Limitations of Interpretable Machine Learning Methods

Contents

List of Tables

List of Figures

Preface

This project explains the limitations of current approaches in interpretable machine learning, such as partial dependence plots (PDP, Accumulated Local Effects (ALE), permutation feature importance, leave-one-covariate out (LOCO) and local interpretable model-agnostic explanations (LIME). All of those methods can be used to explain the behavior and predictions of trained machine learning models. The interpretation methods might not work well in the following cases:

- if a model models interactions (e.g. when a random forest is used)
- if features strongly correlate with each other
- if the model does not correctly model causal relationships
- if parameters of the interpretation method are not set correctly

FIGURE 1: Creative Commons License

‘‘‘

Structure of the book

TODO

[1] http://creativecommons.org/licenses/by-nc-sa/4.0/

1

Introduction

TODO

2

Introduction to Partial Dependence Plots (PDP) and Individual Conditional Expectation (ICE)

2.1 Partial Dependence Plots (PDP)

The Partial Dependence Plot (PDP) is a rather intuitive and easy-to-understand visualization of the features' impact on the predicted outcome. If the assumptions for the PDP are met, it can show the way a feature impacts an outcome variable. More precisely, mapping the marginal effect of the selected variable(s) uncovers the linear, monotonic or nonlinear relationship between the predicted response and the individual feature variable(s).(Molnar, 2019)

The underlying function can be described as follows:

Let x_S be the set of features of interest for the PDP and x_C the complement set which contains all other features. While the general model function $f(x) = f(x_S, x_C)$ depends on all input variables, the partial dependence function marginalizes over the feature distribution in set C (Hastie et al., 2013):

$$f_{x_S}(x_S) = \mathbb{E}_{x_C}[f(x_S, x_C)]$$

The partial dependence function can be estimated by averaging the actual feature values of x_C in the training data at given values of x_S or, in other words, it computes the marginal effect of x_S on the prediction. In order to obtain realistic results, a major assumption of the PDP is that the features in x_S and x_C are independent and thus uncorrelated.(Hastie et al., 2013)

$$\hat{f}_{x_S}(x_S) = \frac{1}{n}\sum_{i=1}^{n} f(x_S, x_C^{(i)})$$

An example of a PDP based on the 'Titanic' data set, which contains information on the fate of 2224 passengers and crew members during the Titanic's maiden voyage, is given in figure 2.1.

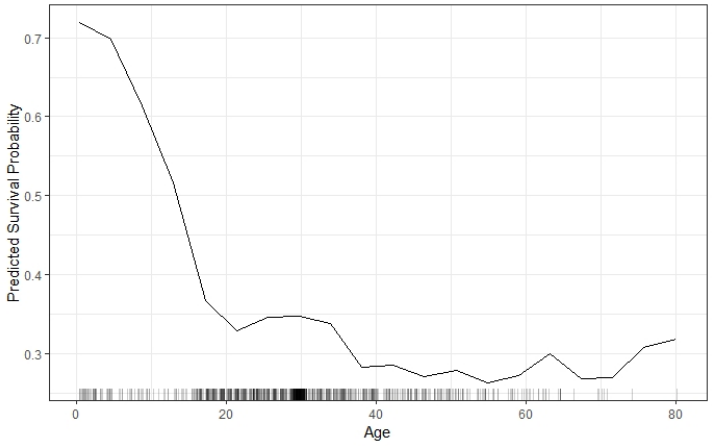


FIGURE 2.1: PDP for predicted survival probability and numeric feature variable 'Age'. The probability to survive sharply drops at a young age and more moderately afterwards. The rug on the x-axis illustrates the distribution of observed training data.

When a feature is categorical, rather than continuous, the partial dependence function is modeled separately for all of the K different classes of said feature. It maps the predictions for each respective class at given feature values of x_S.(Hastie et al., 2013)

For such categorical features, the partial dependence function and the resulting plot are produced by replacing all observed x_S-values with the respective category and averaging the predictions. This procedure is repeated for each of the features' categories.(Molnar, 2019) As an example, figure 2.2 shows the partial dependence for the survival probability prediction for passengers on the Titanic and the categorical feature 'passenger class'.

2.1.1 Advantages and Limitations of Partial Dependence Plots

Partial Dependence Plots are easy to compute and a popular way to explain insights from black box Machine Learning models. With their intuitive character, PDPs are perfect for communicating to a non-technical audience. However, due to limited visualization techniques and the restriction of human perception to a maximum of three dimensions, only one or two features can reasonably be displayed in one PDP.(Molnar, 2019) 2.3 shows that the combination of one numerical (Age) and one categorical (Sex) feature still allows rather precise interpretation. The combination of two numerical features (Age & Fare) still works, but already degrades the interpretability with its colour intensity scale as shown in figure 2.4.

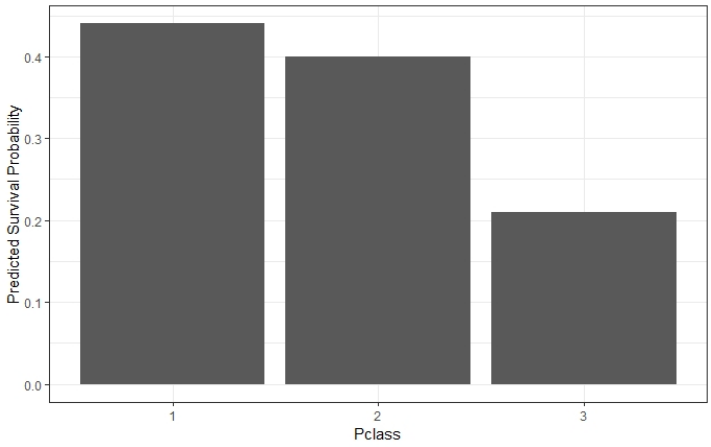


FIGURE 2.2: The PDP for survival probability and categorical feature ' passenger class' reveals that passengers in lower classes had a lower probability to survive than those in a higher class.

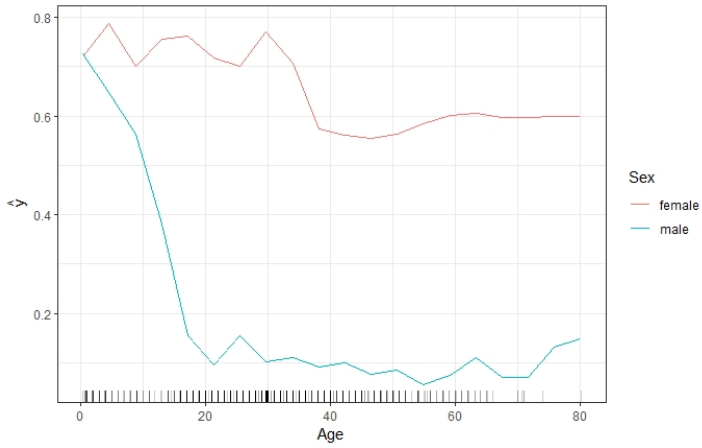


FIGURE 2.3: Two-dimensional PDP for predicted survival probability and numerical feature 'Age', together with the categorical feature 'Sex'. The PDP shows that while the survival probability for both genders declines as age increases, there is a difference between genders. It is clear that the decrease is much steeper for males.

