Interpretability/Explanability Methods of Machine Learning Models

Jennifer Jara

Septembre 2020

About data

We have a simulated data set from lung cancer patients treated and then followed up post-treatment. The variable we are trying to explain is the patient's response to treatment, based on a set of clinical and radiomic variables.

Exploring data

Data import

The database is composed by 300 individuals and 14 variables, including 11 radiomic features, age, weight and the variable to be explained *"Response to treatment"*. The dataset used in this tutorial was created from the generation of observations from a multivariate normal distribution, the variable to be explained was created from the estimation of a logistic model on the baseline data.

```
load(file = "/Users/jenniferjara/Documents/Code R/Covid19/Tutoriel/data_tutorial.RData")
colnames(data)[which(names(data) == "Y_sim")] <- "Treatment.response"
```

Summary of radiomic features

We present a summary of the statistics of centrality and variability of the set of variables.

The distribution of the variable of interest *Response to treatment* corresponds to 40% of *Favorable responses* and 60% of *Non-favorable responses* to treatment. The table below shows the difference-of-means tests for patients in each group.

```
tab1.Treat <-
  tableby(Treatment.response ~ ., data = data, numeric.test = "kwt", cat.test = "fe")

tab2.Treat <-
  cbind(as.data.frame(summary(tab1.Treat)), "Adjusted p-value" = as.data.frame(
    summary(padjust(tab1.Treat, method = "BH")))$`p value`)

kableExtra::kable(
  tab2.Treat,
  col.names = c("", "Unfavorable (N=175)", "Favorable (N=125)", "Total (N=300)",
                "p-valeur", "p-valeur Ajustée"),
```

```
format = "markdown",
caption = "Description of the study population")
```

Table 1: Description of the study population

	Unfavorable (N=175)	Favorable (N=125)	Total (N=300)	p-valeur	p-valeur Ajustée
Age				< 0.001	< 0.001
Mean (SD)	63.760 (10.985)	69.000 (10.645)	65.943 (11.132)		
Range	39.000 - 88.000	46.000 - 89.000	39.000 - 89.000		
Weight				0.003	0.007
Mean (SD)	66.263 (14.303)	61.152 (13.592)	64.133 (14.214)		
Range	28.000 - 114.000	27.000 - 91.000	27.000 - 114.000		
Recist.cm				0.028	0.045
Mean (SD)	6.591 (3.081)	5.796 (2.614)	6.259 (2.918)		
Range	0.360 - 16.450	0.050 - 12.760	0.050 - 16.450		
Surface.cm2				< 0.001	< 0.001
Mean (SD)	170.718 (208.429)	70.515 (67.261)	128.967 (172.064)		
Range	6.970 - 1725.270	5.930 - 362.060	5.930 - 1725.270		
Volume.cm3				< 0.001	< 0.001
Mean (SD)	123.573 (233.917)	34.338 (58.818)	86.392 (187.666)		
Range	0.890 - 1959.070	0.410 - 374.480	0.410 - 1959.070		
Number.of.grey.levels				0.017	0.032
Mean (SD)	1017.954 (280.074)	931.912 (270.354)	982.103 (278.865)		
Range	308.000 - 1737.000	149.000 - 1457.000	149.000 - 1737.000		
Min.value.stat				0.217	0.236
Mean (SD)	-880.754 (78.354)	-892.488 (75.437)	-885.643 (77.242)		
Range	-997.000 - -642.000	-999.000 - -636.000	-999.000 - -636.000		
Max.value.stat				0.218	0.236
Mean (SD)	573.726 (362.271)	547.448 (376.650)	562.777 (367.935)		
Range	76.000 - 2449.000	131.000 - 2366.000	76.000 - 2449.000		
Median.value.stat				0.032	0.046
Mean (SD)	5.309 (97.964)	-18.144 (92.256)	-4.463 (96.170)		
Range	-241.000 - 281.000	-234.000 - 242.000	-241.000 - 281.000		
Skewness.stat				0.557	0.557
Mean (SD)	-1.883 (1.094)	-1.834 (1.257)	-1.863 (1.163)		
Range	-4.180 - 1.450	-5.340 - 1.410	-5.340 - 1.450		
Standard.deviation.stat				< 0.001	< 0.001
Mean (SD)	150.523 (49.065)	198.715 (76.874)	170.603 (66.469)		

