, the number of possible combinations of features to be excluded keeps growing when the amount of features grows, so excluding combinations of features becomes too complex.

4.2 Holding out on correlation pie-charts

For this visualization method, we first calculate the initial feature importances. These importance scores are displayed in the initial pie-chart that becomes the central top pie-chart. The correlation coefficients are then calculated between all features. For the two features with the highest correlation coefficient, we repeat the calculation of the feature importances when these two features are excluded both individually and as a pair. For the individual exclusion, the calculated scores are displayed as a pie-chart underneath and to the left and right of the central top pie-chart. For the exclusion as a pair, the scores are displayed centrally and below the top pie chart. For this new central pie chart, the two features with the highest correlation coefficients are determined again and the previous steps are repeated until there are no features left. An example of what this can look like is depicted in figure 2. The top pie chart displays the importance scores of all features when none is excluded. The correlation coefficient of all features is determined and A and B have the highest coefficient. Then three pie charts are created underneath the top one. For the left one, A is excluded and the importance scores are recalculated. For the right one, B is excluded and the importance scores are recalculated. For the middle one, both A and B are excluded and the importance scores are recalculated. For the features that are left for the middle pie-chart, C and D have the highest correlation coefficient and are thus excluded in the pie-charts that are created underneath.

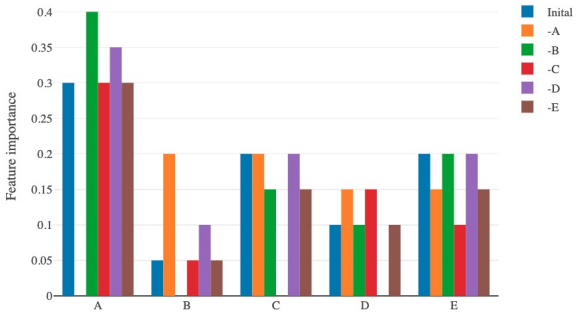

Figure 1: Visualization of the concept of the importance dependency bar-chart. Initial is the feature importance measured when no feature is excluded. For all other entries the feature mentioned in the legend is the one that is excluded.

4.3 Feature importance dependency tree

One of the visualization methods that we propose, is that of a feature importance dependency tree. The tree is built up as follows:

- The root node of the tree is the most important feature according to regular feature permutation importance.
- For all other nodes that branch off to the left from its parent, it holds that the node will be the most important feature excluding all of the nodes that are above it in the tree.
- For all other nodes that branch off to the right from its parent, it holds that the node will be the most important feature excluding all of the nodes that are above it in the tree while those nodes are in fact in the training set.

One example of what this can look like for a tree with 4 features, is shown in figure 3. In this example, we assume that feature A is the most important feature and A and B are highly dependent. This means that if feature A is present in training, feature B does not have a high predictive value and so the first green node coming from the root node is another feature than B. However, if feature A

