DALEX:
Descriptive mAchine Learning EXplanations

Przemysław Biecek Mateusz Staniak

Associate Professor in Machine Learning @ Warsaw University of Technology

15 years of experience in teaching, using and programming in R

Interested in DataVis, Explanations of black-boxes and cancer ML modelling

http://biecek.pl/

MSc student in Mathematics and Statistics @ University of Wrocław

Member of MI2 DataLab.

Working on machine learning interpretability

http://www.mstaniak.pl/

Agenda

Introduction
Model explainers - Continuous variable response
Hands-on
Model explainers - Discrete variable response
Hands-on
Model explainers - Variable importance
Hands-on
Model explainers - Model performance
Hands-on

Prediction explainers - Break-down
Hands-on
Prediction explainers - Live
Hands-on
Summary

What you need to install

```
# From CRAN
# first part
install.packages("DALEX")

# second part
install.packages("auditor")
install.packages("live")
```

Black box models - what they are?

Tradeoff between flexibility vs interpretability.

Complex structure
Ensembles like: Random Forests, Gradient Boosting Machines, Neural Networks have complex structure, highly non-linear and non-additive. It is hard to understand how input variables affect the final model outcome.

Wide input space
Even linear and additive models like Generalised Linear Regression Models, Generalised Additive Models, Rule Based Models suffer for lack of interpretability if there is a lot of non zero model components.

Non uniform/balanced variable distribution
Even additive models with sparse input space it may be hard to trace variable contributions for a single prediction if distribution of some variables is not balanced/uniform.

Why do we need explanations for complex models?

Domain validation
In ML the model overfitting is hard to control due to large number of tested classes of models, hyper-parameters optimisation, data-leaking in frequent validations. Explanations of Machine Learning models give the possibility of cross validate model behaviour agains the domain knowledge.

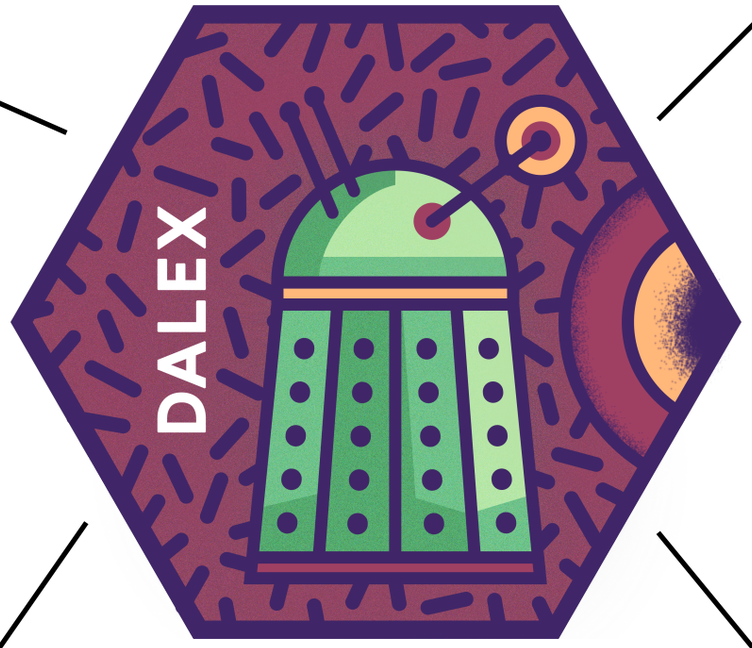
New knowledge
Explanations for well working models may lead to new insights.

Trust
If model predictions are to be consumed they need to be trusted. For high-stake decisions, like medical treatment, this may be even a requirement.

GDPR - General Data Protection Regulation
Right to explanation of all decisions made by automated or artificially intelligent algorithmic systems
(not in the GDPR text, yet often discussed).

DALEX is a set of tools that
helps to understand the way complex predictive models work

How good is the predictive model?



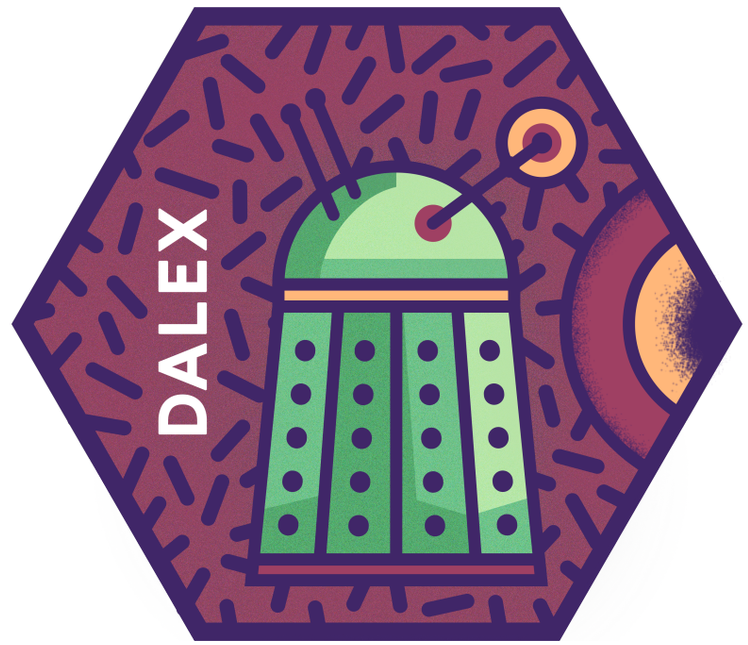
Which variables are the most important in general?

How good is the model fit?

Which variables influence the single prediction?

Variable Explainers
package: pdp, ALEPlot, **factorMerger**

Structure Explainers
package: randomForest



Model Management
package: **archivist**

Model Diagnostic Tools
package: **auditor**, ggfortify

Model Performance Explainers
package: **auditor,** ROCR, caret, mlr

Model Predictions Explainers
package: **breakDown**, **live**, shapleyr, lime

Golden era for Machine Learning Models

nature biomedical engineering

ARTICLES

https://doi.org/10.1038/s41551-018-0195-0

Prediction of cardiovascular risk factors from retinal fundus photographs via deep learning

Ryan Poplin[1,4], Avinash V. Varadarajan[1,4], Katy Blumer[1], Yun Liu[1], Michael V. McConnell[2,3], Greg S. Corrado[1], Lily Peng[1,4]* and Dale R. Webster[1,4]

Traditionally, medical discoveries are made by observing associations, making hypotheses from them and then designing and running experiments to test the hypotheses. However, with medical images, observing and quantifying associations can often be difficult because of the wide variety of features, patterns, colours, values and shapes that are present in real data. Here, we show that deep learning can extract new knowledge from retinal fundus images. Using deep-learning models trained on data from 284,335 patients and validated on two independent datasets of 12,026 and 999 patients, we predicted cardiovascular risk factors not previously thought to be present or quantifiable in retinal images, such as age (mean absolute error within 3.26 years), gender (area under the receiver operating characteristic curve (AUC) = 0.97), smoking status (AUC = 0.71), systolic blood pressure (mean absolute error within 11.23 mmHg) and major adverse cardiac events (AUC = 0.70). We also show that the trained deep-learning models used anatomical features, such as the optic disc or blood vessels, to generate each prediction.

