# 5.9 Shapley Values

A prediction can be explained by assuming that each feature value of the instance is a "player" in a game where the prediction is the payout. Shapley values -- a method from coalitional game theory -- tells us how to fairly distribute the "payout" among the features.

> Interested in an in-depth, hands-on course on SHAP and Shapley values? Head over to the Shapley course page and get notified once the course is available.

## 5.9.1 General Idea

Assume the following scenario:

You have trained a machine learning model to predict apartment prices. For a certain apartment it predicts €300,000 and you need to explain this prediction. The apartment has an area of 50 $m^2$, is located on the 2nd floor, has a park nearby and cats are banned:

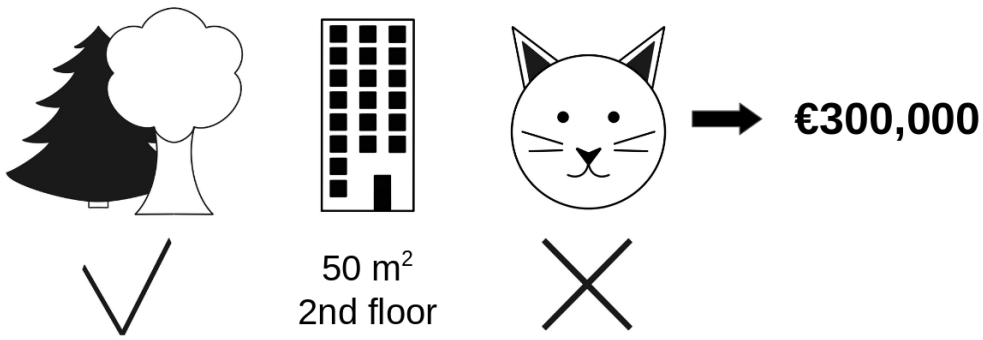


FIGURE 5.43: The predicted price for a 50 $m^2$ 2nd floor apartment with a nearby park and cat ban is €300,000. Our goal is to explain how each of these feature values contributed to the prediction.

The average prediction for all apartments is €310,000. How much has each feature value contributed to the prediction compared to the average prediction?

The answer is simple for linear regression models. The effect of each feature is the weight of the feature times the feature value. This only works because of the linearity of the model. For more complex models, we need a different solution. For example, LIME suggests local models to estimate effects. Another solution comes from cooperative game

theory: The Shapley value, coined by Shapley (1953)[42], is a method for assigning payouts to players depending on their contribution to the total payout. Players cooperate in a coalition and receive a certain profit from this cooperation.

Players? Game? Payout? What is the connection to machine learning predictions and interpretability? The "game" is the prediction task for a single instance of the dataset. The "gain" is the actual prediction for this instance minus the average prediction for all instances. The "players" are the feature values of the instance that collaborate to receive the gain (= predict a certain value). In our apartment example, the feature values `park-nearby`, `cat-banned`, `area-50` and `floor-2nd` worked together to achieve the prediction of €300,000. Our goal is to explain the difference between the actual prediction (€300,000) and the average prediction (€310,000): a difference of -€10,000.

The answer could be: The `park-nearby` contributed €30,000; `area-50` contributed €10,000; `floor-2nd` contributed €0; `cat-banned` contributed -€50,000. The contributions add up to -€10,000, the final prediction minus the average predicted apartment price.

**How do we calculate the Shapley value for one feature?**

The Shapley value is the average marginal contribution of a feature value across all possible coalitions. All clear now?

In the following figure we evaluate the contribution of the `cat-banned` feature value when it is added to a coalition of `park-nearby` and `area-50`. We simulate that only `park-nearby`, `cat-banned` and `area-50` are in a coalition by randomly drawing another apartment from the data and using its value for the floor feature. The value `floor-2nd` was replaced by the randomly drawn `floor-1st`. Then we predict the price of the apartment with this combination (€310,000). In a second step, we remove `cat-banned` from the coalition by replacing it with a random value of the cat allowed/banned feature from the randomly drawn apartment. In the example it was `cat-allowed`, but it could have been `cat-banned` again. We predict the apartment price for the coalition of `park-nearby` and `area-50` (€320,000). The contribution of `cat-banned` was €310,000 - €320,000 = -€10,000. This estimate depends on the values of the randomly drawn apartment that served as a "donor" for the cat and floor feature values. We will get better estimates if we repeat this sampling step and average the contributions.