	Unfavorable (N=175)	Favorable (N=125)	Total (N=300)	p-valeur	p-valeur Ajustée
Range	68.800 - 300.830	84.120 - 454.420	68.800 - 454.420		
Contrast.GLCM				0.116	0.150
Mean (SD)	65.172 (32.772)	72.124 (39.032)	68.069 (35.618)		
Range	3.640 - 150.250	3.020 - 174.070	3.020 - 174.070		
Homogeneity.GLCM				0.017	0.032
Mean (SD)	0.277 (0.095)	0.252 (0.086)	0.266 (0.092)		
Range	0.085 - 0.604	0.098 - 0.493	0.085 - 0.604		

Modeling data

Separation of data training and data test

We will separate the data set in a learning and test bases. For this application, 2/3 of the observations in the data set will define the training data set and the remaining 1/3 will be considered the test data set.

```
set.seed(12345)
validation_index <- sample(seq(1, nrow(data), by=1), size = (nrow(data)*(2/3)), replace = F)
apprentissage <- data[validation_index, ]
test <- data[-validation_index, ]
```

Training model based on train dataset

We want to explain *response to treatment*, depending on radiomic features, age and weight of the patients. For this, a *Random Forest* model is trained from the learning base. The optimization of the model is made from a cross-validation (10-folds) considering the *AUC* as optimization metric.

```
set.seed(1234)
control <- trainControl(method="repeatedcv", number=10, repeats=5,
                        savePredictions = "final",
                        summaryFunction =  twoClassSummary, classProbs = TRUE)

tunegrid_tunning <- expand.grid(.mtry = (1:10))
rf_tunning <-  caret::train(Treatment.response ~ ., data = data,
                            method = "rf", ntrees = 1000, trControl = control,
                            tuneGrid = tunegrid_tunning, metric = "ROC")
```

The number of variables randomly selected in each split in decision trees is optimized in the cross validation, it is fixed at 5 for the model we will train in the next step.

Model Evaluation based on test dataset

We carry out the training of the model on the train data. The performance metrics of the trained model are calculated on test dataset. All metrics observed below are calculated from the confusion matrix, except the *AUC*, which is calculated based on the Sensitivity and Specificity for each possible threshold between 0 and 1.

```r
set.seed(1234)
rf_fitted <-  caret::train(Treatment.response ~ ., data = apprentissage,
                          method = "rf", ntrees = 1000,
                          trControl = control, tuneGrid = expand.grid(.mtry = 5),
                          metric = "ROC")
```

```r
(matrice_conf <- confusionMatrix(predict(object = rf_fitted, newdata = test),
                                 reference = test$Treatment.response, positive = "Yes"))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction No Yes
##        No  42  10
##        Yes 19  29
##
##                Accuracy : 0.71
##                  95% CI : (0.6107, 0.7964)
##     No Information Rate : 0.61
##     P-Value [Acc > NIR] : 0.02415
##
##                   Kappa : 0.4149
##
##  Mcnemar's Test P-Value : 0.13739
##
##             Sensitivity : 0.7436
##             Specificity : 0.6885
##          Pos Pred Value : 0.6042
##          Neg Pred Value : 0.8077
##              Prevalence : 0.3900
##          Detection Rate : 0.2900
##    Detection Prevalence : 0.4800
##       Balanced Accuracy : 0.7161
##
##        'Positive' Class : Yes
##
```

Global Interpretability

These methods will allow to identify the most important variables and their impact in the prediction.

Feature Importance (FI)