Risk stratification is central to identifying and managing groups at risk for cardiovascular disease, which remains the leading cause of death globally[1]. Although the availability of cardiovascular disease risk calculators, such as the Pooled Cohort equations[2], Framingham[3–5] and Systematic COronary Risk Evaluation (SCORE)[6,7], is widespread, there are many efforts to improve risk predictions. Phenotypic information, particularly of vascular health, may further refine or reclassify risk prediction on an

changes[22,23] ar
temic health
clinical utility
work, we dem
cardiovascula

Machine le
ety of classifi
of eye disease

SCIENTIFIC REPORTS

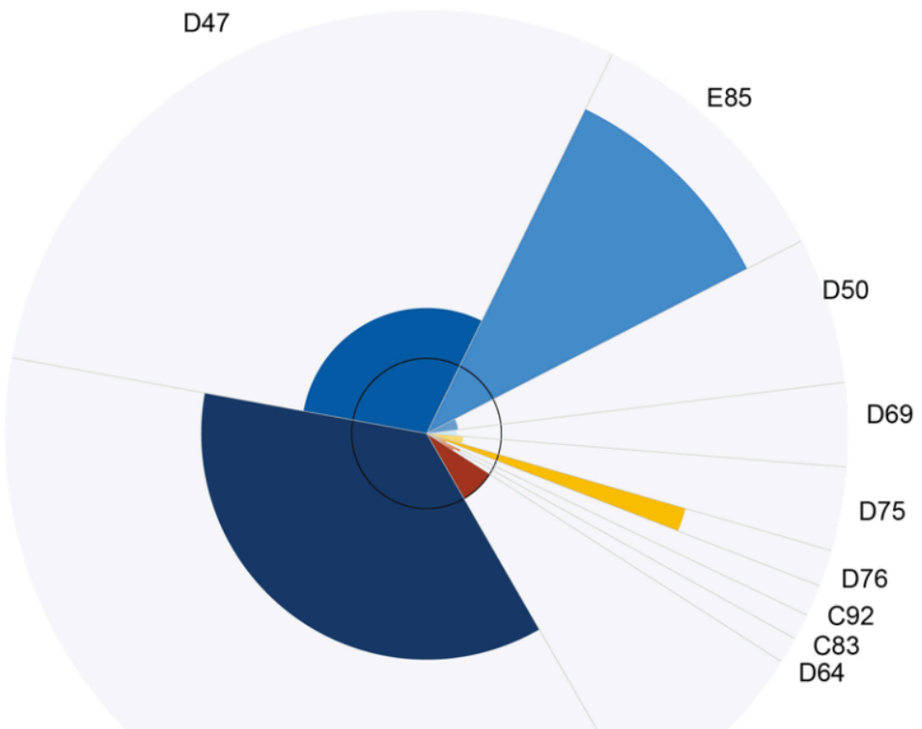
OPEN

An application of machine learning to haematological diagnosis

Gregor Gunčar[1], Matjaž Kukar[1], Mateja Notar[1], Miran Brvar[2], Peter Černelč[3], Manca Notar[1] & Marko Notar[1]

ved: 2 August 2017
ted: 14 December 2017
hed online: 11 January 2018

Quick and accurate medical diagnoses are crucial for the successful treatment of diseases. Using machine learning algorithms and based on laboratory blood test results, we have built two models to predict a haematologic disease. One predictive model used all the available blood test parameters and the other used only a reduced set that is usually measured upon patient admittance. Both models produced good results, obtaining prediction accuracies of 0.88 and 0.86 when considering the list of five most likely diseases and 0.59 and 0.57 when considering only the most likely disease. The models did not "finger and inc clinical special tests al unprec



ICD code	Prediction	Information score	Disease category
C90	36.20%	2.01	Multiple myeloma and malignant plasma cell neoplasms
D47	29.40%	0.67	Other neoplasms of uncertain or unknown behaviour of lymphoid. haematopoietic and related tissue
E85	10.20%	3.81	Amyloidosis
D50	5.60%	-1.37	Iron deficiency anaemia
D69	3.20%	-1.50	Purpura and other haemorrhagic conditions
D75	3.20%	-1.05	Other diseases of blood and blood-forming organs
D76	1.40%	2.60	Certain diseases involving lymphoreticular tissue and reticulohistiocytic system
C92	1.20%	-2.55	Myeloid leukaemia
C83	1.00%	-1.01	Diffuse non-Hodgkin lymphoma
D64	1.00%	-1.41	Other anaemias

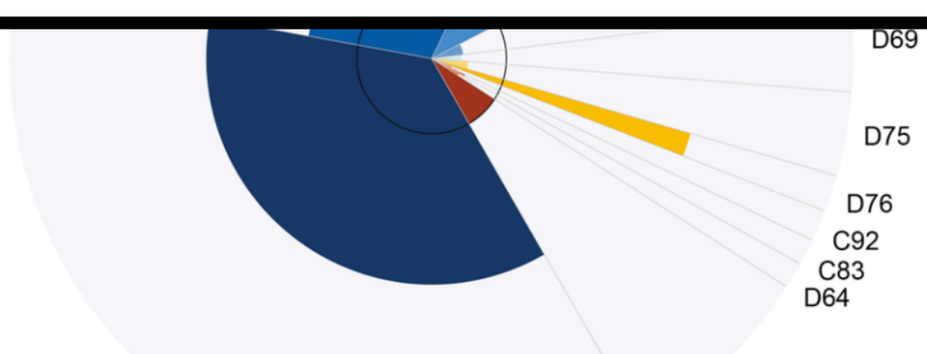
SCIENTIFIC REPORTS

OPEN

An application of machine learning to haematological diagnosis

ved: 2 August
ted: 14 Dece
hed online: 1

Imagine that these are predictions for you.
Would you trust them?



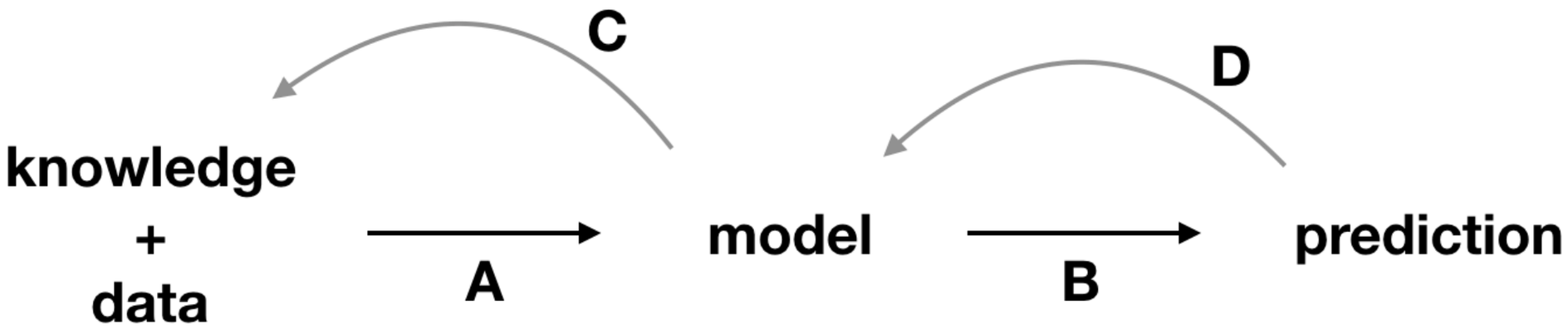
D69	3.20%	-1.50	Purpura and other haemorrhagic conditions
D75	3.20%	-1.05	Other diseases of blood and blood-forming organs
D76	1.40%	2.60	Certain diseases involving lymphoreticular tissue and reticulohistiocytic system
C92	1.20%	-2.55	Myeloid leukaemia
C83	1.00%	-1.01	Diffuse non-Hodgkin lymphoma
D64	1.00%	-1.41	Other anaemias