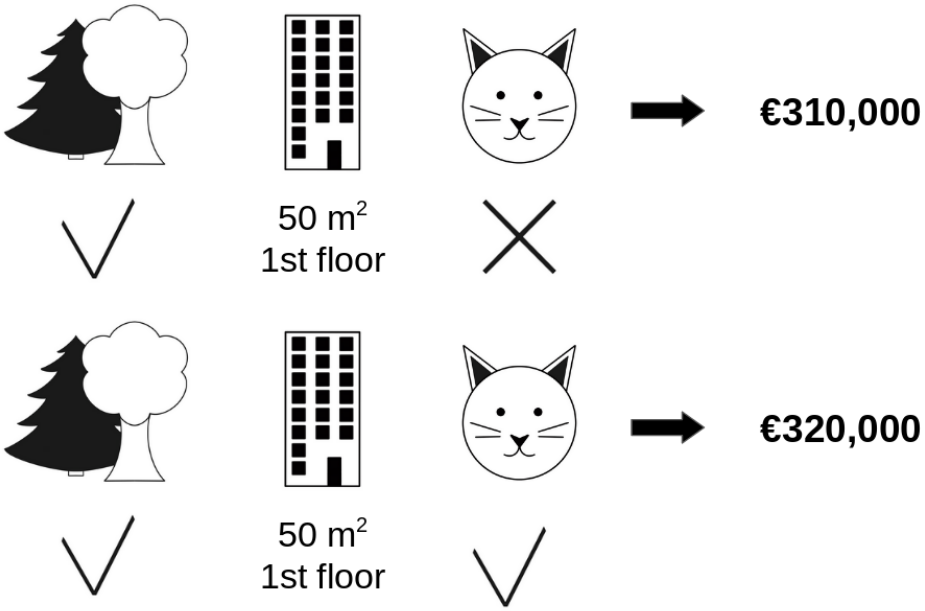


**FIGURE 5.44: One sample repetition to estimate the contribution of `cat-banned` to the prediction when added to the coalition of `park-nearby` and `area-50`.**

We repeat this computation for all possible coalitions. The Shapley value is the average of all the marginal contributions to all possible coalitions. The computation time increases exponentially with the number of features. One solution to keep the computation time manageable is to compute contributions for only a few samples of the possible coalitions.

The following figure shows all coalitions of feature values that are needed to determine the Shapley value for `cat-banned`. The first row shows the coalition without any feature values. The second, third and fourth rows show different coalitions with increasing coalition size, separated by "|". All in all, the following coalitions are possible:

- `No feature values`
- `park-nearby`
- `area-50`
- `floor-2nd`
- `park-nearby + area-50`
- `park-nearby + floor-2nd`
- `area-50 + floor-2nd`
- `park-nearby + area-50 + floor-2nd`.

For each of these coalitions we compute the predicted apartment price with and without the feature value `cat-banned` and take the difference to get the marginal contribution. The Shapley value is the (weighted) average of marginal contributions. We replace the feature values of features that are not in a coalition with random feature values from the apartment dataset to get a prediction from the machine learning model.