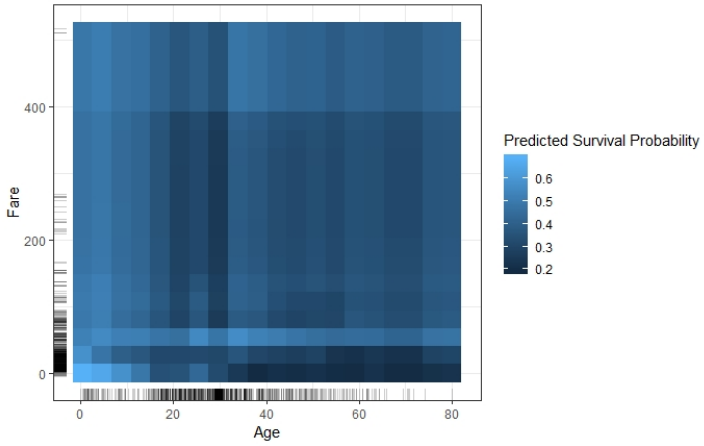


FIGURE 2.4: Two-dimensional PDP for predicted survival probability and numerical features 'Age' and 'Fare'. The PDP illustrates that the survival probability of younger passengers is fairly uniform for varying fares, while adults travelling at a lower fare also had a much lower probability to survive compared to those that paid a high fare.

Drawing a PDP with one or two feature variables allows a straight-forward interpretation of the marginal effects. This holds true as long as the features are not correlated. Should this independence assumption be violated, the partial dependence function will produce unrealistic data points. For instance, correlation between height and weight leading to a data point for someone taller than 2 meters weighing less than 50 kilos. Furthermore, opposite effects of heterogeneous subgroups might remain hidden through averaging the marginal effects, which could lead to wrong conclusions.(Molnar, 2019)

2.2 Individual Conditional Expectation Curves

While partial dependence plots provide the average effect of a feature, Individual Conditional Expectation (ICE) plots are a method to disaggregate these averages. ICE plots visualize the functional relationship between the predicted response and the feature separately for each instance. In other words, a PDP averages the individual lines of an ICE plot.(Molnar, 2019)

More formally, ICE plots can be derived by considering the estimated response function $\hat{f}$ and the observations $(x_S^{(i)}, x_C^{(i)})_{i=1}^N$. The curve $\hat{f}_S^{(i)}$ is plotted against

the observed values of $x_S^{(i)}$ for each of the observed instances while $x_C^{(i)}$ remains fixed at each point on the x-axis.(Molnar, 2019)(Goldstein et al., 2013)

As shown in figure 2.5, each line represents one instance and visualizes the effect of varying the feature value $x_S^{(i)}$ (Age) of a particular instance on the model's prediction, given all other features remain constant (c.p.). An ICE plot can highlight the variation in the fitted values across the range of a feature. This suggests where and to what extent heterogeneities might exist.

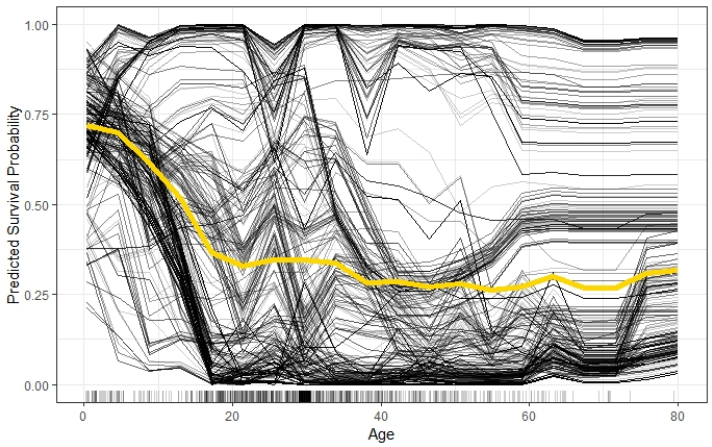


FIGURE 2.5: ICE plot of survival probability by Age. The yellow line represents the average of the individual lines and is thus equivalent to the respective PDP. The individual conditional relationships indicate that there might be underlying heterogeneity in the complement set.

2.2.1 Centered ICE Plot

If the curves of an ICE plot are stacked or have a wide range of intercepts it can be difficult to observe heterogeneity in the model. The so called centered ICE plot (c-ICE plot) is a simple solution to this problem. The curves are centered at a certain point in the feature and display only the difference in the prediction to this point.(Molnar, 2019) After anchoring a location x^a in the range of x_s and connecting all prediction lines at that point, the new curves are defined as:

$$\hat{f}_{cent}^{(i)} = \hat{f^{(i)}} - \mathbf{1}\hat{f}(x^a, x_C^{(i)})$$

Experience has shown that the most interpretable plots occur when the anchor point x^a is chosen as minimum or maximum of the observed values. Figure 2.6 shows the effect of centering the ICE curves of survival probability by Age at the minimum of observed ages in the 'Titanic' data set.

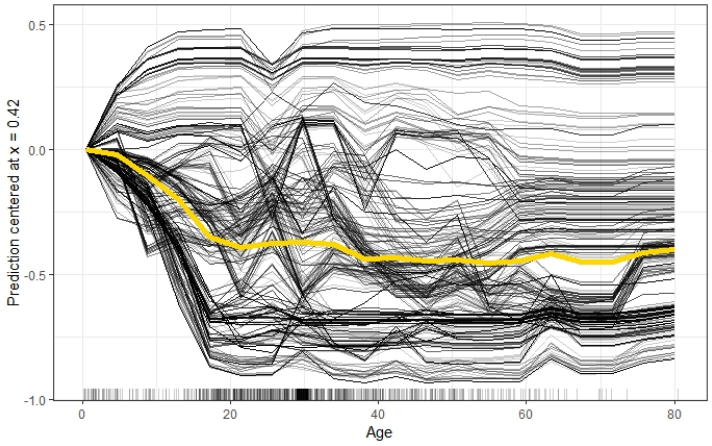


FIGURE 2.6: Centered ICE plot of survival probability by Age. All lines are fixed to 0 at the minimum observed age of 0.42. The y-axis shows the survival probability difference to age 0.42. Centrered ICE plot shows that compared to age 0.42, the predictions for most passengers decrease as age increases. However, there are quite a few passengers with opposite predictions.