Predictive models for apartment prices

Typical workflow in ML

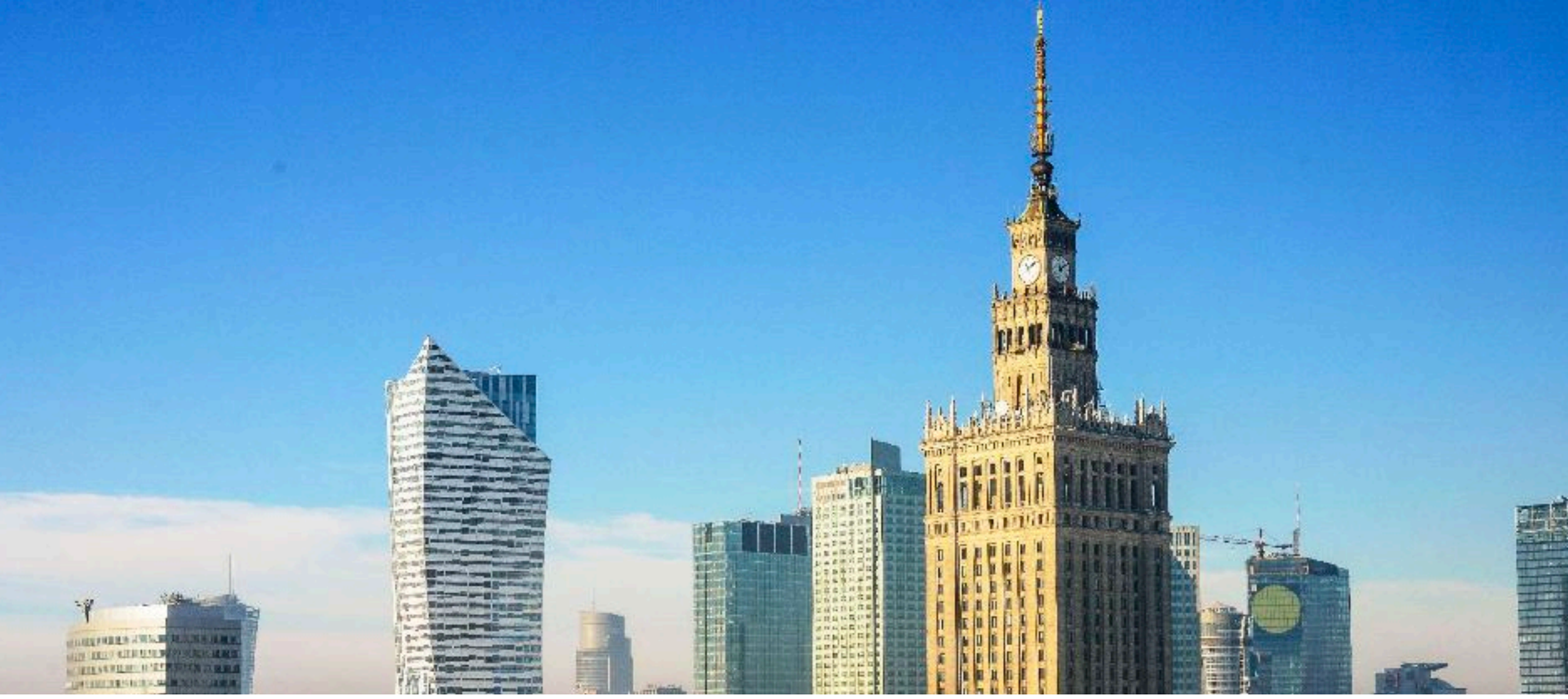
A. Modelling is a process in which domain knowledge and data are turned into models.
B. Models are used to generate predictions.

Typical workflow in ML



A. Modelling is a process in which domain knowledge and data are turned into models.
B. Models are used to generate predictions.
C. Understanding of model structure may increase our knowledge and in consequence leads to a better model. *DALEX helps here.*
D. Understanding of drivers behind particular model predictions may help to correct wrong decisions and in consequence leads to a better model. *DALEX helps here.*

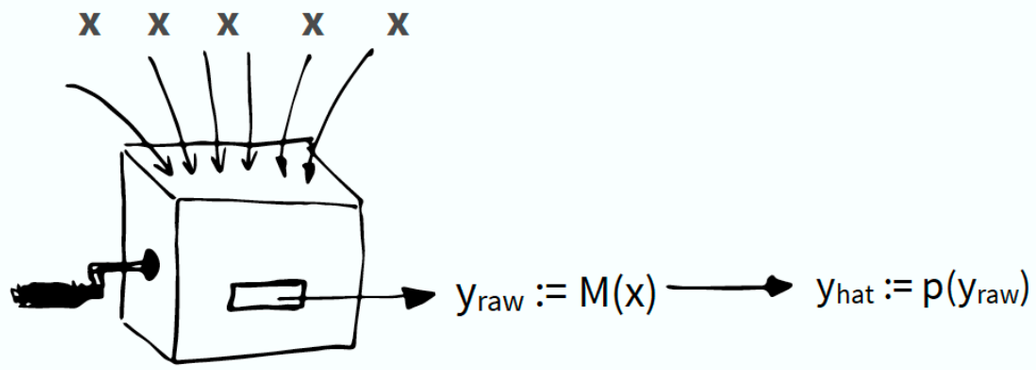
http://www.warszawa.pl/biznes/panorama-warszawy/galeria/dzis_palac_kultury_i_nauki/



```
library("DALEX")
head(apartments)
```

m2.price	construction.year	surface	floor	no.rooms	district
5897	1953	25	3	1	Srodmiescie
1818	1992	143	9	5	Bielany
3643	1937	56	1	2	Praga
3517	1995	93	7	3	Ochota
3013	1992	144	6	5	Mokotow

A)