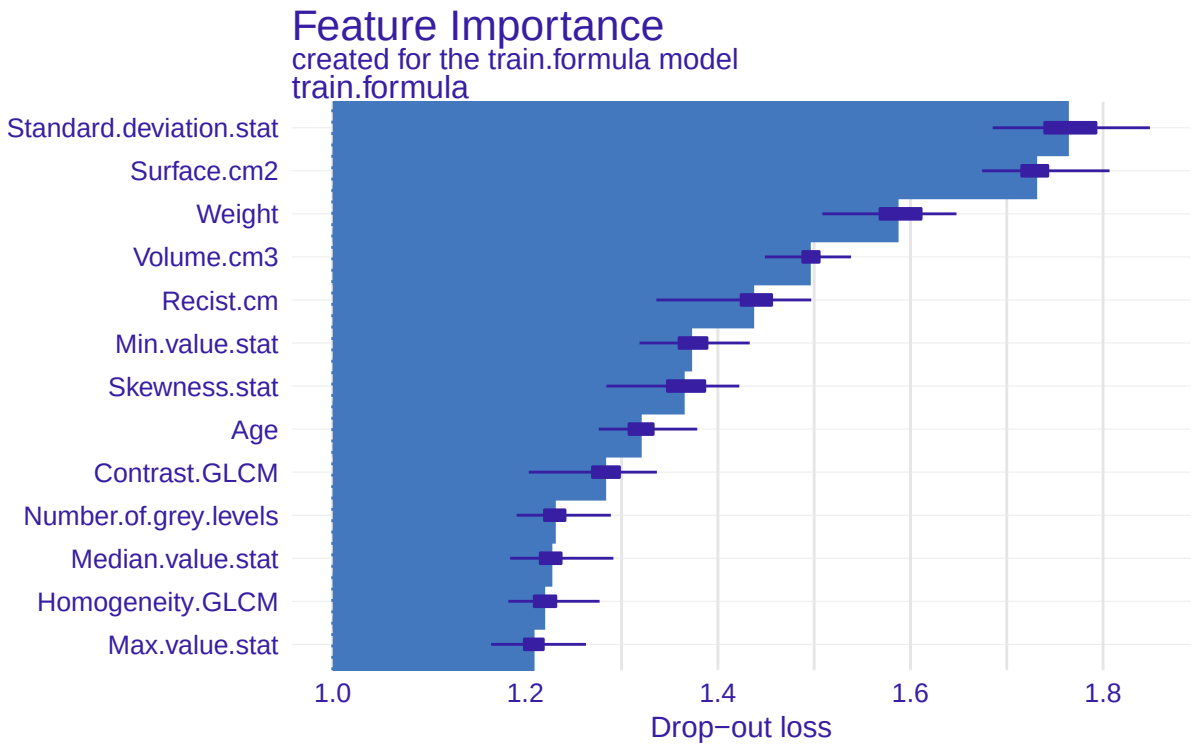
Warning. If the objective is to know the extent to which the model relies on each variable to make predictions, we must apply the *FI* to the *training data*. On the other hand, if we want to know the extent to which a variable contributes to the performance of the model on unknown data, we must apply the *FI* to the *test data*.

We want to determine the variables which contribute most to the prediction. We perform the analysis on the *learning data*. The *explain* object will allow us to obtain the predicted probabilities for patients in the study population from an adjusted model.

```
explain_model <- DALEX::explain(model = rf_fitted,
                                data = apprentissage[,-which(colnames(apprentissage)
                                                             %in% c("Treatment.response"))],
                                y = apprentissage$Treatment.response == "Yes")
```

Then, we use the *variable_importance* function of the *DALEX* package to measure the increase in model prediction error as we swap the values of each feature.

```
set.seed(1234)
var_imp_model <- DALEX::variable_importance(explain_model,
                                            loss_function = DALEX::loss_root_mean_square,
                                            B = 100, type = "ratio")
plot(var_imp_model)
```



We can see that the two most important variables in forecast estimation are : *standard.deviation.stat* and *surface.cm2*. We see, in particular, the RMSE obtained by removing the effect of these variables doubles the RMSE of the initial model. This is why they are considered as the features having the greatest influence on the prediction.

Advantages

* Provides a global overview of the behavior of the model for easy interpretation.
* The method takes into account the main effect of each variable as well as interactions with other variables.
* The *FI* does not require retraining the model, which saves us time at the computational level.

Disadvantages

* It is difficult to determine whether we should use training data or test data.
* This method is related to the model error, so we must apply it on the best performing models.
* It is essential to have access to the true value of the interest variable y.
* Repeating the permutation and averaging the importance measurements obtained stabilizes the metrics but increases the calculation time.
* If the variables are correlated the metric may be biased by unrealistic data (obtained in the permutation).

Predictions - radiomic features relationship

To evaluate the effect of the variables used in the model on the predictions made by the model. We can apply one of these techniques:

Partial Dependence plot (PDP) shows the marginal effect of a variable on the predicted probability obtained by a machine learning model.