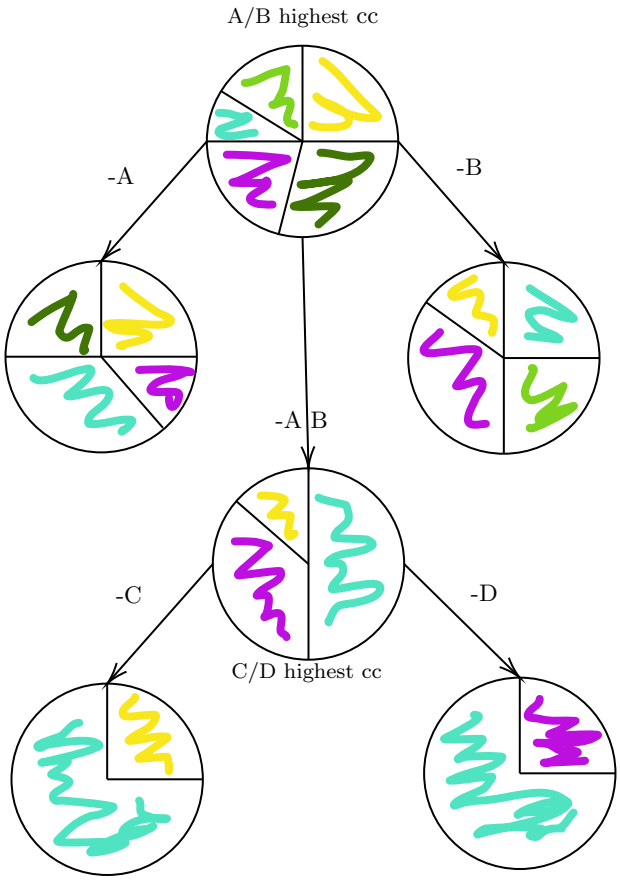

Figure 2: Pie charts visualization of feature importances for different subsets of features. The top pie-chart displays feature importance scores if no feature is excluded. For each central pie-chart, the two features with the highest correlation coefficient are excluded both individually and as a a pair for the pie-charts that come underneath it. The exclusion is individual for the right and left one and paired for the central one.

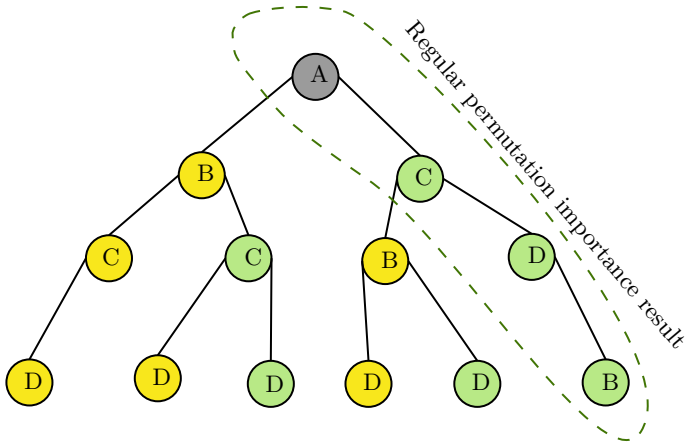

Figure 3: Visual example of feature importance tree concept. The grey root node is the most important feature according to initial feature importance analysis. A yellow node is the most important feature if all nodes above it are excluded completely. A green node is the most important feature if all nodes above it are excluded although those are present during training.

is not present at all, node B has a higher predictive value and hence it will be the most important feature when A is excluded completely. And so it is the first yellow node branched off from the root node. A thing to note is that the area that only contains the root node and green nodes, as shown with a dashed area in figure 3, is in fact the importance order that would be the result of regular feature importance analysis. This is because for all of these nodes they are the most important feature if the other nodes are excluded while those are present in training.

5. EVALUATION OF VISUALIZATIONS

The goal of the visualizations is to add the dimension of dependencies to feature importance analysis in order to display the effect that these dependencies have on the feature importances. With this evaluation, we aim to determine how efficient and effective the proposed visualization methods are in achieving this goal. The visualization methods are evaluated based on four questions. We will go over these questions below.

Q1: To what extent can we determine the order of feature importance if no feature is excluded?

With this question, we aim to determine whether the visualizations show the order of feature importances that conventional feature importance analysis would return. This allows us to compare that order to orders that are shown when certain features are excluded, which is essential if we want to determine the effect of dependencies on the importance orders.

Q2: To what extent can we determine the feature importance scores of all features if no feature is excluded?

This question provides us with information that is supplementary to the importance order of Q1. It shows how large differences in importance there are between the features. This is important since it allows us to see how much the importance scores change when features are excluded.

Q3: How many alternative orders of feature importance can be observed (compared to the order if no feature is

excluded)?

The purpose of this question is to determine whether the three methods differ in the number of importance orders that they display. This could be an indication of how much information the methods are able to display.

Q4: How well can we determine which features are highly dependent?

With this question, we aim to compare how well we are able to extract information about dependencies from the different visualization methods.

These questions are evaluated on synthetic data. This synthetic data will consist of artificially generated importance scores and correlation coefficients. There are several pros to generating feature importances and correlations. First of all, we can evaluate how our methods perform in different situations because we can control input variables that determine what is generated. For example, we can generate the values such that there are no correlations so that we can see how this is visualized with our methods. Secondly, we do not have to (re)train machine learning models. This makes it easy to generate many visualizations in a short amount of time because the required computing power is lower.

An obvious con of these artificial methods is that dependencies between features in real data sets can be more subtle and complex. And so the visualization methods may be less effective for those data sets compared to our generated values.