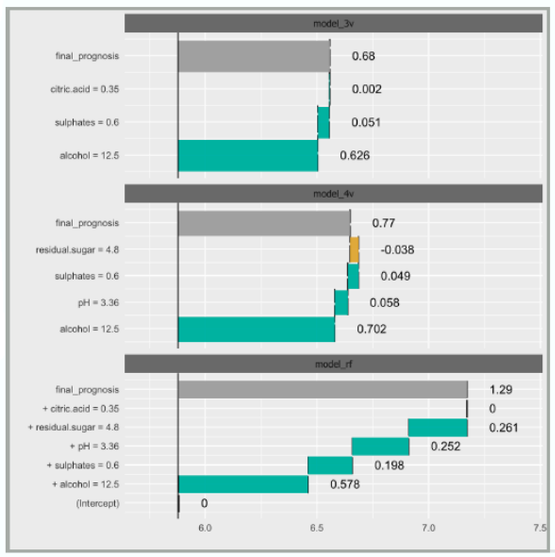


B)

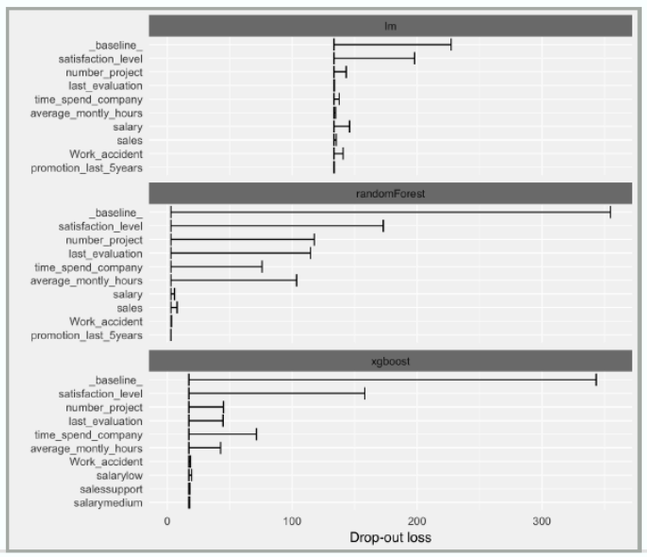
explain(model; data; y; predict_function; trans)

C)

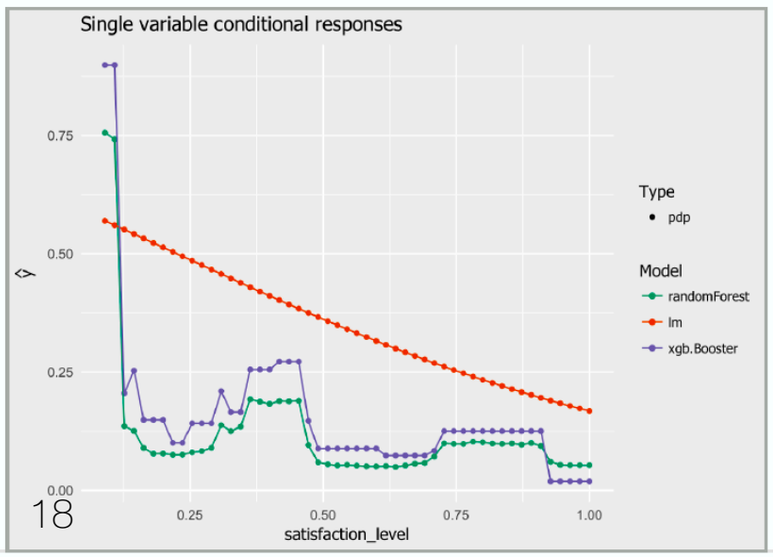
single_prediction(explainer, x)

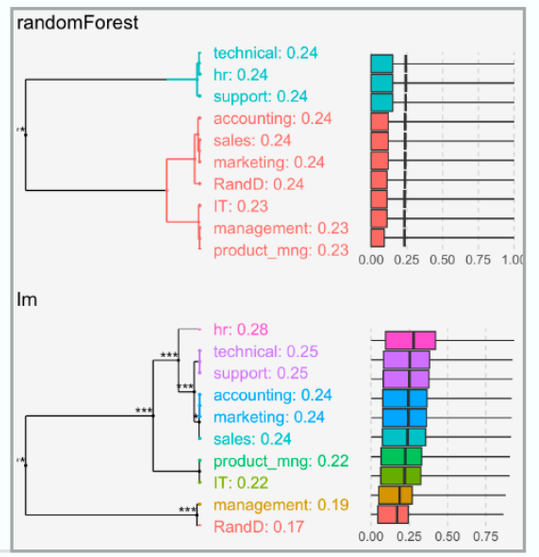


variable_dropout(explainer)

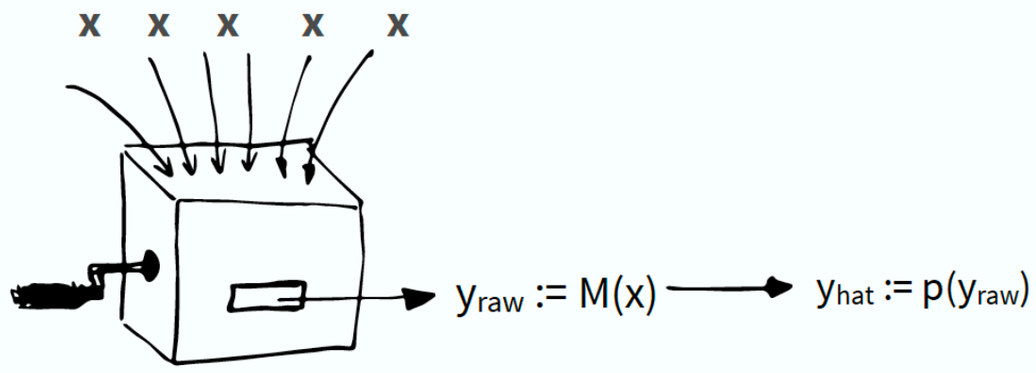


single_variable(explainer, variable)





A)



B)