2.2.2 Derivative ICE Plot

Another way to explore the heterogeneity is to show plots of the partial derivative of $\hat{f}$ with respect to x_s. Assume that x_s does not interact with the other predictors in the fitted model, the prediction function can be written as:

$$\hat{f}(x) = \hat{f}(x_s, x_C) = g(x_s) + h(x_C),$$

so that

$$\frac{\partial \hat{f}(\mathbf{x})}{\partial x_s} = g'(x_s)$$

When no interactions are present in the fitted model, all curves in the d-ICE plot are equivalent and the plot shows a single line. When interactions do exist, the derivative lines will be heterogeneous. As it can be difficult to visually assess derivatives from ICE plots, it is useful to plot an estimate of the partial derivative directly.(Goldstein et al., 2013)

2.2.3 Advantages and Limitations of ICE Plots

The major advantage of ICE plots is that they are even more intuitive than PDPs which enables data scientists to drill much deeper to explore individual differences. This may help to identify subgroups and interactions between model inputs. However, there are also some disadvantages of ICE plots. Firstly,

only one feature can be plotted in an ICE plot meaningfully. Otherwise, there would be a problem of overplotting and it would be hard to distinguish anything in the plot. Secondly, just like PDPs, ICE plots for correlated features may produce invalid data points. Finally, without additionally plotting the PDP it might be difficult to see the average in ICE plots.(Molnar, 2019)

3

PDP and Correlated Features

Author: Veronika Kronseder

3.1 Problem Description

As outlined in chapter 2, PDPs and ICE plots are meaningful graphical tools to visualize the impact of individual feature variables. This is particularly true for black box algorithms, where the mechanism of each feature and its influence on the generated predictions may be difficult to retrace. (Goldstein et al., 2013)
The reliability of the produced curves, however, strongly builds on the independence assumption of the features. Furthermore, results can be misleading in areas with no or little observations, where the curve is drawn as a result of extrapolation. In this chapter, we want to illustrate and discuss the issue of depencencies between different types of variables, missing values and the associated implications on marginalized feature effects with a particular focus on PDPs.

3.1.1 What is the issue with dependent features?

When looking at PDPs, one should bear in mind that by definition the partial dependence function does not reflect the isolated effect of x_S while the features in x_C are ignored. This approach would correspond to the conditional expectation $\tilde{f}_S(x_S) = \mathbb{E}_{x_C}[f(x_S, x_C)|x_S]$, which is only congruent to the partial dependence function $\hat{f}_{x_S}(x_S) = \mathbb{E}_{x_C}[f(x_S, x_C)]$ in case of x_S and x_C being independent. (Hastie et al., 2013)
Although unlikely in many practical applications, the independence of feature variables is one of the major assumptions to produce meaningful PDPs. Its violation would mean that, when calculating averages of the features in x_C, the estimated partial dependence function $\hat{f}_{x_S}(x_S)$ takes unrealistic data points into consideration. (**?**)
FIGURE 1 illustrates the problem by contrasting the common distribution of the independent features x_1 and x_2 on the left with an example where the two

features have a strong linear dependency, and thus are highly correlated, on the right. When computing the PDP for feature x_1, we take x_2 into account by calculating its mean (here: $\overline{x}_2 \approx 0$) and keeping the value in $\hat{f}_{x_S}(x_1)$ constant at any point on the x-axis. This makes sense in the independent case, where observations are randomly scattered. However, when looking at the correlated features in the right part of FIGURE 1, the average of x_2 is not a realistic value in combination with x_1-values in the very left and the very right part of the feature distribution.

FIGURE 1

3.1.2 What is the issue with extrapolation?

Generally speaking, extrapolation means leaving the distribution of observed data. On the one hand, this can affect the predictions, namely in the event of the prediction function doing 'weird stuff' in unobserved areas. In chapter ?? we will see an example where this instant leads to a failure of the Partial Dependence Plots. (**?**) \ On the other hand, in PDPs are directly exposed to extrapolation problems due to the fact that the estimated partial dependence function $\hat{f}_{x_S}$ is evaluated at each observed $x_S^{(i)}$, giving a set of N ordered pairs: $\{(x_S^{(i)}, \hat{f}_{x_S^{(i)}})\}_{i=1}^N$. In the PDP, the resulting coordinates are plottet against each other and joined by lines. Not only outside the margins of observed values, but also in areas with a larger distance between neighboured x_S values, the indicated relationship with the target variable might be inappropriate and volatile in case of outliers. (Goldstein et al., 2013)

In FIGURE 2, a part of the previously simulated observations has been deleted from both the independent and the correlated example to visualize a data situation which might have an impact on the PDP in terms of extrapolation, which we will see in chapter 3.?. The shift in observed areas can also be noticed from the marginal distribution of x_2.

FIGURE 2

The extrapolation problem in PDPs is strongly linked to the aforementioned independence assumption. Independent features are a prerequisite for the computation of meaningful extrapolation results, therefore one could say that both problems go hand in hand. In the following chapters, the failure of PDPs in case of a violation of the independence assumption shall be discussed by means of real data examples (chapter 3.2) and based on simulated cases (chapter 3.4).

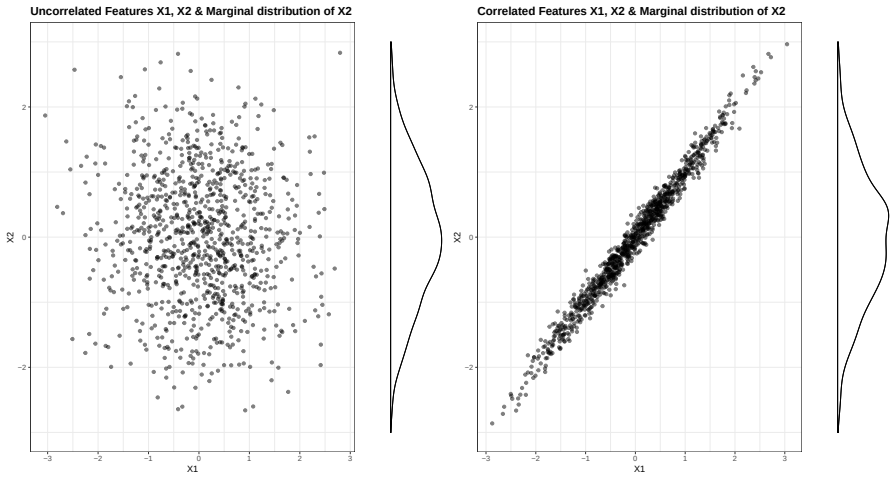

Uncorrelated Features X1, X2 & Marginal distribution of X2
Correlated Features X1, X2 & Marginal distribution of X2
X1
X2

FIGURE 3.1: a

4

PDP and Feature Interactions

Author: firstname lastname

5

PDP and Causal Interpretation

Author: firstname lastname

6

Introduction to Accumulated Local Effects (ALE)

6.1 Motivation

As seen in section 2 PDPs don't work well as soon as two or more features are correlated. This gives rise to the definition of ALEs. Although their definition makes sense for high dimensional feature spaces including categorical features, within this section we only treat a space with two continous features.