5.1 Methods for evaluation

5.1.1 Generating visualizations

The approach for generating the visualization is as follows:

- N is the set of features
- I_A is the importance score of feature A, I is a random number between 0.00 - 0.50 for each feature.
- P is the number of distinct feature pairs, $P = N(N-1)/2$
- CC_{ab} is the correlation coefficient between feature A and B
- X is the number of feature pairs with a high CC randomly generated between 0.80 - 1.00
- $P - X$ is the number of feature pairs with a low CC randomly generated between 0.00 - 0.20
- If at any point a feature D is dropped for every feature F out of $N - D$, $I_F = I_F + CC_{FD} * I_D$
- The size of N is varied between 4 and 8
- X is varied between 0 and P, with steps of 1
- For each combination of the size of N and X, 10 visualizations are generated.

.

This approach for generating visualizations is obviously a simplified concept of how feature dependencies impact feature importances, but this gives us the opportunity to evaluate our methods in a controlled setting. The combinations of size N and P that we use as described above can also be seen in table 1.

Table 1: Combinations of number of features and number of highly correlated pairs of features that are used for the generation of visualizations.

no. features	no. highly correlated feature pairs
4	0 - 6
5	0 - 10
6	0 - 15
7	0 - 21
8	0 - 28

5.1.2 Method for evaluating Q1/Q2

To evaluate Q1 and Q2, we randomly pick visualizations from the pool of visualizations from each number of features. Then we check if the defined questions Q1 and Q2 can be answered. This is done qualitatively for 5 visualizations of each feature size, since determining the original feature importance order and the score is done in the same way for each repetition and this process is not influenced by the feature correlations.

5.1.3 Method for evaluating Q3

For **Q3** we define a measure for each of the three methods that are used to evaluate how many different orders of importance are visualized.

For the bar charts method, the measure is defined as follows. Let O_i be the initial importance order when no feature has been excluded yet. According to the defined method each feature, f out of N features is removed once from the original pool of features and the importances are recalculated for the remaining features. This leads to an order O_{-f}. Then if $O_i - f \neq O_{-f}$ there has been a shift in order when feature f got excluded. We then count how many different orders there are. Note that this means that a bar chart visualization can at most have $N + 1$ different orders. One for the initial order and orders for each of the exclusion of N features.

The following order measure has been defined for the pie-charts method. Let us again have an initial order O_i, which is represented by the central top pie-chart. For all the other charts the order is compared with the closest central pie-chart above it. So, let us say that we have a certain chart where feature f is excluded which has an order O_{-f} and the closest central chart above it has got an order O_c, then we say that there is a different order if the order $O_{-f} \neq Oc - f$.

Finally, the measure for the tree method is defined as follows. For each end node, there is a path from the root node to that end node. For each path the order of features is determined. The amount of different orders that are found, is taken as the measure.

5.1.4 Method for evaluating Q4

For each feature size between 4 and 8, we at random choose X between 0 and P (see section 5.1.1 for the definition of X and P). For each visualization method, a visualization that was generated in section 5.1.1 and that corresponds to that feature size and randomly chosen X, is opened. A timer is started and we note down which features we think are highly correlated. The time is stopped when we think we have noted all of them. After each run, it is calculated how many pairs of features we have wrongly identified as highly correlated and also how many we have missed. However, these numbers are not shown during the experiment. This is repeated 5 times for each size of N and each visualization method. We use the same X for each visualization method because this number affects how

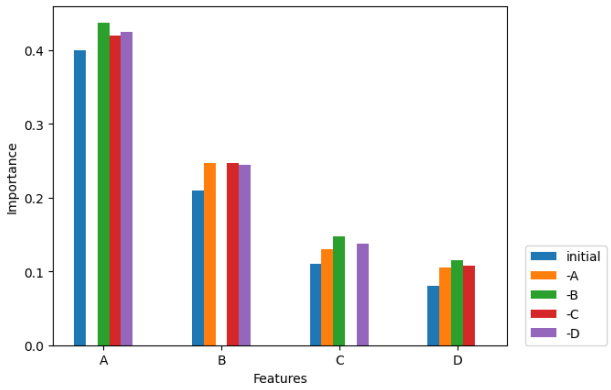

Figure 4: One of the visualizations that was generated for the bar charts method with 4 features and 0 pairs of features with a high correlation coefficient. Initial are the importances when no feature is excluded.

easy it is to determine which features are highly correlated. However, we shuffle the order in which they are evaluated for each method to minimize learning from previous runs.