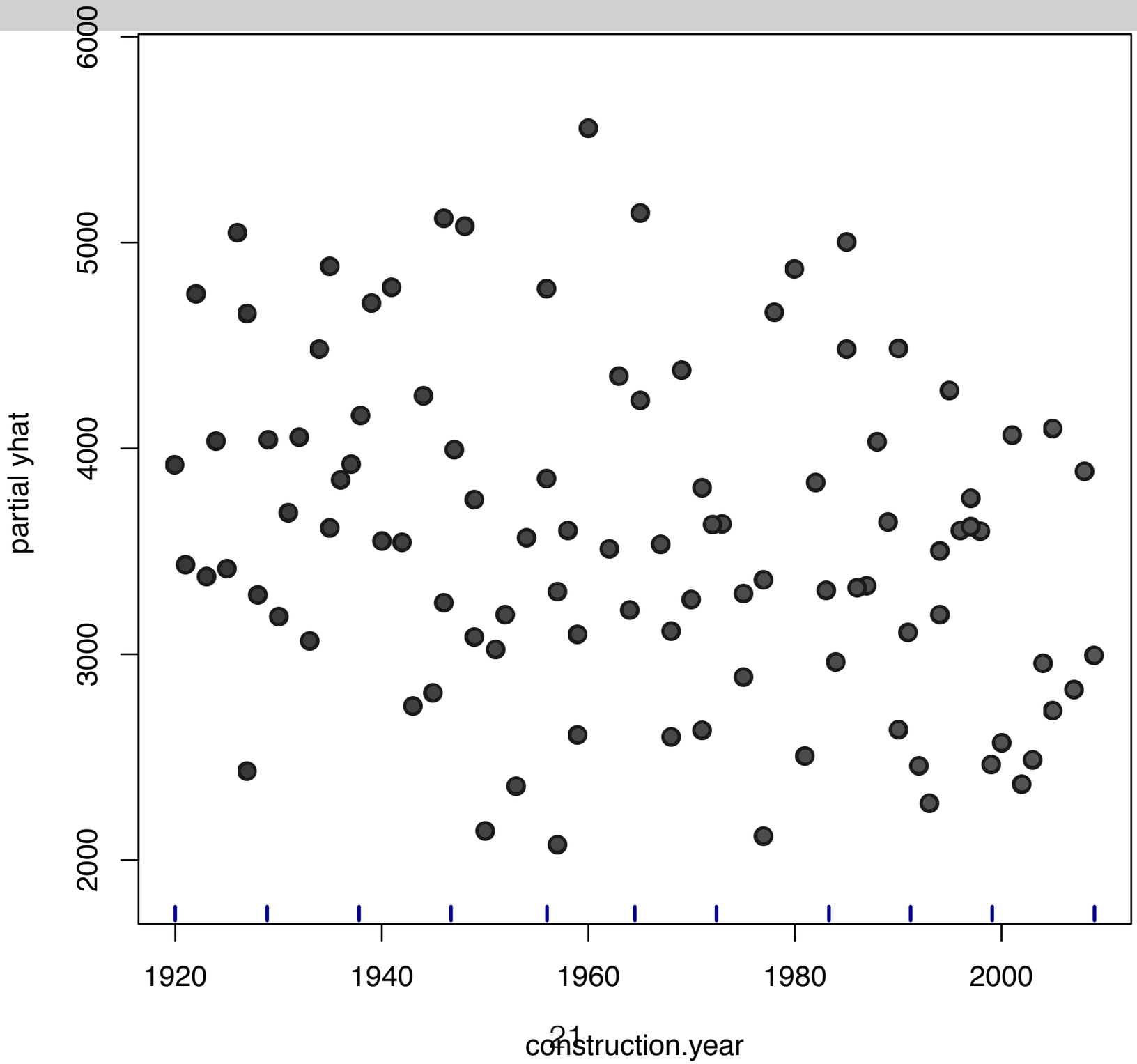
explain(model; data; y; predict_function; trans)

```
library("DALEX")
apartments_lm_model <- lm(m2.price ~ construction.year + surface + floor +
                          no.rooms + district, data = apartments)
library("randomForest")
set.seed(59)
apartments_rf_model <- randomForest(m2.price ~ construction.year + surface + floor
                          no.rooms + district, data = apartments)

explainer_lm <- explain(apartments_lm_model,
                        data = apartmentsTest[,2:6], y = apartmentsTest$m2.price)
explainer_rf <- explain(apartments_rf_model,
                        data = apartmentsTest[,2:6], y = apartmentsTest$m2.price)
```

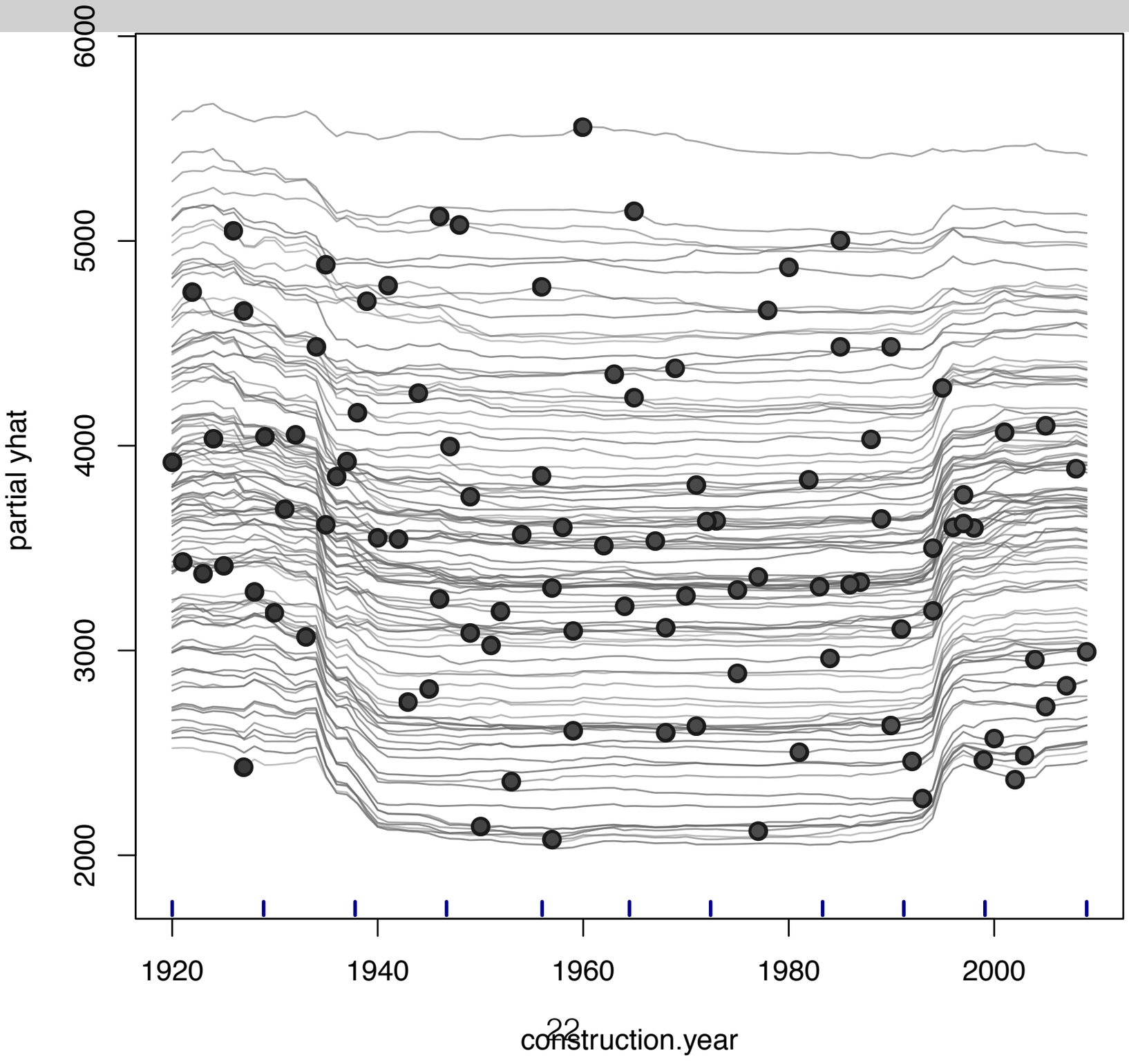
Model explainers - Continuous variable response

How a single feature affects the model response?