6.2 The Theoretical Formula

The uncentered ALE with respect to a starting point $z_{0,j}$ is defined by (Apley, 2016) as

$$\widetilde{ALE}_{\hat{f},\ j}(x) = \hat{f}_{x_j,ALE}(x) = \int_{z_{0,\ j}}^{x} E_{X_c|X_j}[\hat{f}^j(X_j,\ X_c) \mid X_j = z_j]\ dz_j,$$

where $\hat{f}$ is an arbitrary prediction function on the featurespace $X = (X_j, X_c)$ (with feature of interest X_j and other features X_c) as well as its j-th partial derivative $\hat{f}^j(*,*)$.

6.2.1 Centering

The ALE (centered ALE) is defined as

$$ALE_{\hat{f},\ j}(x) = \widetilde{ALE}_{\hat{f},\ j}(x) - E_{X_j}[\widetilde{ALE}_{\hat{f},\ j}(X_j)]$$

The centering makes sense as it helps interpreting the ALE in a reasonable way. This will be explained in section 3.4..

6.3 Estimation Formula

Since this theoretical formula is of no use for a blackbox model with unknown or even non existing gradients, an approximative approach will be used. The uncentered ALE can be approximated by the formula

$$\widehat{\widetilde{ALE}}_{\hat{f},\ j}(x) = \int_{z_{0,\ j}}^{x} \sum_{k=1}^{K} 1_{]z_{k-1,\ j},\ z_{k,\ j}]}(x_j)\, \frac{1}{n_j(k)} \sum_{i:x_j^{(i)} \in N_j(k)} \frac{[\hat{f}(z_{k,\ j}, x_j^{(i)}) - \hat{f}(z_{k-1,\ j}, x_j^{(i)})]}{z_{k,\ j} - z_{k-1,\ j}}\, dx_j\ .$$

In a first step the relevant dimension of the feature space is divided into K intervals beginning with the starting point $z_{0,j}$. As it is not clear how to exactly divide the feature space, section 3.x deals with that question. The upper boundary of the k-th interval is denoted by $z_{k,\ j}$ as well as the lower boundary by $z_{k-1,\ j}$. $N_j(k)$ denotes the k-th interval, i.e. $]z_{k-1,\ j}, z_{k,\ j}]$ and $n_j(k)$ the total number of observations having the j-value within this interval. $x_j^{(i)}$ is the j-value of the i-th observation and correspondingly $x_j^{(i)}$ the values of the other features. The term on the right approximates the expected partial derivative within each interval. Therefore each instance within an interval is shifted to the upper and lower limit of the interval and the total difference of the prediction is calculated. Devided by the length of the interval this is a reasonable approximation for the "local" effect on the prediction, if the feature of interest changes (cet. par.). By averaging these approximations over all observations within the k-th interval, we recieve a rough estimator for the term $E_{X_c|X_j}[\hat{f}^j(X_j,\ X_c) \mid X_j \in N_j(k)]$, which we take as constant effect for the k-th interval. By integrating over this step function, which represents the locally estimated derivatives, the (local) changes are accumulated. Thats why the name Accumulated Local Effects is quite reasonable. The approximative formula for the centered ALE follows directly as

$$\widehat{ALE}_{\hat{f},\ j}(x) = \widehat{\widetilde{ALE}}_{\hat{f},\ j}(x) - \frac{1}{n} \sum_{i=1}^{n} \widehat{\widetilde{ALE}}_{\hat{f},\ j}(x_j^{(i)})\ .$$

6.3.1 Implementation Formula

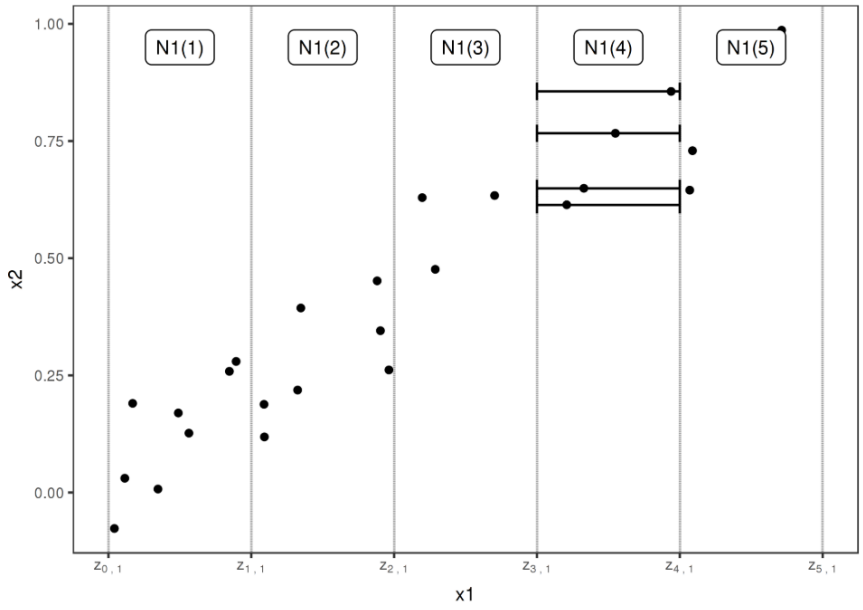
As both the centered and the uncentered ALE estimations are piecewise linear functions (integration over a step function), one can first calculate the ALE at the interval boundaries and interpolate in a second step. Therefore the folowing formula proposed by (Apley, 2016, page 11) , with slightly changed notation will be useful. The definitions of its components are as above. Additionally $k_j(x)$ is defined as the number of the interval that contains x, i.e. $x \in\ ]z_{k_j(x)-1,\ j},\ z_{k_j(x),\ j}]$.

$$\widehat{\widetilde{ALE}}_{steps,\ \hat{f},\ j}(x) = \sum_{k=1}^{k_j(x)} \frac{1}{n_j(k)} \sum_{i:\ x_j^{(i)} \in N_j(k)} [\hat{f}(z_{k,\ j}, x_j^{(i)}) - \hat{f}(z_{k-1,\ j},\ x_j^{(i)})].$$

This formula returns a step function. The values in each interval are the accumulated values of the averaged total differences in each interval. To transfer this formula into the correct estimator of the uncentered ALE one has to linearely interpolate the points $(z_{k-1,\ j},\ \widehat{\widetilde{ALE}}_{steps,\ \hat{f},\ j}(z_{k-1,\ j}))$ with $(z_{k,\ j}, \widehat{\widetilde{ALE}}_{steps,\ \hat{f},\ j}(z_{k,\ j}))$ for $k \in \{1, ..., K\}$ and $\widehat{\widetilde{ALE}}_{steps,\hat{f},j}(z_{0,\ j}) = 0$.
Since in this formula there is no integral, it is easier to implement.

6.4 Intuition and Interpretation

As the former sections introduced the theoretical basics for the ALE, this section shall provide an intuition as well for the calculation method as for the interpretation. As described above, the local behaviour of the model with respect to the variable of interest is estimated by moving the existing data points to the boundaries of their interval and evaluating the total difference of the prediction for the "new" data points. Figure **??** first offered by (Molnar, 2019) gives a good intuition for this procedure.

