

# live :: Local Interpretable (Model-agnostic) Visual Explanations



## Basics

Black-box models, like random forest or extreme gradient boosting, are commonly used due to their high performance. They are very accurate on average on the test set. The problem is, that these models cannot justify nor explain predictions. So why shall you trust their predictions?

The **live** package helps to understand the local behaviour of a black-box model. The idea behind comes from the LIME method [Ribeiro 2016] and is adopted to models based on a mixed data.

### How does it work?

1. Sample a set of points (adjacency) that are close to the instance to be explained.
2. Use the original black-box model to calculate predictions for adjacency.
3. Fit a white-box model to the predictions generated for adjacency.
4. Present the white-model.

### Extra features

- Variable selection.
- Tools for model visualization.
- Focus on interpretable white-boxes

### References

- Bernd Bischl, Michel Lang, Lars Kotthoff, Julia Schiffner, Jakob Richter, Erich Studerus, Giuseppe Casalicchio, and Zachary M. Jones: mlr: Machine learning in R, Journal of Machine Learning Research 17(2016)
- Max Gordon and Thomas Lumley, forestplot: Advanced forest plot using 'grid' graphics, 2017
- Brandon Greenwell, pdp: An R Package for Constructing Partial Dependence Plots, The R Journal 9(2017), no. 1
- S. Lundberg and S.-I. Lee A unified approach to interpreting model predictions, ArXiv e-prints (2017)
- Carolin Strobl, Anne-Laure Boulesteix, Thomas Kneib, Thomas Augustin, and Achim Zeileis, Conditional variable importance for random forests, BMC Bioinformatics 9(2008)
- M. Tulio Ribeiro, S. Singh, and C. Guestrin: Model-Agnostic Interpretability of Machine Learning, ArXiv e-prints (2016)

## Use-Case

*Why are our best and most experienced employees leaving prematurely?* Let's see with a dataset from Kaggle Human Resources competition <https://www.kaggle.com/ludobenistant/hr-analytics/data>.