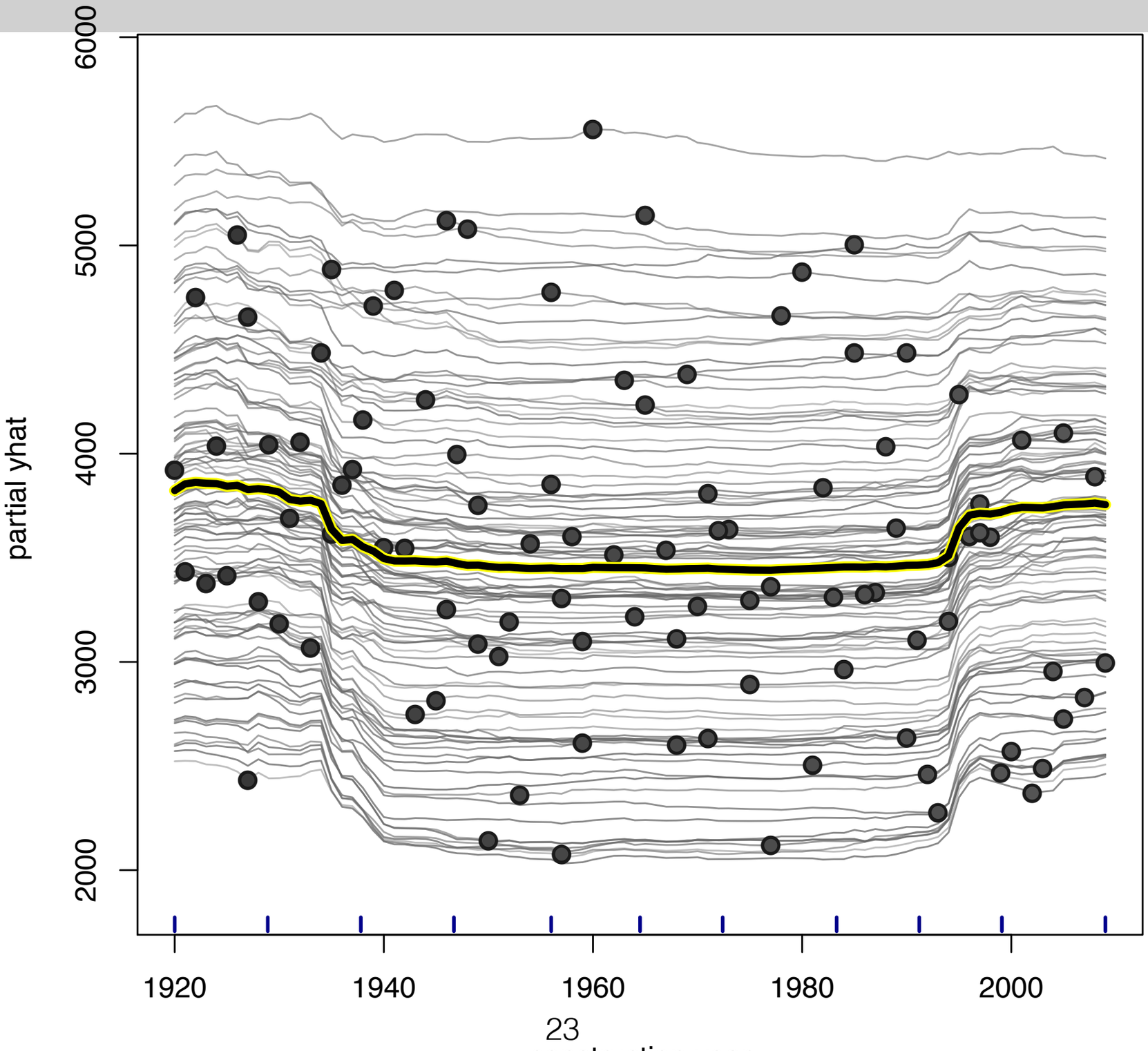
How a single variable affect the response?



How a single variable affect the response?



How a single variable affect the response?



factorMerger
Cheat Sheet

Agnieszka Sitko [aut, cre]
Przemysław Biecek [aut, ths]
University of Warsaw

Introduction

How to check if averages are different among k groups? Use **ANOVA**!

How to visualise how these groups are different? Use **factorMerger**!

The aim of **factorMerger** is to provide informative and easy to understand visualisations of *post-hoc* comparisons. It gives consistent and non-overlapping adaptive fusing of groups based on likelihood ratio test (LRT). The package **factorMerger** works for wide spectrum of families like Gaussian, binomial or survival.

Results from the adaptive fusing are presented with the *Merging Paths Plots* - a hierarchical representation of LRT-based distances among groups.

In addition, the *Generalized Information Criterion (GIC)* is presented for fused models. This criterion may be used to choose the optimal segmentation of groups.

Graphical summary of the variable of interest in each group is presented in the right panel.

Find more in *https://arxiv.org/abs/1709.04412*

Example

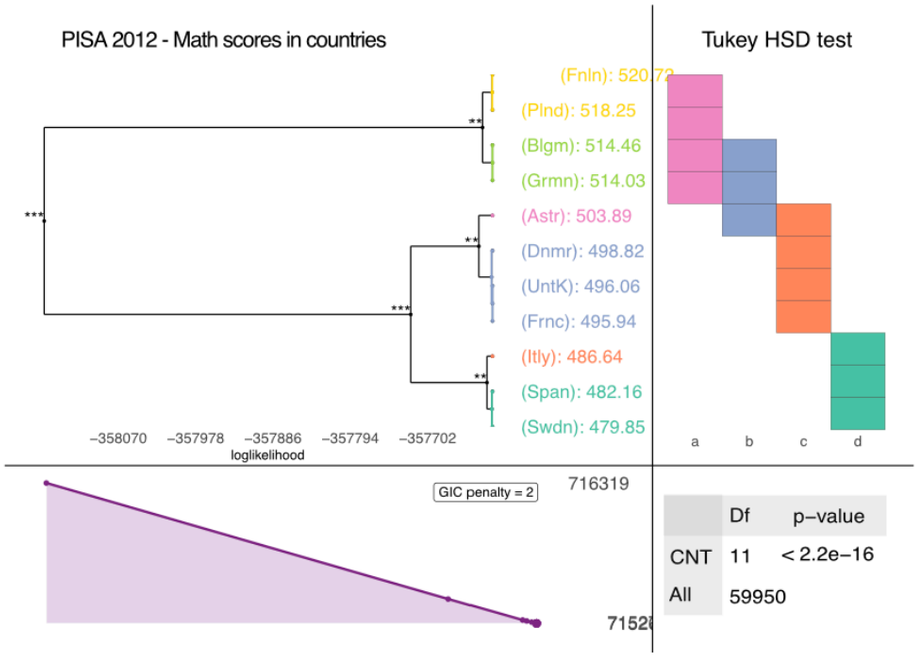
```
library(factorMerger)

fmAll <- mergeFactors(
    response  = pisaEuro$math,
    factor = pisaEuro$country,
    method = "fast-adaptive",
    family = "gaussian")

print(fmAll)

plot(fmAll,
    panel = "all",
    responsePanel = "tukey")
```

Set of tools to support post-hoc segmentation of groups

Four panels to see it all

A. The Merging Paths Plot panel
Shows the hierarchical structure of similarity between groups.
Stars presents how significant are differences between two clusters.
Height of the join corresponds to the likelihood of the combined model.