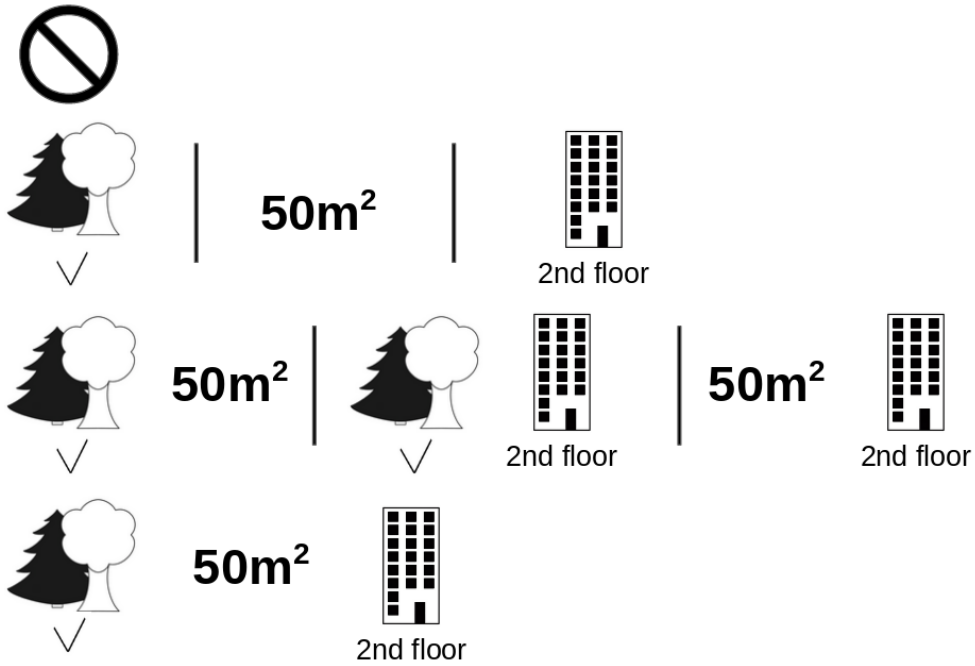


**FIGURE 5.45: All 8 coalitions needed for computing the exact Shapley value of the `cat-banned` feature value.**

If we estimate the Shapley values for all feature values, we get the complete distribution of the prediction (minus the average) among the feature values.

## 5.9.2 Examples and Interpretation

The interpretation of the Shapley value for feature value j is: The value of the j-th feature contributed $\phi_j$ to the prediction of this particular instance compared to the average prediction for the dataset.

The Shapley value works for both classification (if we are dealing with probabilities) and regression.

We use the Shapley value to analyze the predictions of a random forest model predicting cervical cancer:

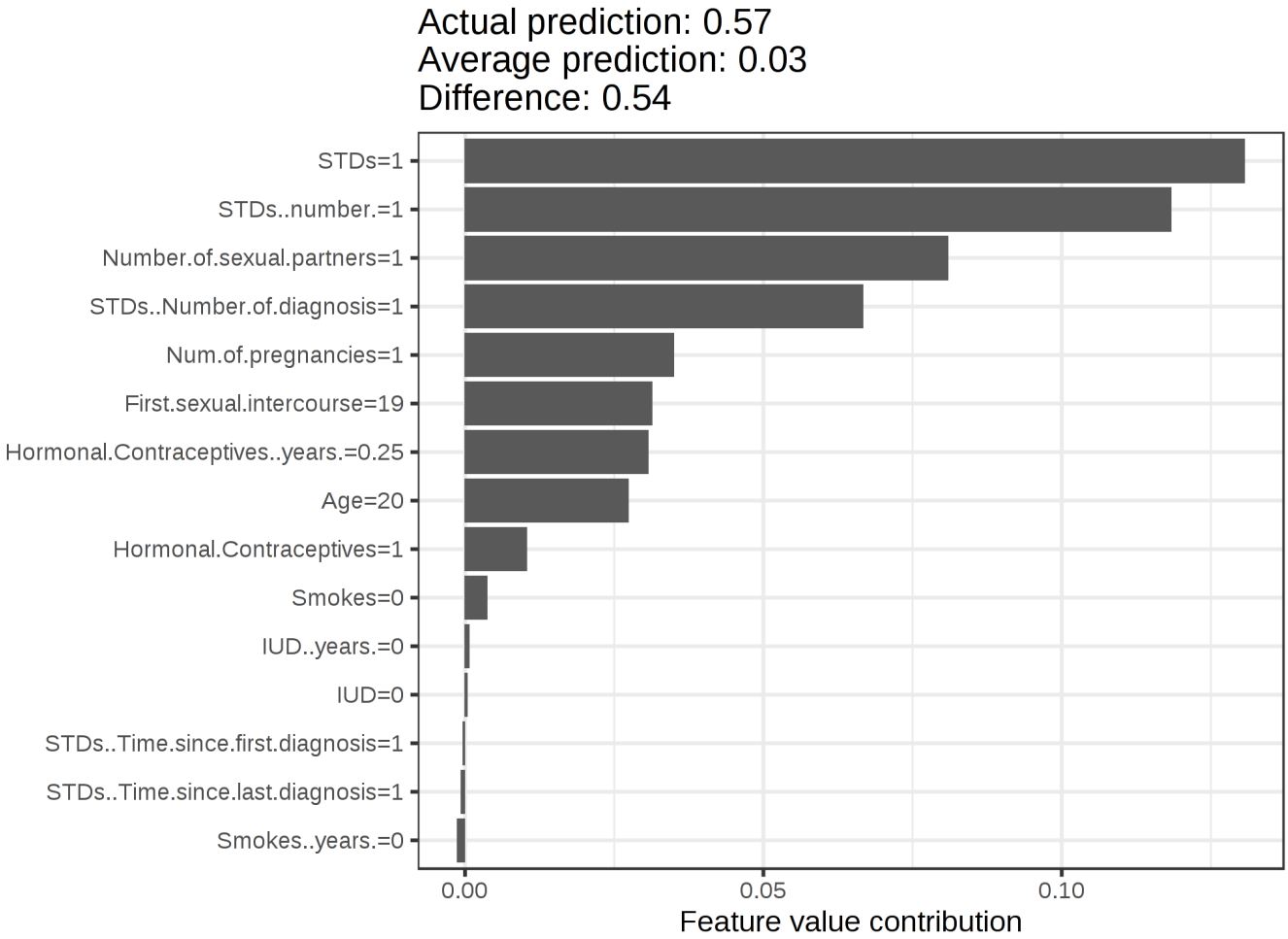


**FIGURE 5.46: Shapley values for a woman in the cervical cancer dataset. With a prediction of 0.57, this woman's cancer probability is 0.54 above the average prediction of 0.03. The number of diagnosed STDs increased the probability the most. The sum of contributions yields the difference between actual and average prediction (0.54).**

For the bike rental dataset, we also train a random forest to predict the number of rented bikes for a day, given weather and calendar information. The explanations created for the random forest prediction of a particular day:

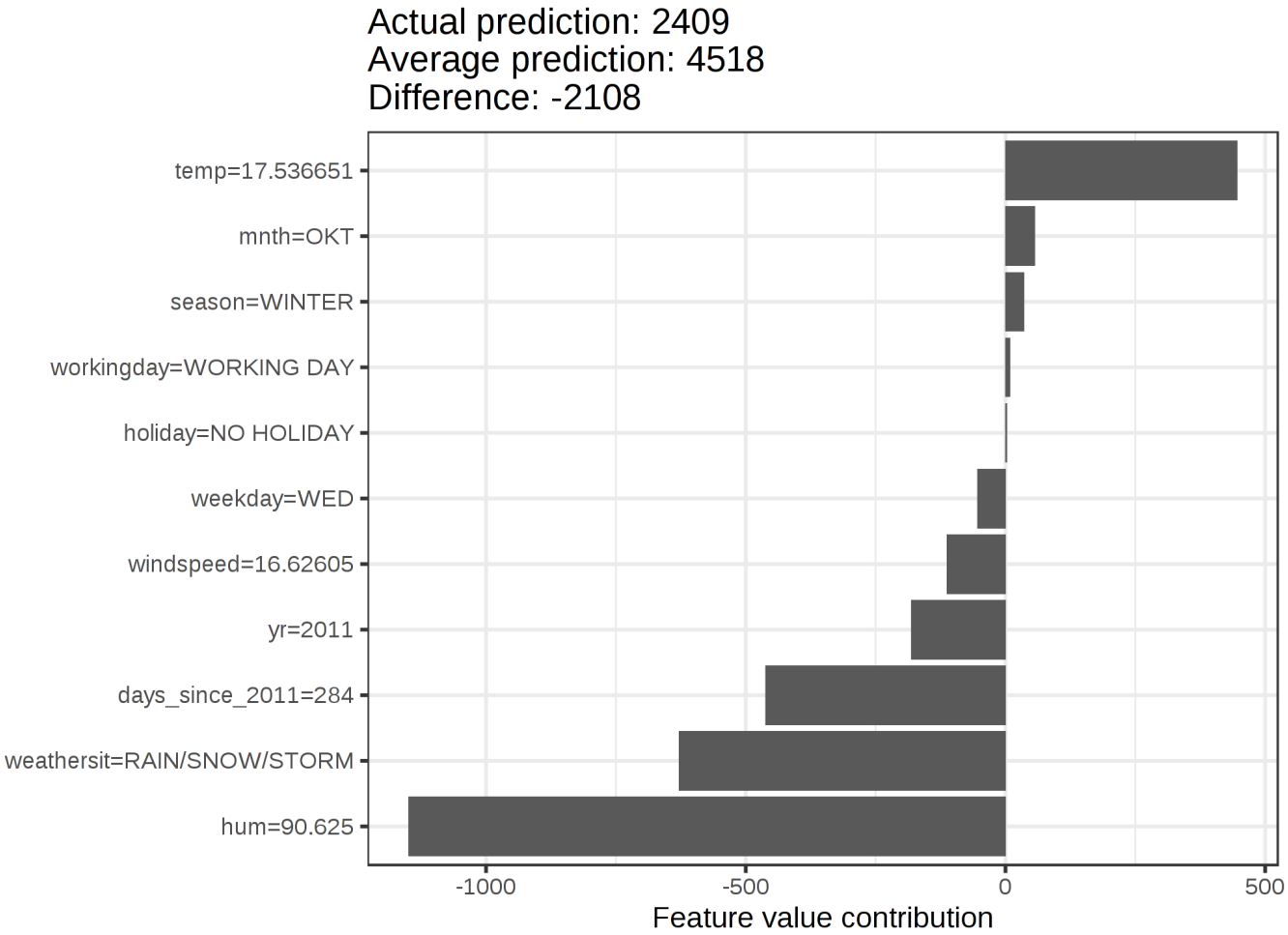


**FIGURE 5.47: Shapley values for day 285. With a predicted 2409 rental bikes, this day is -2108 below the average prediction of 4518. The weather situation and humidity had the largest negative contributions. The temperature on this day had a positive contribution. The sum of Shapley values yields the difference of actual and average prediction (-2108).**

Be careful to interpret the Shapley value correctly: The Shapley value is the average contribution of a feature value to the prediction in different coalitions. The Shapley value is NOT the difference in prediction when we would remove the feature from the model.

## 5.9.3 The Shapley Value in Detail

This section goes deeper into the definition and computation of the Shapley value for the curious reader. Skip this section and go directly to "Advantages and Disadvantages" if you are not interested in the technical details.

We are interested in how each feature affects the prediction of a data point. In a linear model it is easy to calculate the individual effects. Here is what a linear model prediction looks like for one data instance:

$$\hat{f}(x) = \beta_0 + \beta_1 x_1 + \ldots + \beta_p x_p$$

where x is the instance for which we want to compute the contributions. Each $x_j$ is a feature value, with j = 1,...,p. The $\beta_j$ is the weight corresponding to feature j.

The contribution $\phi_j$ of the j-th feature on the prediction $\hat{f}(x)$ is:

$$\phi_j(\hat{f}) = \beta_j x_j - E(\beta_j X_j) = \beta_j x_j - \beta_j E(X_j)$$

where $E(\beta_j X_j)$ is the mean effect estimate for feature j. The contribution is the difference between the feature effect minus the average effect. Nice! Now we know how much each feature contributed to the prediction. If we sum all the feature contributions for one instance, the result is the following:

$$\begin{aligned}\sum_{j=1}^{p} \phi_j(\hat{f}) &= \sum_{j=1}^{p}(\beta_j x_j - E(\beta_j X_j)) \\ &= (\beta_0 + \sum_{j=1}^{p} \beta_j x_j) - (\beta_0 + \sum_{j=1}^{p} E(\beta_j X_j)) \\ &= \hat{f}(x) - E(\hat{f}(X))\end{aligned}$$

This is the predicted value for the data point x minus the average predicted value. Feature contributions can be negative.