5.2 Results

Here, we go over the results of the artificial evaluation of the visualization methods.

5.2.1 Results Q1 and Q2

For the bar charts method, it is straightforward to determine the feature importance if no feature is excluded. An example of one of the visualizations that were generated is shown in figure 4. To get the importance order when no features are excluded we simply take the importance order of the bars from the initial group (blue bars in figure 4). When the amount of features increases it becomes somewhat harder to determine the order, but it still remains straightforward. With regards to Q2, it is still possible to get the feature importance values if no feature is excluded. We are not able to see the exact importance scores, but it is possible to make a precise estimation using the y-axis.

One example of a generated visualization for the pie-charts method is shown in figure 5. The figure shows that determining the initial importance order can be done using the top pie chart. However, we can already see that if the amount of features increases, the number of wedges in the pie chart will become so high that it becomes harder to distinguish between them. The importance scores can also be read from the charts, but this also becomes harder when the amount of features increases.

Finally, a generated visualization for the tree method is depicted in figure 6. The most right nodes on, each level of the tree, together form the initial feature importance order and so this is still visible in this method. The feature importances are also depicted for each node. As the tree grows, it becomes more complex to fit the importance scores in the visualization.

Overall, we see that the initial importance orders and scores can be extracted from all three visualization methods. However, when the number of features increases, this becomes more complex. At some point, the number of features will be too large for them to be effectively displayed.

5.2.2 Results Q3

Figure 7 shows the results coming from the defined measures for the number of different orders for each method with 4 and 6 features. In general, we see that first, the

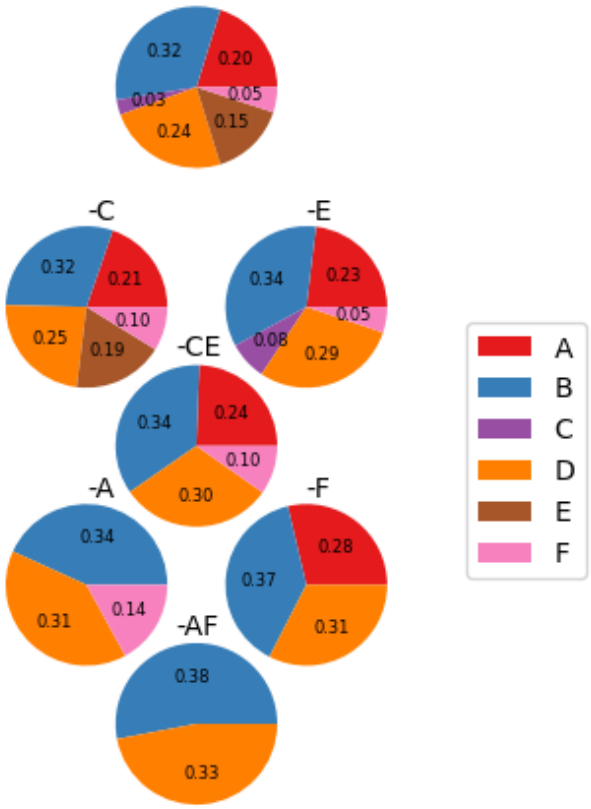

Figure 5: Visualization generated for the pie-charts method with 6 features and 0 highly dependent pairs of features. The top pie-chart shows importances when no feature is excluded. For the others the features depicted above the visualizations are excluded.