First, let's create a black-box model for prediction. Here we are using randomForest

```
library(live)
library(randomForest)

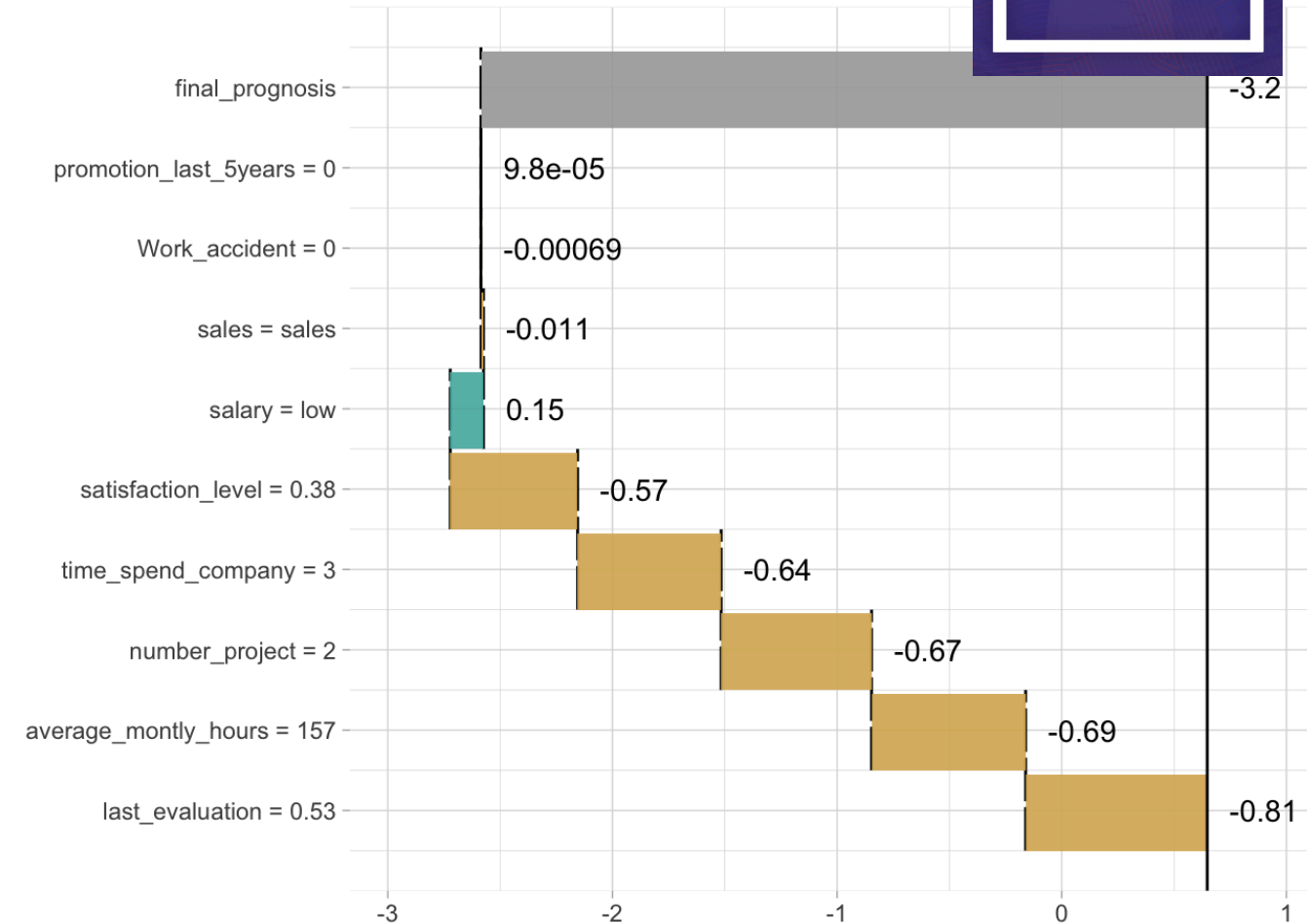
head(HR_data, 2)
#> satisfaction_level last_evaluation
#> 1 0.38 0.53
#> number_project average_monthly_hours
#> 1 2 157
#> time_spend_company Work_accident left
#> 1 3 0 1
#> promotion_last_5years sales salary
#> 1 0 sales low
trees <- randomForest(left~., data = HR_data,
  ntree=1000)
  Type of random forest: classification
  Number of trees: 1000
  No. of variables tried at each split: 3

  OOB estimate of error rate: 0.69%
Confusion matrix:
  0 1 class.error
0 11412 16 0.00140007
1 88 3483 0.02464296
```

Then 1) create the adjacency (function **sample\_locally**), 2) fit the white box (function **fit\_explanations**) and 3) plot the white box.

For linear models one can use forest plots or breakDown plots, for classification trees one can use the core package.

```
similar <- sample_locally(data = HR_data,
  explained_instance = HR_data[2,],
  explained_var = "left",
  size = 2000)
trained <- fit_explanation(
  live_object = similar,
  white_box = "regr.lm",
  selection = FALSE)
plot_explanation(trained, "waterfallplot",
  explained_instance = HR_data[1,])
plot_explanation(trained, "forestplot",
  explained_instance = HR_data[1,])
```



Variable	Observed	Estimate	Lower	Upper
time_spend_company	3	0.23	0.23	0.24
last_evaluation	0.53	2.58	2.5	2.66
average_monthly_hours	157	0.01	0.01	0.01
satisfaction_level	0.38	1.41	1.35	1.48
number_project	2	0.23	0.22	0.24
salarylow		0.15	0.08	0.22
salessales		-0.13	-0.21	-0.05
Work_accident	0	0.06	0.01	0.12
salesmanagement		-0.05	-0.16	0.06
promotion_last_5years	0	-0.07	-0.22	0.09
salesIT		-0.04	-0.14	0.07
salesRandD		-0.02	-0.13	0.09
salestechnical		0.02	-0.08	0.11
salarymedium		-0.01	-0.07	0.06
salesproduct_mng		0.01	-0.1	0.12
salesmarketing		0.01	-0.1	0.11
saleshr		-0.01	-0.13	0.11
salessupport	0	-0.1	-0.1	0.09