Can we do the same for any type of model? It would be great to have this as a model-agnostic tool. Since we usually do not have similar weights in other model types, we need a different solution.

Help comes from unexpected places: cooperative game theory. The Shapley value is a solution for computing feature contributions for single predictions for any machine learning model.

### 5.9.3.1 The Shapley Value

The Shapley value is defined via a value function val of players in S.

The Shapley value of a feature value is its contribution to the payout, weighted and summed over all possible feature value combinations:

$$\phi_j(val) = \sum_{S \subseteq \{x_1,\ldots,x_p\} \setminus \{x_j\}} \frac{|S|!\,(p - |S| - 1)!}{p!}\left(val\left(S \cup \{x_j\}\right) - val(S)\right)$$

where S is a subset of the features used in the model, x is the vector of feature values of the instance to be explained and p the number of features. $val_x(S)$ is the prediction for feature values in set S that are marginalized over features that are not included in set S:

$$val_x(S) = \int \hat{f}(x_1, \ldots, x_p) d\mathbb{P}_{x \notin S} - E_X(\hat{f}(X))$$

You actually perform multiple integrations for each feature that is not contained S. A concrete example: The machine learning model works with 4 features x1, x2, x3 and x4 and we evaluate the prediction for the coalition S consisting of feature values x1 and x3:

$$val_x(S) = val_x(\{x_1, x_3\}) = \int_{\mathbb{R}} \int_{\mathbb{R}} \hat{f}(x_1, X_2, x_3, X_4) d\mathbb{P}_{X_2 X_4} - E_X(\hat{f}(X))$$

This looks similar to the feature contributions in the linear model!

Do not get confused by the many uses of the word "value": The feature value is the numerical or categorical value of a feature and instance; the Shapley value is the feature contribution to the prediction; the value function is the payout function for coalitions of players (feature values).

The Shapley value is the only attribution method that satisfies the properties **Efficiency**, **Symmetry**, **Dummy** and **Additivity**, which together can be considered a definition of a fair payout.

**Efficiency** The feature contributions must add up to the difference of prediction for x and the average.

$$\sum_{j=1}^{p} \phi_j = \hat{f}(x) - E_X(\hat{f}(X))$$

**Symmetry** The contributions of two feature values j and k should be the same if they contribute equally to all possible coalitions. If

$$val(S \cup \{x_j\}) = val(S \cup \{x_k\})$$

for all

$$S \subseteq \{x_1, \ldots, x_p\} \setminus \{x_j, x_k\}$$

then

$$\phi_j = \phi_k$$

**Dummy** A feature j that does not change the predicted value -- regardless of which coalition of feature values it is added to -- should have a Shapley value of 0. If

$$val(S \cup \{x_j\}) = val(S)$$

for all

$$S \subseteq \{x_1, \ldots, x_p\}$$

then

$$\phi_j = 0$$

**Additivity** For a game with combined payouts val+val$^+$ the respective Shapley values are as follows:

$$\phi_j + \phi_j^+$$

Suppose you trained a random forest, which means that the prediction is an average of many decision trees. The Additivity property guarantees that for a feature value, you can calculate the Shapley value for each tree individually, average them, and get the Shapley value for the feature value for the random forest.

### 5.9.3.2 Intuition

An intuitive way to understand the Shapley value is the following illustration: The feature values enter a room in random order. All feature values in the room participate in the game (= contribute to the prediction). The Shapley value of a feature value is the average change in the prediction that the coalition already in the room receives when the feature value joins them.