B. The response panel
Shows graphical summaries for group separately.
Use the responsePanel argument to select the desired summary.



C. The GIC panel
Shows the *Generalized Information Criterion* for all models in MPP panel.
Default *penalty=2* corresponds to the AIC criterion.
Colours in the MPP panel corresponds to the optimal model.

D. The summary panel
Shows the no. groups, no. observations,
and the p-value for global hypothesis
that parameter of interest is equall in all groups.

Key arguments of factorMerger

The **factorMerger** package is highly customisable.

The **panel** argument selects panels to be plotted.

The **nodesSpacing** argument selects how MPP nodes shall be spaced.

The **responsePanel** argument selects which summary shall be presented in panel B.

The **responsePanelPalette** argument selects the colouring palette.

The **colorClusters** argument selects if segments of groups shall be colour coded.

The **penalty** argument selects penalty for GIC criterion.

Argument	Summary
`panel = "all"`	All panels
`panel = "response"`	Panels A and B
`panel = "GIC"`	Panels A and C
`panel = "tree"`	Only panel A

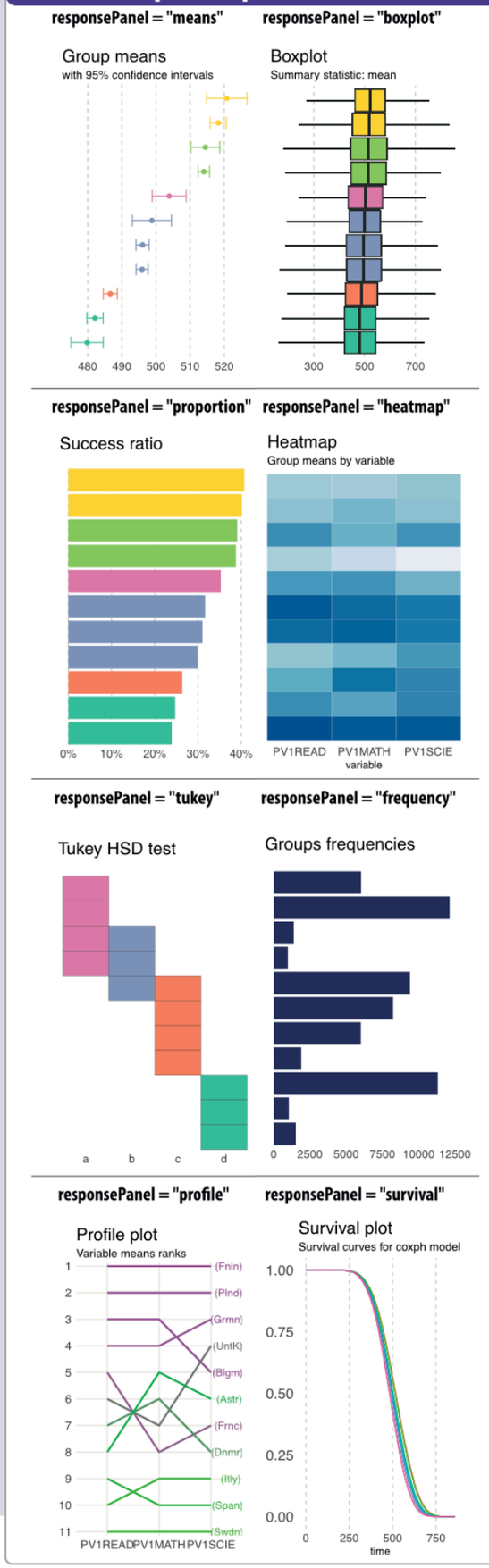
The **method** argument determines how pairs of groups are considered for grouping. The most computational demanding is *adaptive*. For larger number of groups one may consider only pairs of groups with closest averages or use the *fast-fixed* heuristic (fastest solution).

Argument	Summary
`method = "adaptive"`	Compare all pairs of groups
`method = "fast-adapive"`	Compare nearby groups
`method = "fixed"`	Delete or merge heuristic
`method = "fast-fixed"`	Reduced DMR method

The **family** argument determines what models are to be compared with likelihood ratio test. Either linear models, generalised linear models or Cox regression models.

Argument	Summary
`family = "gaussian"`	For 1/n-dimensional Gaussian
`family = "binomial"`	For binomial regression
`family = "survival"`	For Cox regression

The response panel summaries

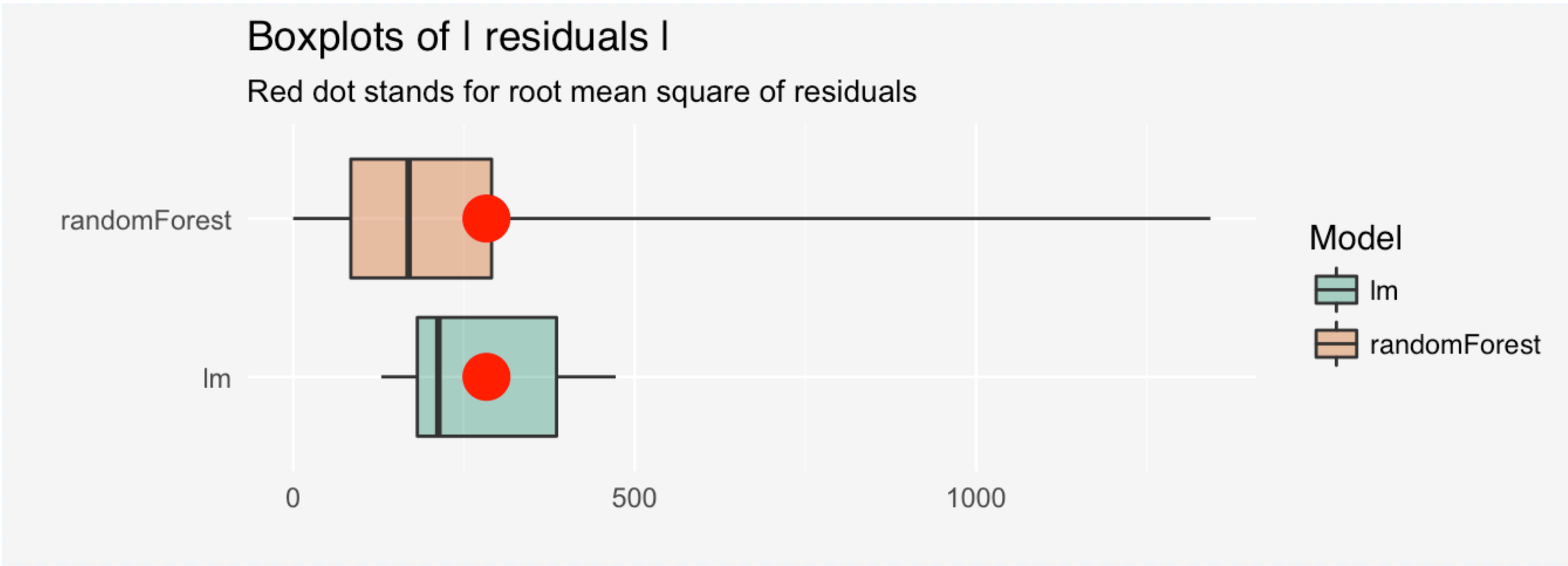


3. Model performance

How the accurate is the model?