Individual Conditional Expectation plot (ICE) is the equivalent of the PDP for each individual. An ICE plot shows the dependence of the prediction on an explanatory variable for each individual independently.

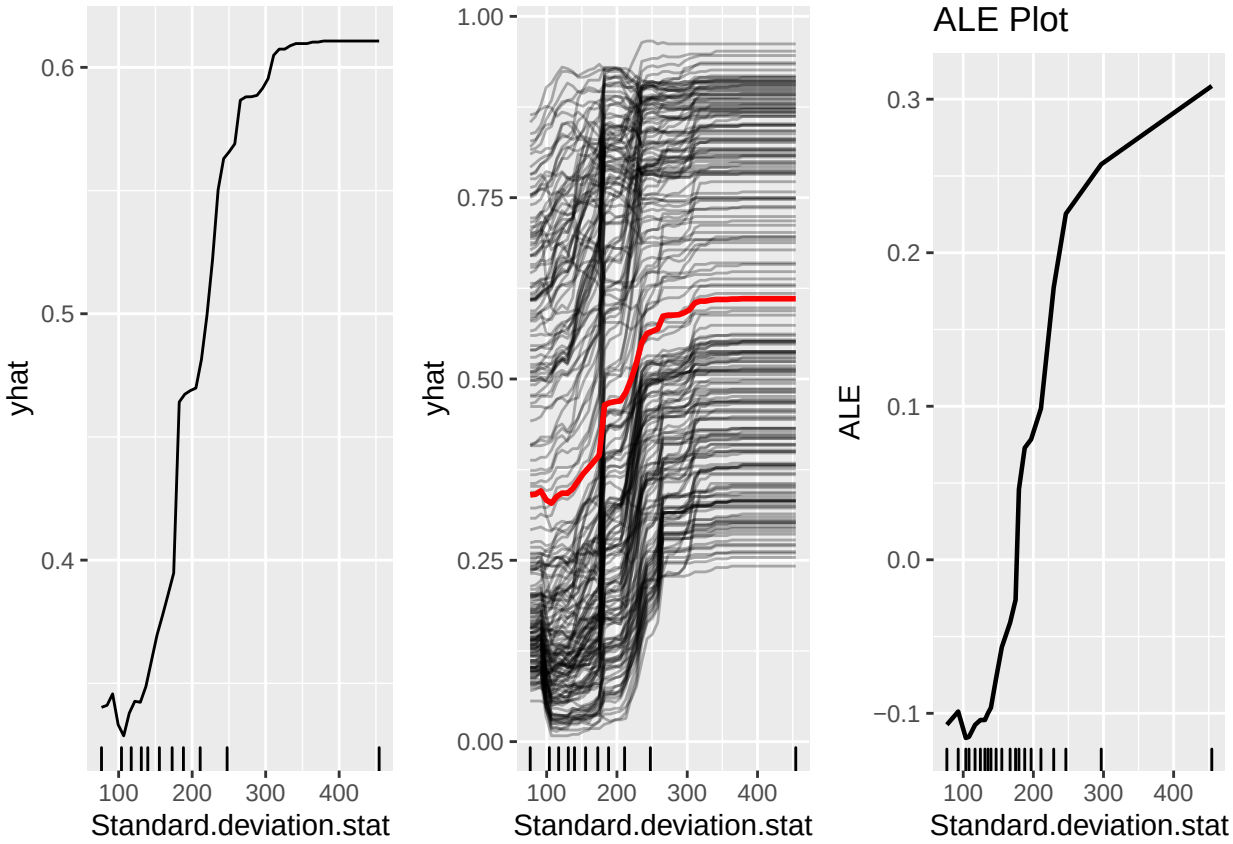
Accumulated Local Effects plot (ALE) describes the conditional effect of a variable on the mean prediction of an ML model. Preferred method when variables are correlated.

```
pred.prob.pdp <- function(object, newdata) {
  pred <- predict(object, newdata, type = "prob")
  return(mean(pred[,2])) }

pred.prob.ice <- function(object, newdata) {
  pred <- predict(object, newdata, type = "prob")
  return(pred[,2]) }

pred.prob.ale <- iml::Predictor$new(rf_fitted, data = apprentissage,
                                    y = "Treatment.response", class = "Yes", type = "prob")
```

```
ale_rf_var1 <- iml::FeatureEffect$new(pred.prob.ale, feature = "Standard.deviation.stat")
ggarrange(
    pdp::partial(rf_fitted, pred.var = "Standard.deviation.stat", pred.fun =pred.prob.pdp,
                 plot = T, rug = TRUE, plot.engine = "ggplot", title = "PDP"),
    pdp::partial(rf_fitted, pred.var = "Standard.deviation.stat", pred.fun =pred.prob.ice,
                 plot = T, rug = T, alpha = 0.3, plot.engine = "ggplot", title="ICE Plot"),
    ggplot(ale_rf_var1$results, aes(x = Standard.deviation.stat, y = .value)) +
      geom_line(size = 0.8) +  geom_rug(sides="b") + labs(x = "Standard.deviation.stat",
                                                          y = "ALE", title = "ALE Plot"),
    nrow = 1, ncol = 3)
```