### 5.9.3.3 Estimating the Shapley Value

All possible coalitions (sets) of feature values have to be evaluated with and without the j-th feature to calculate the exact Shapley value. For more than a few features, the exact solution to this problem becomes problematic as the number of possible coalitions exponentially increases as more features are added. Strumbelj et al. (2014)[43] propose an approximation with Monte-Carlo sampling:

$$\hat{\phi}_j = \frac{1}{M}\sum_{m=1}^{M}\left(\hat{f}\left(x_{+j}^m\right) - \hat{f}\left(x_{-j}^m\right)\right)$$

where $\hat{f}\left(x_{+j}^m\right)$ is the prediction for x, but with a random number of feature values replaced by feature values from a random data point z, except for the respective value of feature j. The x-vector $x_{-j}^m$ is almost identical to $x_{+j}^m$, but the value $x_j^m$ is also taken from the sampled z. Each of these M new instances is a kind of "Frankenstein Monster" assembled from two instances.

**Approximate Shapley estimation for single feature value:**

- Output: Shapley value for the value of the j-th feature

- Required: Number of iterations M, instance of interest x, feature index j, data matrix X, and machine learning model f
- For all m = 1,...,M:
  - Draw random instance z from the data matrix X
  - Choose a random permutation o of the feature values
  - Order instance x: $x_o = (x_{(1)}, \ldots, x_{(j)}, \ldots, x_{(p)})$
  - Order instance z: $z_o = (z_{(1)}, \ldots, z_{(j)}, \ldots, z_{(p)})$
  - Construct two new instances
    - With feature j: $x_{+j} = (x_{(1)}, \ldots, x_{(j-1)}, x_{(j)}, z_{(j+1)}, \ldots, z_{(p)})$
    - Without feature j: $x_{-j} = (x_{(1)}, \ldots, x_{(j-1)}, z_{(j)}, z_{(j+1)}, \ldots, z_{(p)})$
  - Compute marginal contribution: $\phi_j^m = \hat{f}(x_{+j}) - \hat{f}(x_{-j})$
- Compute Shapley value as the average: $\phi_j(x) = \frac{1}{M}\sum_{m=1}^{M}\phi_j^m$

First, select an instance of interest x, a feature j and the number of iterations M. For each iteration, a random instance z is selected from the data and a random order of the features is generated. Two new instances are created by combining values from the instance of interest x and the sample z. The instance $x_{+j}$ is the instance of interest, but all values in the order before feature j are replaced by feature values from the sample z. The instance $x_{-j}$ is the same as $x_{+j}$, but in addition has feature j replaced by the value for feature j from the sample z. The difference in the prediction from the black box is computed:

$$\phi_j^m = \hat{f}(x_{+j}^m) - \hat{f}(x_{-j}^m)$$

All these differences are averaged and result in:

$$\phi_j(x) = \frac{1}{M}\sum_{m=1}^{M}\phi_j^m$$

Averaging implicitly weighs samples by the probability distribution of X.

The procedure has to be repeated for each of the features to get all Shapley values.

## 5.9.4 Advantages

The difference between the prediction and the average prediction is **fairly distributed** among the feature values of the instance -- the Efficiency property of Shapley values. This property distinguishes the Shapley value from other methods such as LIME. LIME does not guarantee that the prediction is fairly distributed among the features. The Shapley value might be the only method to deliver a full explanation. In situations where the law requires

explainability -- like EU's "right to explanations" -- the Shapley value might be the only legally compliant method, because it is based on a solid theory and distributes the effects fairly. I am not a lawyer, so this reflects only my intuition about the requirements.

The Shapley value allows **contrastive explanations.** Instead of comparing a prediction to the average prediction of the entire dataset, you could compare it to a subset or even to a single data point. This contrastiveness is also something that local models like LIME do not have.

The Shapley value is the only explanation method with a **solid theory**. The axioms -- efficiency, symmetry, dummy, additivity -- give the explanation a reasonable foundation. Methods like LIME assume linear behavior of the machine learning model locally, but there is no theory as to why this should work.

It is mind-blowing to **explain a prediction as a game** played by the feature values.