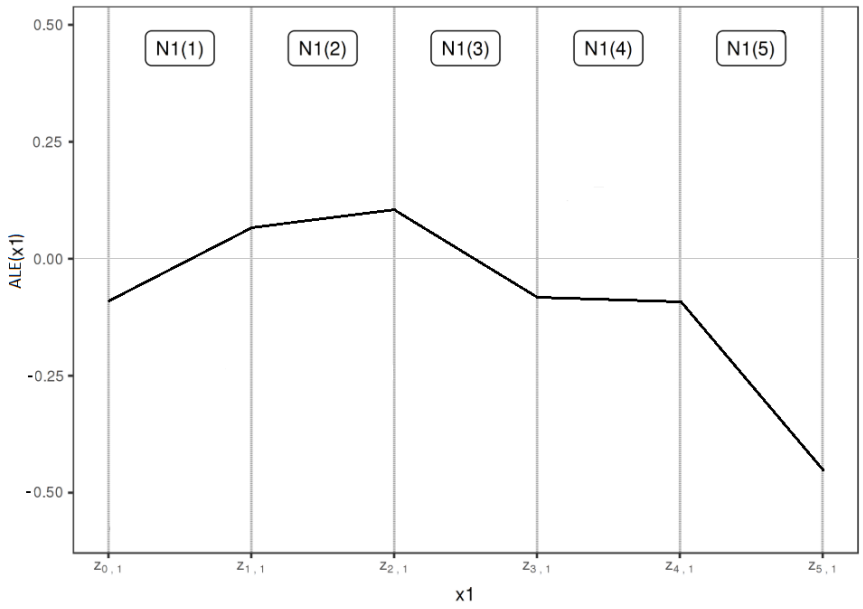


First one splits the total range of the variable of interest (in this case x_1) to intervals of suitable size.
For each interval the contained data points are moved to the interval boundaries. One gets twice as much "simulated" new data points as originally contained in each interval. The prediction function is now evaluated at this simulated points and the total difference of the prediction (for the given interval) is estimated as the mean change. Divided by the length of the interval one gets an estimation for the partial derivative within this interval. Theoretically one recieves the uncentered ALE by integration over this step function. Technically in a first step the total change per interval is accumulated. In a second step linear interpolation at the interval boundaries simulates a constant change within each interval. Both variants lead to the same result.

As the evaluation is ideally done on relatively small intervals, on the one hand the local behaviour of the model is estimated. On the other hand the covariance structure of the features is taken into account, as only "realistic" data points are simulated. This is in accordance with sampling from the conditional distribution.

In a last step the uncentered ALE is centered, i.e. shifted by a constant such that the expectation of the centered ALE is zero.

Figure **??** shows an example ALE which could match the data situtaion of Figure **??** :



To understand the interpretation of the ALE it can be usefull to first have a look on the intuition behind the uncentered ALE. If the value of the uncen-

tered ALE at $x_1 = 2$ equals 1, this means that if one samples a data point from the joint distribution of both features but only knows that $x_1 = 2$, one would expect the prediction to be 1 higher than the average prediction for realistic data points at $x_1 = z_{0,1}$ (i.e. data points sampled from the conditional distribution at $x_1 = z_{0,1}$). This expectation strongly depends on the reference point $z_{0,1}$, which per definition is smaller than the smallest x_1-value of the data. By substracting the expectation of the uncentered ALE - which is the mean difference of the prediction of a data point from the joint distibution to the prediction of a realistic data point(i.e. from the conditional distribution) at $x_1 = z_{0,1}$ - the interpretation becomes a lot easier. If the value of the (centered) ALE at $x_1 = 2$ equals for example 2, this means that, if one samples a data point from the joint distribution of both features and x_1 equals 2, one would expect the prediction to be 2 higher than the average prediction for an average data point of the joint distribution.

So far only the case of 2-dimensional feature spaces with one feature of interest was taken into account. In the following chapters methods and interpretation for ALE with two numeric features (second order effects) or one categorical feature will be in the focus. Furthermore we will have a look on the size of the intervals the data is evaluated on, which can be crucial for the expressiveness of the ALE.

7

Comparison of ALE and PDP

Author: firstname lastname

8

ALE Intervals, Piece-Wise Constant Models and Categorical Features

Author: firstname lastname

9

Introduction to Feature Importance

As in previous chapters already discussed, there exist a variety of methods that enable a better understanding of the relationship between features and the outcome variables, especially for complex machine learning models. For instance, Partial Dependence (PD) plots visualize the feature effects on a global, aggregated level, whereas Individual Conditional Expectation (ICE) plots unravel the average feature effect by analyzing individual observations, allowing to detect, if existing, any heterogeneous relationships. Yet, these methods do not provide any insights to what extent a feature contributes to the predictive power of a model - in the following defined as feature importance. This perspective becomes interesting when recalling that so far black-box machine learning models aim for predictive accuracy rather than for inference, and hence, it is persuasive to also establish agnostic-methods that focus on the performance dimension. In the following, the two most common approaches, Permutation Feature Importance (PFI) by Breiman (2001) and Leave-One-Covariate-Out (LOCO) by Lei et al. (2018), for calculating and visualizing a Feature Importance metric, are introduced. At this point, however, it is worth to clarify that the concepts of feature effects and feature importance can by no means be ranked but rather should be considered as mutual complements that enable the interpretability from different angles. After introducing the concepts of PFI and LOCO, a brief discussion of their interpretability but also its non-negligible limitations will follow.

9.1 Permutation Feature Importance (PFI)

The concept of Permutation Feature Importance was first introduced by Breiman (2001) and applied on a random forest model. The main principle is rather straightforward and easily implemented. The idea is as follows: When permuting the values of feature j, its explanatory power mitigates, as it breaks the association to the outcome variable y. Therefore, if the model relied on the feature j, the prediction error $e = L(y, f(X))$ of the model f should increase when predicting with the "permuted feature" dataset X_{perm} instead of with the "initial feature" dataset X. The importance of feature j is then evaluated by the increase of the prediction error which can be either determined by tak-

ing the difference $e_{perm} - e_{orig}$ or taking the ratio e_{perm}/e_{orig}. Note, taking the ratio can be favorable when comparing the result across different models. A feature is considered less important, if the increase in the prediction error was comparably small and the opposite if the increase was large. Thereby, it is important to note that when calculating the prediction error based on the permuted features there is no need to retrain the model f. This property constitutes computational advantages, especially in case of complex models and large feature spaces. Below, a respective PFI algorithm based on Fisher et al. (2018) is outlined. Note however, that their original algorithm has a slightly different specification and was adjusted here for general purposes.

The Permutation Feature Importance algorithm based on Fisher, Rudin, and Dominici (2018):

Input: Trained model f, feature matrix X, target vector y, error measure $L(y, f(X))$

1. Estimate the original model error $e_{orig} = L(y, f(X))$ (e.g. mean squared error)
2. For each feature $j = 1, ..., p$ do:
   - Generate feature matrix X_{perm} by permuting feature j in the data X
   - Estimate error $e_{perm} = L(y, f(X_{perm}))$ based on the predictions of the permuted data
   - Calculate permutation feature importance $PFI_j = e_{perm}/e_{orig}$. Alternatively, the difference can be used: $PFI_j = e_{perm} - e_{orig}$
3. Sort features by descending FI.