We notice that the high values of the variable *Standard.deviation.stat*, have a positive influence on the predicted probability. The higher the value of the deviation, the higher the probability of responding positively to treatment.

Advantages
* *PDP* and *ICE*: are two effective metrics when the variables are not correlated.
* *ICE*: Analyzing the set of curves is more intuitive than the average curve of *PDP*.
* *ICE*: the curves can reveal heterogeneous relationships (especially with the d-ICE plot).
* *ALE*: is unbiased in presence of correlated variables. It is faster in computation time than *PDP*.
* *ALE*: the graphs are centered at zero, so each point of the curve corresponds to the difference from the mean prediction.

Disadvantages
* *PDP*: graphic representation of maximum two features.
* *PDP* and *ICE*: the independence hypothesis. In presence of correlated features, we create new improbable or invalid observations.
* *PDP* and *ICE* : heterogeneous effects are not taken into account since *PDP* only shows average marginal effects.
* *ICE*: graphic representation of a single feature. The presence of many lines may overload the graph and show nothing.
* *ALE*: Graphics can be sensitive with a high number of intervals.
* *ALE*: even if the graphs are unbiased, interpretation becomes difficult when the features are highly correlated.

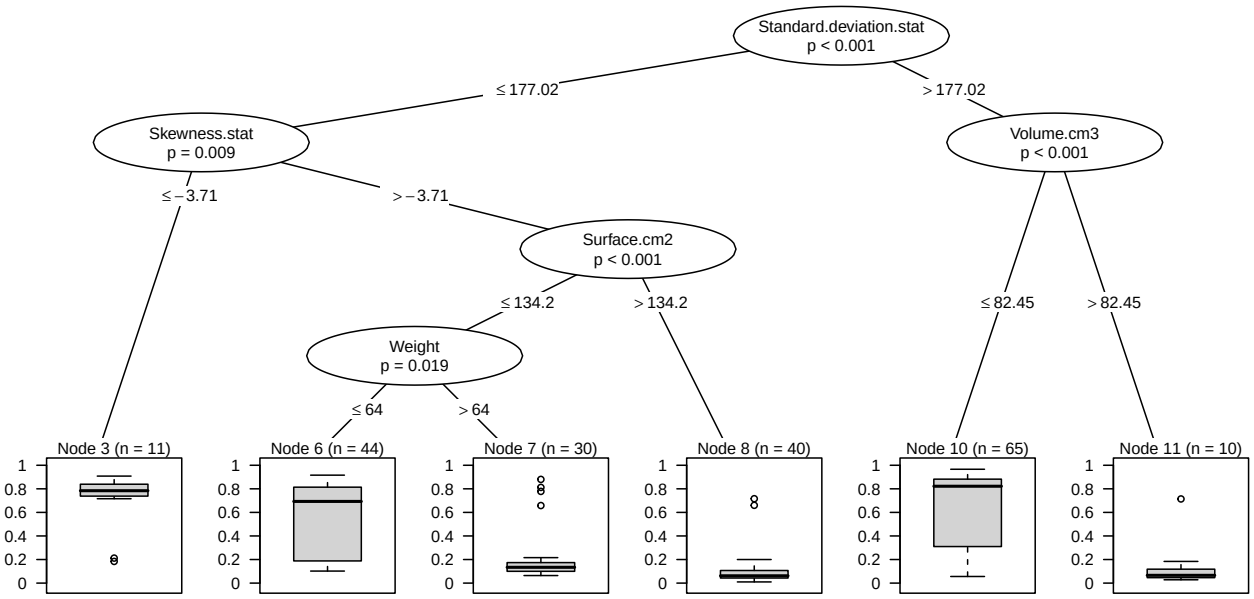
Surrogate model

The objective of surrogate models is to estimate the predictions of the underlying model (ML) as precisely as possible in order to facilitate its interpretation. The surrogate model frequently chosen is the decision