## 5.9.5 Disadvantages

The Shapley value requires **a lot of computing time.** In 99.9% of real-world problems, only the approximate solution is feasible. An exact computation of the Shapley value is computationally expensive because there are $2^k$ possible coalitions of the feature values and the "absence" of a feature has to be simulated by drawing random instances, which increases the variance for the estimate of the Shapley values estimation. The exponential number of the coalitions is dealt with by sampling coalitions and limiting the number of iterations M. Decreasing M reduces computation time, but increases the variance of the Shapley value. There is no good rule of thumb for the number of iterations M. M should be large enough to accurately estimate the Shapley values, but small enough to complete the computation in a reasonable time. It should be possible to choose M based on Chernoff bounds, but I have not seen any paper on doing this for Shapley values for machine learning predictions.

The Shapley value **can be misinterpreted.** The Shapley value of a feature value is not the difference of the predicted value after removing the feature from the model training. The interpretation of the Shapley value is: Given the current set of feature values, the contribution of a feature value to the difference between the actual prediction and the mean prediction is the estimated Shapley value.

The Shapley value is the wrong explanation method if you seek sparse explanations (explanations that contain few features). Explanations created with the Shapley value method **always use all the features.** Humans prefer selective explanations, such as those

produced by LIME. LIME might be the better choice for explanations lay-persons have to deal with. Another solution is SHAP introduced by Lundberg and Lee (2016)[44], which is based on the Shapley value, but can also provide explanations with few features.

The Shapley value returns a simple value per feature, but **no prediction model** like LIME. This means it cannot be used to make statements about changes in prediction for changes in the input, such as: "If I were to earn €300 more a year, my credit score would increase by 5 points."

Another disadvantage is that **you need access to the data** if you want to calculate the Shapley value for a new data instance. It is not sufficient to access the prediction function because you need the data to replace parts of the instance of interest with values from randomly drawn instances of the data. This can only be avoided if you can create data instances that look like real data instances but are not actual instances from the training data.

Like many other permutation-based interpretation methods, the Shapley value method suffers from **inclusion of unrealistic data instances** when features are correlated. To simulate that a feature value is missing from a coalition, we marginalize the feature. This is achieved by sampling values from the feature's marginal distribution. This is fine as long as the features are independent. When features are dependent, then we might sample feature values that do not make sense for this instance. But we would use those to compute the feature's Shapley value. One solution might be to permute correlated features together and get one mutual Shapley value for them. Another adaptation is conditional sampling: Features are sampled conditional on the features that are already in the team. While conditional sampling fixes the issue of unrealistic data points, a new issue is introduced: The resulting values are no longer the Shapley values to our game, since they violate the symmetry axiom, as found out by Sundararajan et. al (2019)[45] and further discussed by Janzing et. al (2020)[46].

## 5.9.6 Software and Alternatives

Shapley values are implemented in both the `iml` and fastshap packages for R.

SHAP, an alternative estimation method for Shapley values, is presented in the next chapter.

Another approach is called breakDown, which is implemented in the `breakDown` R package[47]. BreakDown also shows the contributions of each feature to the prediction, but computes them step by step. Let us reuse the game analogy: We start with an empty team,

add the feature value that would contribute the most to the prediction and iterate until all feature values are added. How much each feature value contributes depends on the respective feature values that are already in the "team", which is the big drawback of the breakDown method. It is faster than the Shapley value method, and for models without interactions, the results are the same.


42. Shapley, Lloyd S. "A value for n-person games." Contributions to the Theory of Games 2.28 (1953): 307-317.↩

43. Štrumbelj, Erik, and Igor Kononenko. "Explaining prediction models and individual predictions with feature contributions." Knowledge and information systems 41.3 (2014): 647-665.↩

44. Lundberg, Scott M., and Su-In Lee. "A unified approach to interpreting model predictions." Advances in Neural Information Processing Systems. 2017.↩

45. Sundararajan, Mukund, and Amir Najmi. "The many Shapley values for model explanation." arXiv preprint arXiv:1908.08474 (2019).↩

46. Janzing, Dominik, Lenon Minorics, and Patrick Blöbaum. "Feature relevance quantification in explainable AI: A causal problem." International Conference on Artificial Intelligence and Statistics. PMLR, 2020.↩

47. Staniak, Mateusz, and Przemyslaw Biecek. "Explanations of model predictions with live and breakDown packages." arXiv preprint arXiv:1804.01955 (2018).↩