In Figure 9.1 it is illustrated, by a fictional example, how the permutation algorithm alters the original dataset. For each of the p features, the respectively permuted dataset is then used to first predict the outcomes and then calculate the prediction error.

To show, how the PFI for all features of a model can be visualized and thereby more conveniently compared, the PFI algorithm with a random forest model is applied on the dataset "Boston" (see Figure 9.2), which is available in R via the `MASS` package. To predict the house price, seven variables are included, whereby as the results show, the PFIs vary substantially across the variables. The depicted points correspond to the median PFIs over all shuffling iterations of one feature and the boundaries of the bands illustrate the 0.05- and 0.95-quantiles, respectively (see `iml` package).

Original data set

	x_1	x_2	...	x_p	y	$\hat{y}$
1	2	0.2		female	1	1
2	8	0.6		male	0	0
3	7	0.5		male	1	0
4	3	1.1		female	0	0
5	14	0.8		female	1	1
...						
n	11	0.4		male	1	1

Data set with permuted covariate x_1

	x_1	x_2	...	x_p	y	$\hat{y}$
1	14	0.2		female	1	0
2	11	0.6		male	0	1
3	2	0.5		male	1	1
4	7	1.1		female	0	1
5	3	0.8		female	1	0
...						
n	8	0.4		male	1	1

$$PFI_1 = L(y, f(X_{\text{perm},1})) - L(y, f(X))$$

Original data set

	x_1	x_2	...	x_p	y	$\hat{y}$
1	2	0.2		female	1	1
2	8	0.6		male	0	0
3	7	0.5		male	1	0
4	3	1.1		female	0	0
5	14	0.8		female	1	1
...						
n	11	0.4		male	1	1

Data set with permuted covariate x_p

	x_1	x_2	...	x_p	y	$\hat{y}$
1	2	0.2		male	1	1
2	8	0.6		female	0	0
3	7	0.5		male	1	0
4	3	1.1		male	0	1
5	14	0.8		female	1	1
...						
n	11	0.4		female	1	0

$$PFI_p = L(y, f(X_{\text{perm,p}})) - L(y, f(X))$$

FIGURE 9.1: Example for Permutation Feature Importance

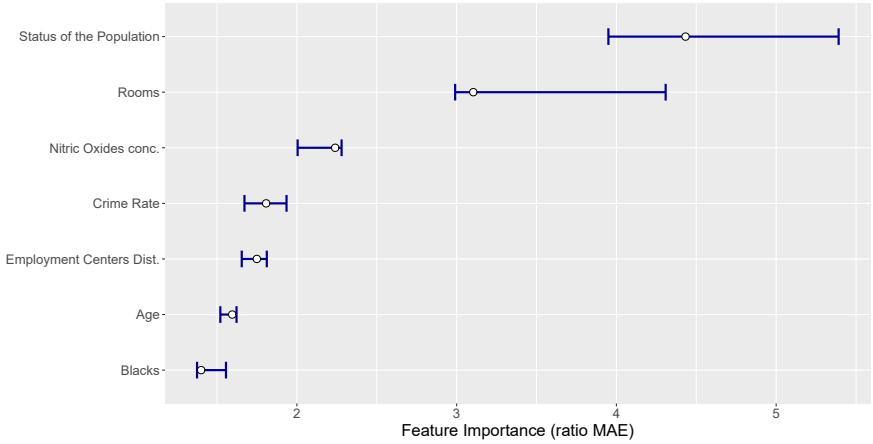


FIGURE 9.2: Visualization of Permutation Feature Importance with a random forest applied on Boston dataset

9.2 Leave-One-Covariate-Out (LOCO)

The concept of Leave-One-Covariate-Out (LOCO) is as already mentioned a different approach to PFI with the same objective, to gain insights on the importance of a specific feature for the prediction performance of a model. Although applications of LOCO exist, where comparable to PFI, the initial values of feature j are replaced by its mean, median or zero (see Hall et al., 2017), and hence, circumvent the disadvantage of re-training the model f, the common approach follows the idea to simply leave the respective feature out. The overall prediction error of the re-trained model f_{-j} is then compared to the prediction error resulted from the baseline model. However, re-training the model results in higher computational costs, becoming more severe with an increasing feature space. Besides, as one is actually interested in assessing the Feature Importance of a fixed model f, comparing the performance of a fixed model with the performance of a model f_{-j} fitted merely with a subset of the data, might raise plausible concerns (see Molnar, 2019). The pseudo-code shown below, illustrates the algorithm for the common case where the feature is left out (see Lei et al., 2018).

The Leave-One-Covariate-Out algorithm based on Lei et al. (2018):

Input: Trained model f, feature matrix X, target vector y, error measure $L(y, f(X))$

1. Estimate the original model error $e_{orig} = L(y, f(X))$ (e.g. mean squared error)
2. For each feature $j = 1, ..., p$ do:
   - Generate feature matrix X_{-j} by removing feature j in the data X
   - Refit model f_{-j} with data X_{-j}
   - Estimate error $e_{-j} = L(y, f_{-j}(X_{-j}))$ based on the predictions of the reduced data
   - Calculate LOCO Feature Importance $FI_j = e_{-j}/e_{orig}$. Alternatively, the difference can be used: $FI_j = e_{-j} - e_{orig}$
3. Sort features by descending FI.

In Figure 9.3 it is shown, how the LOCO algorithm alters the original dataset, whereby it always differs, depending on the respective feature that is left out. Note, that the qualitative and quantitative interpretations correspond the ones from the PFI method. So do the visualization tools and therefore at this point it is refrained from providing the reader with an additional real data example.