tree mainly for its simplicity of interpretation.

```
tree_surrogate <-
    partykit::ctree(explain_model$y_hat ~ .,
      data = apprentissage[,-which(colnames(apprentissage) %in% c("Treatment.response"))],
      control = partykit::ctree_control(maxdepth = 4))

plot(tree_surrogate, gp = gpar(fontsize = 8), ip_args = list(abbreviate = F, id = F))
```



The surrogate model explains 0.4537938 of the variance of the predictions made by the ML model, which suggests a bad approximation. The visualization presented shows the distribution of the probabilities (from initial model) in the different decision nodes built by the tree.

Advantages

* Substitution models are flexible and can be applied regardless of the black box model.
* It's intuitive and easy to implement, it's also simple to explain to professionals from other disciplines.
* The R^2 allows us to measure if the surrogate model is a good approximation of the machine learning model.
* Decision trees avoid the linearity assumptions of parametric regression models.

Disadvantages

* It is important to remember that we obtain conclusions on the Machine Learning model ($\hat{y}$) and not on the real y. The surrogate model never takes into account the real value of the variable of interest.
* There is no decision threshold for R^2 that allows to substitute an Machine Learning model by a surrogate model.
* A surrogate model can be widely divergent for different data subsets.

Local Interpretability

These methods will allow to explain the individual predictions made by a machine learning model, for a patient in the lung cancer dataset.

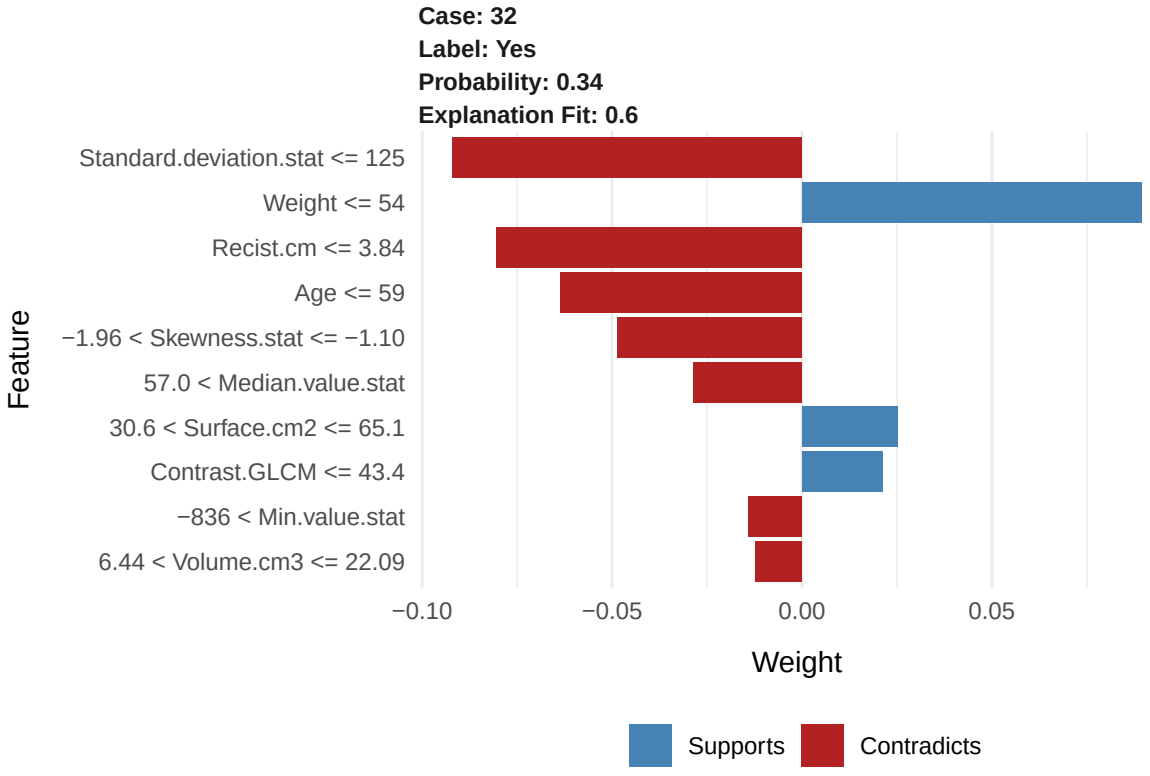
Local Surrogate Model (LIME)