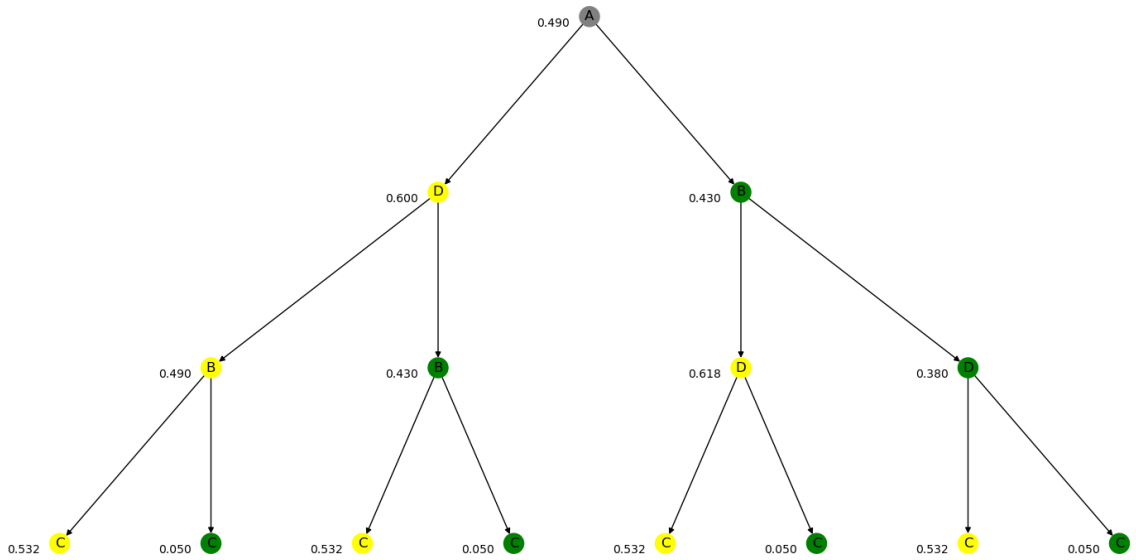

Figure 6: Example of a visualization generated for the tree method with 4 features and 0 highly dependent pairs of features.

number of different orders increases when the number of highly correlated features increases. This is something we want from our visualizations because if there are no features that highly depend on each other, excluding features does not often change the order of importance. However, when the fraction of highly correlated features becomes larger than 0.5, the number of orders decreases again. This follows from the fact that less high correlated features lead to fewer shifts in importances and hence also fewer shifts in order. When the fraction approaches 1, almost all of the features will have shifts in importance which means that their order becomes less likely to shift as well. We also see that the number of orders increases with the number of features, which logically follows from the fact that the amount of possible orders increases as the number of features grows. Finally, there are no clear differences between the methods with regards to the number of different orders.

Figure 8 shows the results coming from the defined measures for the number of different orders for each method with 8 features. Here we see that, compared to figure 7, there is now a difference between the tree method and the other two methods. The tree method shows a higher number of orders. This has to do with the way we defined the measure, but also with the structure of the tree itself. As the number of features grows, the number of end nodes grows exponentially. Since we have defined the number of orders as the number of different paths from the root node to an end node, the number of possible paths increases when there are many highly correlated features.

In general, we see that the three visualizations are similar with regard to the number of order measures that we defined when the number of features is six or lower. At a higher number of features, the tree method has more different orders. So the tree method may be able to show more information when the number of features is higher. However, this extra information may not be that useful and may even make it harder to interpret the visualization.

5.3 Results Q4

Figure 9 shows the meantime that was taken to determine which features were highly correlated for the different visualization methods. We see that the time taken in general increases as the number of features increases. This is no surprise since more information is displayed in the visualizations if there are more features. We also see that for a lower amount of features it took more time with the tree method, but a higher number of features this time was similar compared to the other methods.

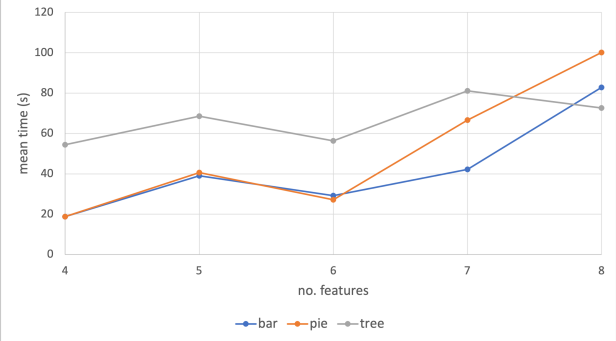

Figure 9: Time taken to determine which features were highly correlated in visualizations with different amounts of features. n = 5 with the number of highly correlated features randomly chosen.