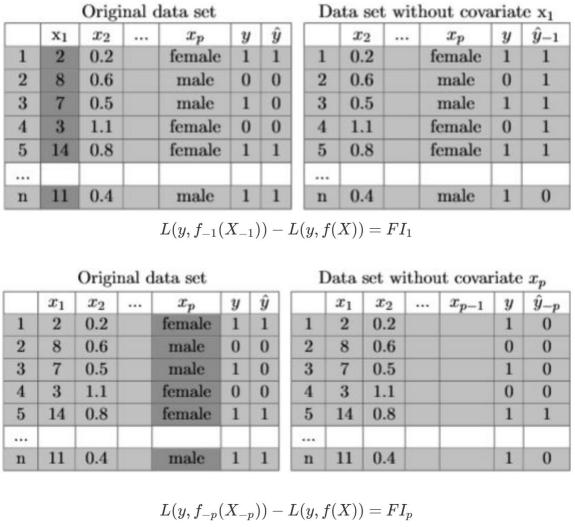


FIGURE 9.3: Example for Leave-One-Covariate-Out Feature Importance

9.3 Interpretability of Feature Importance and its Limitations

After both methods are discussed, the question to what extent these agnostic-methods can contribute to a more comprehensive interpretability of machine learning models will be now touched on. By doing so, it is necessary to also reflect upon the limitations of the model regarding the interpretability of the results. The latter will constitute the main focus in the following chapters. Conveniently, both methods are highly adaptable on whether using classification or regression models, as they are non-rigid towards the prediction error metric (e.g. Accuracy, Precision, Recall, AUC, Average Log Loss, Mean Absolute Error, Mean Squared Error etc.). This allows to assess Feature Importance based on different performance measures. Besides, the interpretation can be conducted on a high-level, as both concepts do consider neither the shape of the relationship between the feature and outcome variable nor the direction of the feature effect. However, as illustrated in Figure 9.2, PFI and LOCO only return for each feature a single number and thereby neglect possible variations between subgroups in the data. Chapter XX will focus on how this limitation can be, at least for PFI, circumvented and introduces the concepts of Partial Importance (PI) and Individual Conditional Importance (ICI) which both avail themselves on the conceptual ideas of PD and ICE (see Casalicchio et al., 2018). Besides, two general limitations appear when some features in the feature space are correlated. First, correlation makes an

isolated analysis of the explanatory power of a feature complicated which results in an erroneous ranking in Feature Importance and hence, in incorrect conclusions. Second, if correlation exists and only in case of applying the PFI method, permuting a feature can result in unrealistic data instances so that the model performance is evaluated based on data which is never observed in reality. This makes comparisons of prediction errors complicated and therefore it should always be checked for this problem, if applying the PFI method. Chapter XX will focus on this limitations by comparing the performance of PFI and LOCO for different models and different levels of correlation in the data. Beyond these limitations, it is evident to also question whether these agnostic-methods should be computed on training or test data. As answering that depends highly on the research question and data, it is refrained from going into more detail at this point but will be examined and further discussed in chapter XX.

10

PFI, LOCO and Correlated Features

Author: firstname lastname

11

Partial and Individual Permutation Feature Importance

Author: firstname lastname

12

PFI: Training vs. Test Data

Author: firstname lastname

13

Introduction to Local Interpretable Model-Agnostic Explanations (LIME)

When doing machine learning we always build models. Models are simplifications of reality. Even if the predictive power of a model may be very strong, it will still only be a model. However, models with high predictive capacity do most of the time not seem simple to a human as seen throughout this book. In order to simplify a complex model we could use another model. These simplifying models are referred to as surrogate models. They imitate the black box prediction behaviour of a machine learning model subject to a specific and important constraint: surrogate models are interpretable. For example, we may use a neural network to solve a classification task. While a neural network is anything but interpretable, we may find that some of the decision boundaries are explained reasonably well by a logistic regression which in fact yields interpretable coefficients.

In general, there are two kinds of surrogate models: global and local surrogate models. In this chapter, we will focus on the latter ones.

The concept of local surrogate models is heavily tied to Ribeiro et al. (2016), who propose local interpretable model-agnostic explanations (LIME). Different from global surrogate models, local ones aim to rather explain single predictions by interpretable models than the whole black box model at once. These surrogate models, also referred to as explainers, need to be easily interpretable (like linear regression or decision trees) and thus may of course not have the adaptability and flexibility of the original black box model which they aim to explain. However, we actually don't care about a **global** fit in this case. We only want to have a very **local** fit of the surrogate model in the proximity of the instance whose prediction is explained.

A LIME explanation could be retrieved by the following algorithm:

1. Get instance x out of the data space for which we desire an explanation for its predicted target value.
2. *Perturb* your dataset X and receive a perturbed data set Z of increased size.
3. Retrieve predictions $\hat{y}_Z$ for Z using the black box model f.

4. Weight Z w.r.t. the proximity/neighbourhood to x.
5. Train an explainable weighted model g on Z and the associated predictions $\widehat{y}_Z$.

Return: An explanation for the interpretable model g.

The visualisation below nicely depicts the described algorithm for a two-dimensional classification problem based on simulated data. We start only with our data split into two classes: 1 and 0. Then, we fit a model that can perfectly distinguish between the two classes. This is indicated by the sinus-shaped function drawn as a black curve. We do not perturb the data in this case. (However, we may argue that our perturbation strategy is to use the original data. We will more formally discuss perturbation later on.) Now, we choose the data point, which we want an explanation for. It is coloured in yellow. With respect to this point, we weight our data by giving close observations higher weights. We illustrate this by the size of data points. Afterwards, we fit a classification model based on these weighted instances. This yields an interpretable linear decision boundary - depicted by the purple line. As we can see, this is indeed locally very similar to the black box decision boundary and seems to be a reasonable result.

This way we receive a single explanation. This one explanation can only help to understand and validate the corresponding prediction. However, the model as a whole can be examined and validated by multiple (representative) LIME explanations.

So far so good. However, the previous outline was not very specific and leaves (at least) three questions. First, what does neighbourhood refer to? Second, what properties should suitable explainers have? Third, what data do we use, why and how do we perturb this data?

To better assess these open questions it may be helpful to study the mathematical definition of $LIME$. The explanation for a datapoint x, which we aim to interpret, can be represented by the following formula:

$$explanation\,(x) = arg\,min_{g \in G}\,\mathcal{L}\,(f, g, \pi_x) + \Omega\,(g)$$

Let's decompose this compact, yet precise definition:

x can be an instance that is entirely new to us as long as it can be represented in the same way as the training data of the black box model. The final explanation for x results from the maximisation of the loss-like fidelity term $\mathcal{L}\,(f, g, \pi_x)$ and a complexity term $\Omega\,(g)$. f refers to the black box model we want to explain and g to the explainer. G represents the complete hypothesis space of a given interpretable learner. The explanation has to deal with two trade-off terms when minimising: The first term $\mathcal{L}\,(f, g, \pi_x)$ is responsible to