Local substitution models aim to explain the individual prediction of a black box model, through a simpler model, in this case we use a *RIDGE model*. This allows us to identify the predominant characteristics that influenced an individual's prediction.

```
explainer_local <- lime::lime(x = apprentissage[,-which(colnames(apprentissage)
                                                        %in% c("Treatment.response"))],
                              model = rf_fitted, n_bins = 4)

set.seed(1234)
lime_explain_local <-
  lime::explain(x = test[3, -which(colnames(test)=="Treatment.response")],
                explainer = explainer_local, # explainer on discretised data
                n_permutations = 1000,       # Permutations to create
                dist_fun = "euclidean",      # Distance function
                kernel_width = 0.75,         # Define the neighborhood (size local region)
                n_features = 10,             # Features to best describe predicted outcomes
                feature_select = "highest_weights", # Algorithm to select features
                labels = "Yes")

plot_features(lime_explain_local)
```




```
pred_exp <- cbind(Case = unique(lime_explain_local$case),
                  prob_ML = unique(lime_explain_local$label_prob),
                  prob_explainer = unique(lime_explain_local$model_prediction),
                  R2_explainer = unique(lime_explain_local$model_r2))
```

Table 2: Estimated probabilities ML/LIME

Case	prob_ML	prob_explainer	R2_explainer
32	0.338	0.339981938000484	0.595497093628737

```
kable(pred_exp,
  caption = "Estimated probabilities ML/LIME") %>%
  kable_styling(c("striped", "bordered"))
```

The probability predicted (for the individual of interest) by the model *explainer* 0.3399819 is obtained from the sum of the intercept of the "simple model", (0.543551) and the coefficients obtained by the RIDGE model (-0.2035691), so we have $Prob_{explainer} = \beta_0 + \sum \beta_j I_j = 0.3399819$.

Advantages
* We can measure how close the interpretable model is to the black box predictions, using the R^2.
* The use of a Lasso model or a decision tree, provides short and understandable explanations of an individual's decision.
* The fidelity measure gives a good idea of the accuracy of the interpretable model to explain the predictions of the ML model in the neighborhood of the observation of interest.

Disadvantages
* The definition of the neighborhood is so large, it is one of the biggest problems of LIME. For each application, we need to test different kernels and see for each if the predictions make sense. LIME must be applied with caution.
* LIME assumes a linear behavior of the ML model at the local level.
* Synthetic observations are sampled from a Gaussian distribution ignoring the correlation between features. This can generate inconsistent observations used in the train step of local model.
* Instability of predictions, it is difficult to trust the explanations since they can change widely if the sampling is redone.