Finally, figure 10 shows the number of mistakes that were made when identifying highly correlated features with the different visualization methods. We see that, especially when the number of features increases, most mistakes are made with the tree method, followed by the pie-chart method. So, even though we saw in figure 9 that the time it took did not increase with the number of features for the tree method, the amount of mistakes increases a lot. So the tree method was not able to display all of the interactions between the features or it was too complex

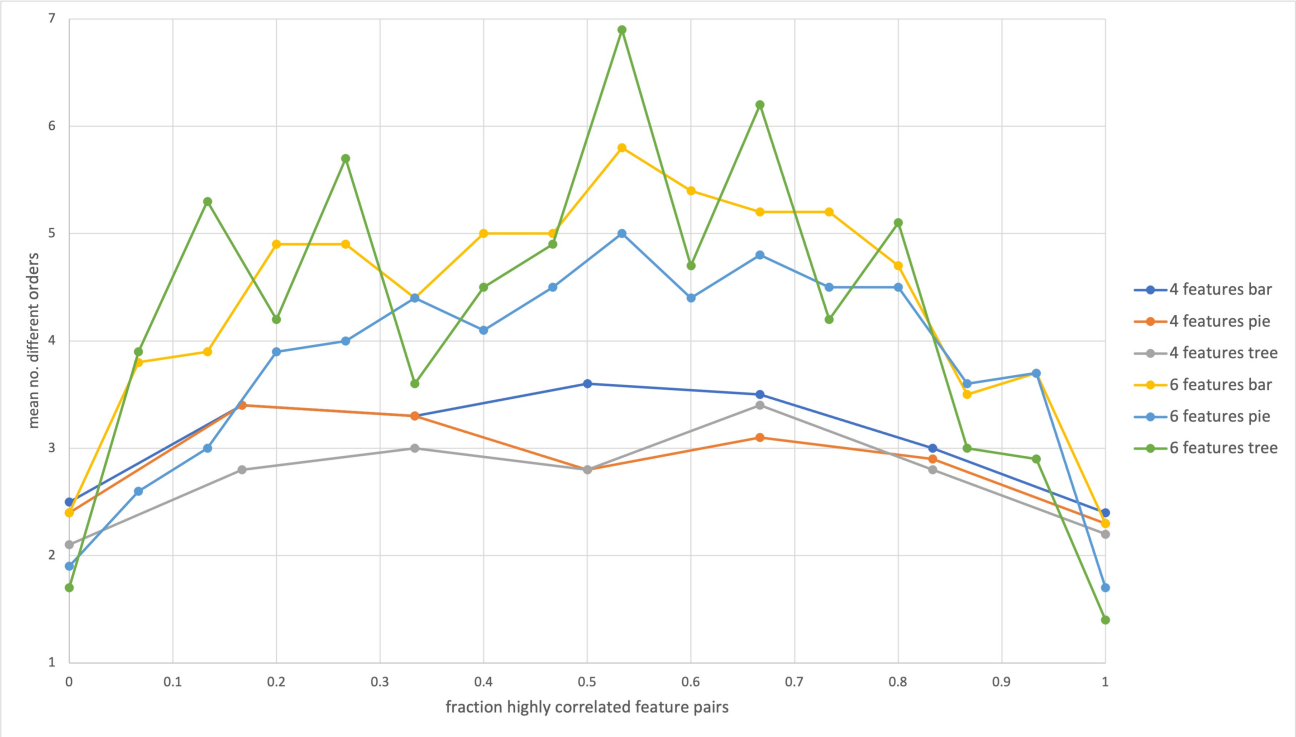

Figure 7: Number of different orders found, with the defined measures, for each method for 4 features and 6 features. Fraction highly correlated feature pairs is the amount of highly correlated feature pairs divided by the maximum possible amount of highly correlated pairs. n=10 for the mean number of different orders.