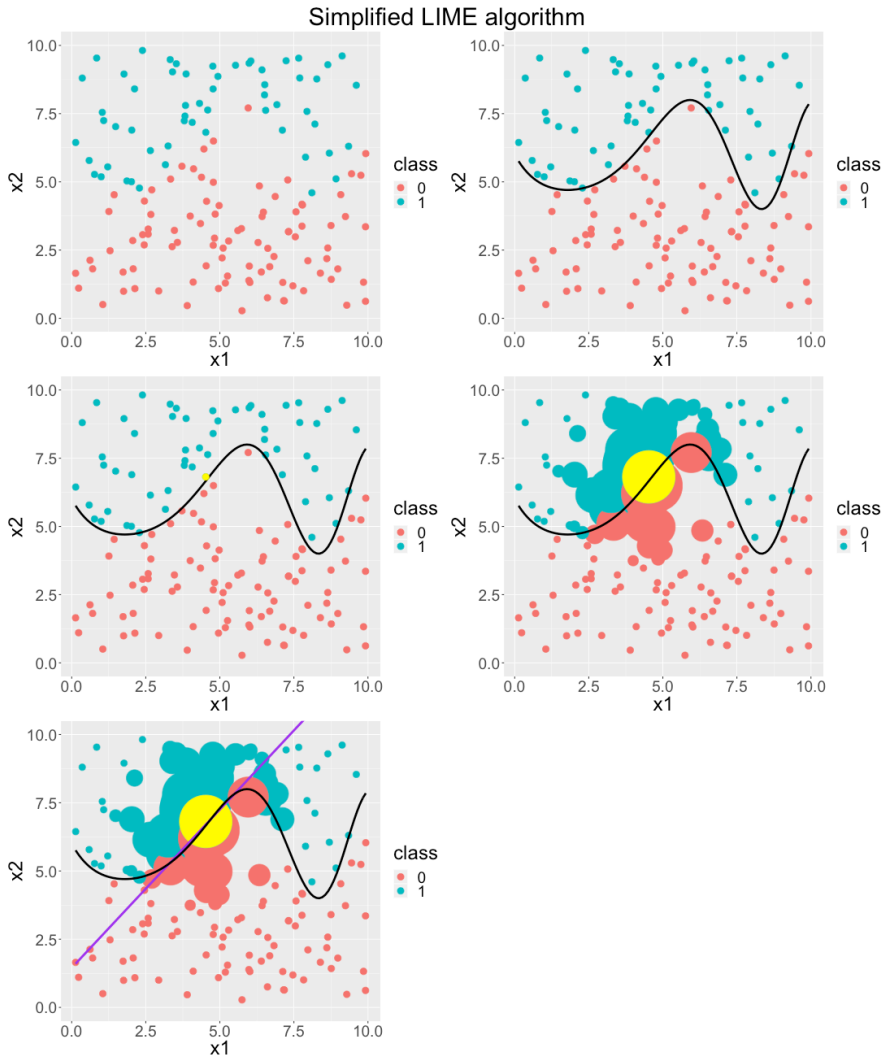


FIGURE 13.1: Simplified graphical representaion of the LIME algorithm. Each single panel represents one step of the described algorithm. It reads from left to right.

deliver the optimal fit of g to the model f while a low *loss* is desirable indicating high (local) *fidelity*. The optimal fit is only found with respect to a proximity measure $\pi_x(z)$ in the neighbourhood of x.

This leads us to the first open question: What does neighbourhood refer to? Neighbourhood is a very vague term. This is for good reason because a priori it is not clear how to specify a neighbourhood properly. Technically, there are many different options to deal with this issue. Weighting the observations w.r.t. their distance to the observation being explained seems like a good idea. This may possibly be implemented as an arbitrarily parametrised kernel. However, this leaves in total many scientific degrees of freedom which makes the neighbourhood definition somewhat problematic. This neighbourhood issue will be discussed in more detail in the next chapter.

We already answered the second open question - what properties suitable explainers should have - in parts. We mentioned the interpretability property and outlined generalised linear models or decision trees as examples. However, we did not discuss further desired properties of these models. Since they have strong assumptions, it is unlikely that they are capable of maintaining an optimal fit to the original black box model. Recall our formula. As we want local optimal fit subjected to a certain (low) degree of explainer complexity - in order to allow interpretation - our formula needs to facilitate this aspect. $\Omega(g)$ is our complexity measure and responsible to choose the model with the lowest complexity. For example, for decision trees, tree depth may describe the complexity. In the case of linear regression, the L_1 norm may indicate how simple the interpretation has to be. The resulting LASSO model allows us to focus only on the most important features.

Having answered the first two open question we still have the last question related to the data and the perturbation unresolved. Besides the tabular data case, we can also interpret models trained on more complex data, like text data or image data. However, some data representations (e.g. word embeddings) are not human-interpretable and must be replaced by interpretable variants (e.g. one-hot-encoded word vectors) for LIME to yield interpretable results. The function modeled by the black box model operates in the complete feature space. It can even yield predictions for instances not seen in the training data. This means that the original data does not sufficiently explore the feature space. Hence, we want to create a more complete *grid* of the data and fill the feature space with new observations so that we can better study the behaviour of the black box model. Still, the data for the explainer should be related to the original data. Otherwise the explainer may be ill-placed in space having nothing in common with the original problem anymore. This is why we perturb the original data. But how does perturbation work? This is a priori not clear at all. For categorical features, perturbation may be realised by randomly changing the categories of a random amount of features, or even recombining all possible levels of these. Numerical features may be drawn from a properly

parametrised (normal) distribution. The perturbed data set, which is used to train the explainer, should be much larger than the original one and supposed to better represent the (possible) feature space, giving the surrogate model more anchor points - especially in sparse areas. Further details on this topic will be studied in chapter X.

As we did not provide any examples on the application of LIME, please refer to the next two chapters or Molnar (2019) for some examples.

The definition of LIME still seems after all very rough and vague. This leaves us many scientific degrees of freedom when implementing it - for the good and for the bad. For example, we see that the model f can be any machine learning model that exists. This gives us the opportunity to drastically change the underlying predictive model while keeping the same explainer g with the same complexity constraints.

On the other hand, LIME being a very generic approach also means that many "hyperparameters", like the neighbourhood definition or the sampling/perturbation strategy, are arbitrary. Hence, it is likely that in some use cases LIME explanations heavily depend on changing the hyperparameters. In these cases, the explanations can hardly be trusted and should be treated with great care.

The following two chapters will focus on two very significant hyperparameters: the neighbourhood definition and the sampling strategy. They will investigate how these affect the results of the method and their interpretability. We will emphasise the coefficient stability of LIME explainers in order to illustrate the trustworthiness of the results.

14

LIME and Neighbourhood

Author: Firstname Lastname

15

LIME and Sampling

Author: Firstname Lastname

Bibliography

Apley, D. W. (2016). *Visualizing the Effects of Predictor Variables in Black Box Supervised Learning Models*.

Breiman, L. (2001). Random forests. *Machine learning*, 45(1):5–32.

Casalicchio, G., Molnar, C., and Bischl, B. (2018). Visualizing the feature importance for black box models. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 655–670. Springer.

Fisher, A., Rudin, C., and Dominici, F. (2018). Model class reliance: Variable importance measures for any machine learning model class, from the" rashomon" perspective. *arXiv preprint arXiv:1801.01489*.

Goldstein, A., Kapelner, A., Bleich, J., and Pitkin, E. (2013). Peeking inside the black box: Visualizing statistical learning with plots of individual conditional expectation. *Journal of Computational and Graphical Statistics*, 24.

Hall, P., Phan, W., and Ambati, S. S. (2017). Ideas on interpreting machine learning.

Hastie, T., Tibshirani, R., and Friedman, J. (2013). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer Series in Statistics. Springer New York.

Lei, J., G'Sell, M., Rinaldo, A., Tibshirani, R. J., and Wasserman, L. (2018). Distribution-free predictive inference for regression. *Journal of the American Statistical Association*, 113(523):1094–1111.

Molnar, C. (2019). *Interpretable Machine Learning*. https://christophm.github.io/interpretable-ml-book/.

Ribeiro, M. T., Singh, S., and Guestrin, C. (2016). 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. ACM.