Shapley values

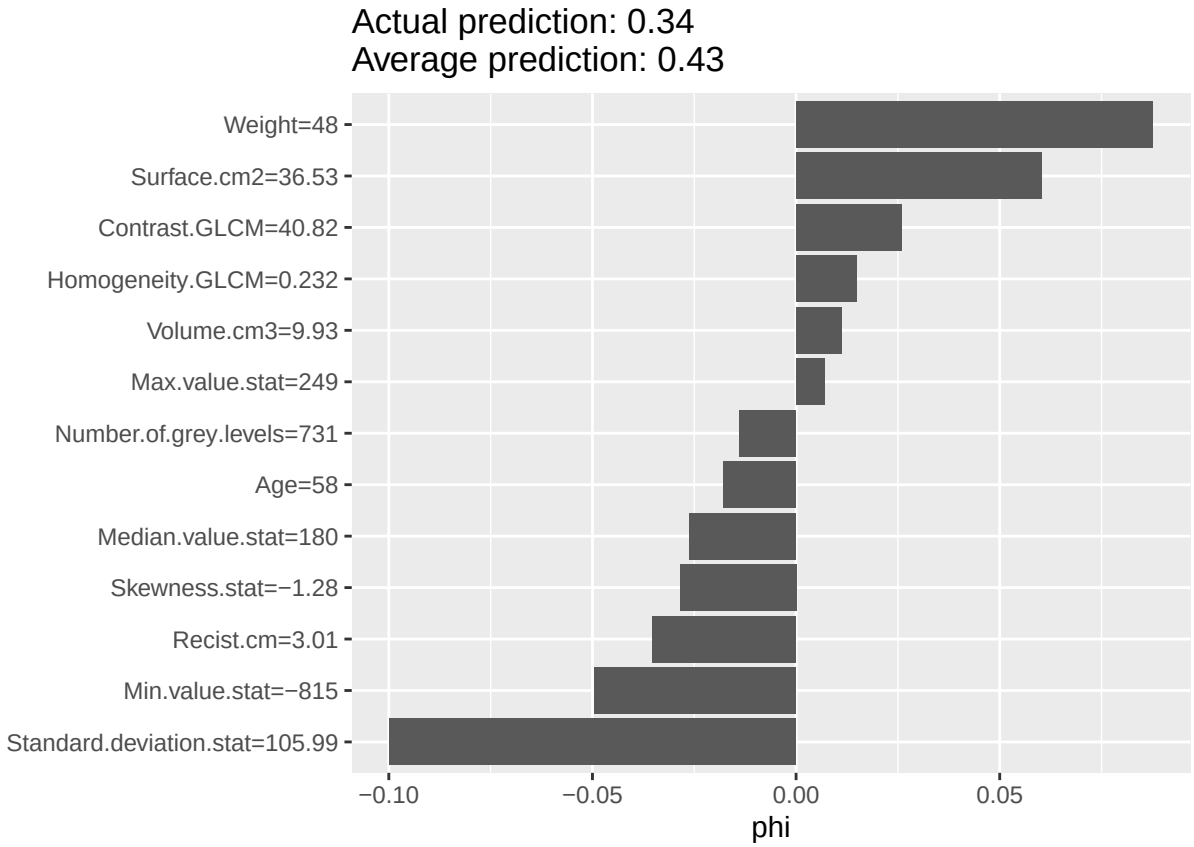
Shapley values help to detect the main differences between the predicted average profile and the prediction of the individual we are trying to explain.

In this example, the predicted mean profile corresponds to patients with *Unfavorable Response*, with a mean probability of 0.43, and the prediction obtained for the individual of interest is 0.34.

```
shapley_values <- iml::Shapley$new(pred.prob.ale, x.interest = test[3, ])
plot(shapley_values)
```

Table 3: Classification of features according their contribution (phi)

	feature	phi
11	Standard.deviation.stat	-0.09974
2	Weight	0.08746
4	Surface.cm2	0.06030
7	Min.value.stat	-0.04944
3	Recist.cm	-0.03516
10	Skewness.stat	-0.02848
9	Median.value.stat	-0.02612
12	Contrast.GLCM	0.02600
1	Age	-0.01778
13	Homogeneity.GLCM	0.01474
6	Number.of.grey.levels	-0.01392
5	Volume.cm3	0.01128
8	Max.value.stat	0.00706




```
phi_high_values <- shapley_values$results[with(shapley_values$results, order(-abs(phi))),]
kable(phi_high_values[,1:2],
  caption = "Classification of features according their contribution (phi)") %>%
  kable_styling(c("striped", "bordered"))
```

This plot shows the Shapley values for a individual in the lung cancer dataset. With a prediction of 0.34, this person's positive response treatment probability is 0.09 below the average prediction of 0.43. The value of *Standard.deviation.stat* feature decreased the probability the most. The sum of contributions yields the

difference between actual and average prediction (0.09).

Warning: the results obtained cannot be generalized to other observations since the Shapley values are valid at individual level.

Advantages
* Shapley is the only method that provides a complete explanation of an individual's decision, using the full set of variables used in the modeling.
* Allows for contrasting interpretations by comparing the prediction of the individual of interest, with the average prediction of the study population.

Disadvantages
* Shapley values can be misinterpreted. The correct interpretation is: considering the 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.
* Like other permutation methods. Shapley values can include inconsistent observations.