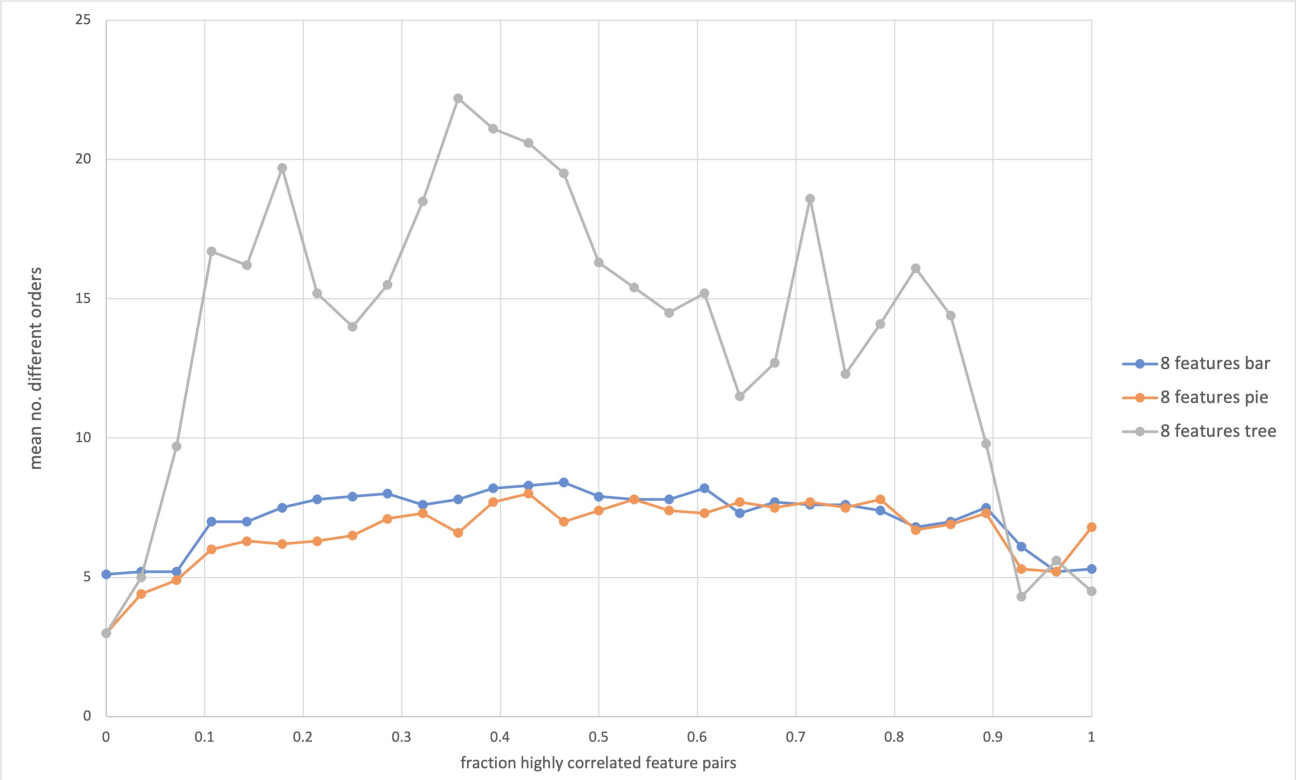

Figure 8: Number of different orders found, with the defined measures, for each method for 8 features. Fraction highly correlated feature pairs is the amount of highly correlated feature pairs divided by the maximum possible amount of highly correlated pairs. n=10 for the mean number of different orders.

of a task to identify them. For the bar and pie charts method, not many mistakes were made. Also, missing a highly correlated pair does not always have to be a bad thing. If two highly correlated features have a really low importance score, removing one of them will not heavily increase the importance of the other feature because in our setup the increase in importance is based on the importance of the feature that is excluded. Furthermore, we do have to note here that in our approach the importance of each feature is increased if a correlated feature is dropped, even though there could be another highly correlated feature that still holds the same information. So in these situations where interactions between features are more complex, more mistakes would be made since the changes in importance scores would be less apparent than in our artificial situation.

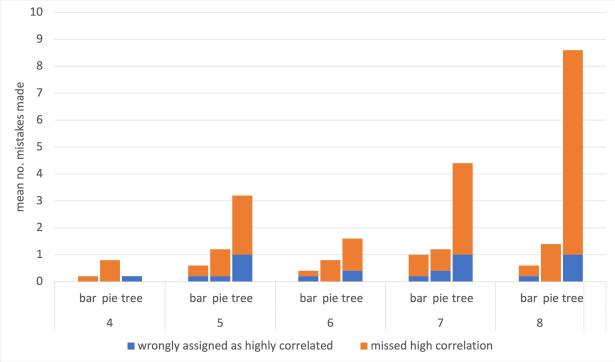

Figure 10: Amount of mistakes made when trying to figure out which of the features were highly correlated using the different visualization methods. n = 5 with the number of highly correlated features randomly chosen.

6. CONCLUSIONS

Here we have defined three different methods for visualizing feature importance dependencies. The first is a bar chart method that excludes every feature once and displays what the effect is on the other features. Secondly, a method that displays feature importances in pie charts and decides which features to exclude first based on the correlation coefficients between features. Thirdly, a method that constructs a tree where the root node is the most important feature without exclusion. Each node that branches off to the right shows the most important feature if the features above it are excluded and each node that branches off to the left shows the most important feature if all features above it are also excluded from training.

For a number of features ranging from four to six, these three methods were able to effectively visualize the importance order and scores that one would get from performing feature importance analysis if no feature is excluded. For a higher number of features, the visualizations may become too crowded, and/or the size of the visualization would become so large that it becomes impractical to effectively analyze the visualizations.

The three visualizations showed a higher number of importance orders when the fraction of highly correlated features increased from 0 until 0.5 and decreased again from 0.5 to 1. This shows that visualizations are able to display dependencies in relation to feature importance. When the number of features was eight, the tree method displays more feature orders than the other methods. However, for this tree method, it is harder to identify which features are dependent based on these different feature orders that are displayed by the method.

To conclude, at a number of features of six and below, all three visualization methods were able to effectively display feature importance dependencies in our artificial setup. When the number of features was eight, the tree method was not effective for determining which features were dependent by looking at the changes in feature importance orders and scores. The bar- and pie-chart methods were similar in effectiveness according to our evaluation, but we would argue that the pie-charts method may be able to display more information since, in contrast to the bar chart method, it displays cases where multiple features are excluded at the same time. So, the visualizations are able to show feature importance dependencies in an artificial setting, and hence can be an effective associated visualization for feature importance analysis, which can help us in showing how dependencies affect feature importances.

7. FUTURE WORK

The visualizations have only been evaluated on synthetic data, so we want to also evaluate them on real data sets to see how effective they are there. Furthermore, we took a naive approach when increasing feature importances when removing a correlated feature. We want to evaluate how effective the visualizations would be if we take into account all interactions between other features if we remove a feature. For example, if three features are highly correlated, excluding one of them would lead to less increase in importance for the other two features than if only two features are highly correlated. Also, we want to investigate how we can improve the visualization methods so that they are easy to analyze when the feature size increases further. Finally, we would like to evaluate another variant of the tree method. In this variant, a certain node would not be the most important if all features above it are excluded. Instead, it would be the most important feature if all features above it that branched off to the same side as their parent, are excluded.

8. REFERENCES

[1] L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 10 2001.

[2] G. Z. Espinoza, R. M. Angelo, P. R. Oliveira, and K. M. Honorio. Evaluating Deep Learning models for predicting ALK-5 inhibition. *PLOS ONE*, 16(1):e0246126, 1 2021.

[3] G. E. Jergensen, A. McGovern, R. Lagerquist, and T. Smith. Classifying convective storms using machine learning. *Weather and Forecasting*, 35(2):537–559, 4 2020.

[4] S. Lall, D. Sinha, A. Ghosh, D. Sengupta, and S. Bandyopadhyay. Stable feature selection using copula based mutual information: Feature selection using copula. *Pattern Recognition*, 112:107697, 4 2021.

[5] M. Nason, S. Emerson, and M. Leblanc. CARTscans: A Tool for Visualizing Complex Models. *Journal of Computational and Graphical Statistics*, 13(4):807–825, 2004.

[6] H. Peng, F. Long, and C. Ding. Feature selection based on mutual information: Criteria of Max-Dependency, Max-Relevance, and Min-Redundancy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(8):1226–1238, 2005.

[7] M. T. Ribeiro, S. Singh, and C. 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*, New York, NY, USA, 2016. ACM.

[8] P. Wei, Z. Lu, and J. Song. Variable importance analysis: A comprehensive review. *Reliability Engineering and System Safety*, 142:399–432, 7 2015.

[9] C. Xu. Decoupling correlated and uncorrelated parametric uncertainty contributions for nonlinear models. *Applied Mathematical Modelling*, 37(24):9950–9969, 12 2013.

[10] C. Xu and G. Z. Gertner. Uncertainty and sensitivity analysis for models with correlated parameters. *Reliability Engineering and System Safety*, 93(10):1563–1573, 10 2008.