Accuracy as a single number is not enough!

```
plot(mp_lm, mp_rf, geom = "boxplot")
```



Error analysis with auditor : : CHEAT SHEET

Basics

Package **auditor** provides several methods for model verification and validation by error analysis. This includes both, graphical methods and scores, which that can be useful for comparison of models performance.

Residual analysis is widely used for linear and generalized linear models, so there are many statistical tools to evaluate the goodness of fit. However, these methods are not suitable for each machine learning model. In particular, it is difficult to apply them to black box models such us random forest, due to the lack of information about the error distribution.

The tools included in auditor can be used to assess the fit of many types of models, including black boxes.

MODEL PREPARATION

We will show the use of a package for a logistic regression model.
The example uses *the Pima Indian Diabetes* data set.

```
library(mlbench)
data("PimaIndiansDiabetes")

mod_glm <- glm(diabetes~.,
               family=binomial,
               data=PimaIndiansDiabetes)
```

In order to analyse the model residuals, we need to convert model into a uniform structure readable by the auditor package.

```
library(auditor)
audit_glm <- audit(mod_glm)
```

An object created with the **audit()** function can be used to draw different diagnostic plots. We will show some of them in this cheatsheet.

More detailed description and additional functionalities are presented in the package vignettes which can be found on the auditor website:
https://mi2-warsaw.github.io/auditor

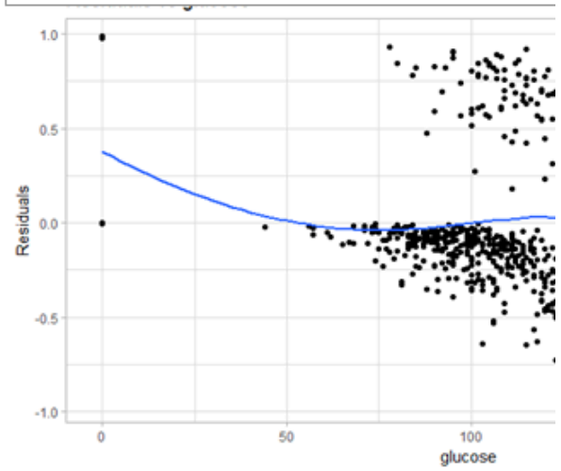
REFERNCES

- *Moral, R., Hinde, J., & Demétrio, C. (2017). Half-Normal Plots and Overdispersed Models in R: The hnp Package. Journal of Statistical Software, 81(10), 1 - 23.*

Basic plots

The **plot()** function can be used to draw several di
By default it is drawn *Residuals vs variable plot*.

```
plot(audit_glm, variable = "glucose"
```

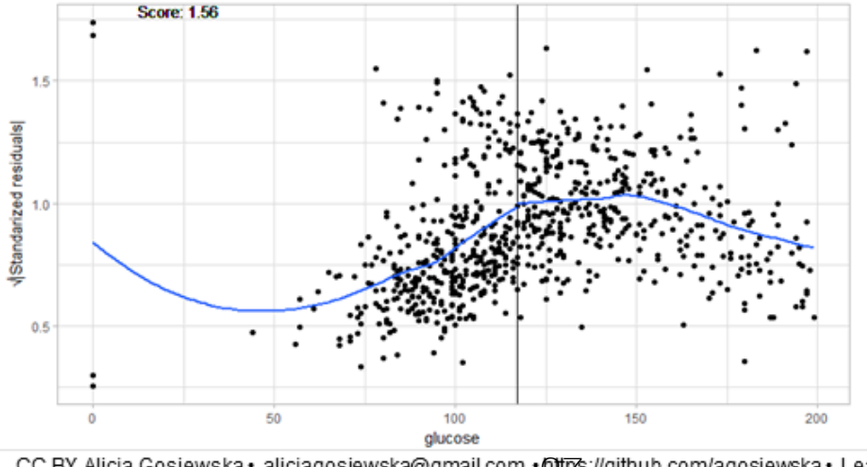


In the example above there were used response
to provide any predict and residual function.
To use non-defaulted functions to calculate the residuals and model prediction, you can use arguments **residual.function** and **predict.function** while using an audit function.

```
p.fun <- function(model, data){
           predict(model, data, type="link")
         }
r.fun <- function(model,y){residuals(model)}

audit_glm_dev <- audit(mod_glm,
                       predict.function = p.fun,
                       residual.function = r.fun)
```

Use **type=plot name** to draw different types of diagnostic plots.
The available parameter values are **'ACF', 'Autocorrelation', 'Cook', 'HalfNormal', 'Residuals' and 'ScaleLocation'**.

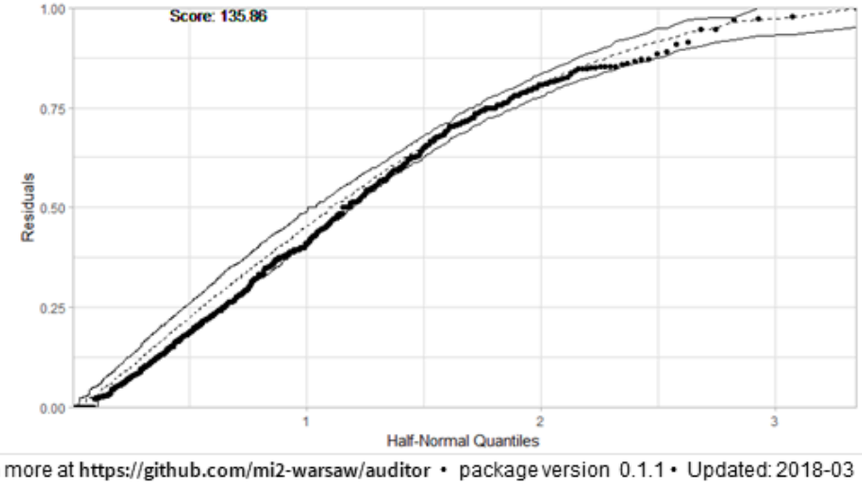
```
plot(audit_glm_dev, type="ScaleLocation",
  variable = "glucose")
```



auditor

Half-Normal plots allow to check the model fit but comparing two models may be difficult.
A useful tool to compare goodness-of-fit of two models is score displayed on the plot. Score is a sum of logarithms of estimated PDF at each point. It is calculated on the basis of simulated data, so it may differ between function calls.

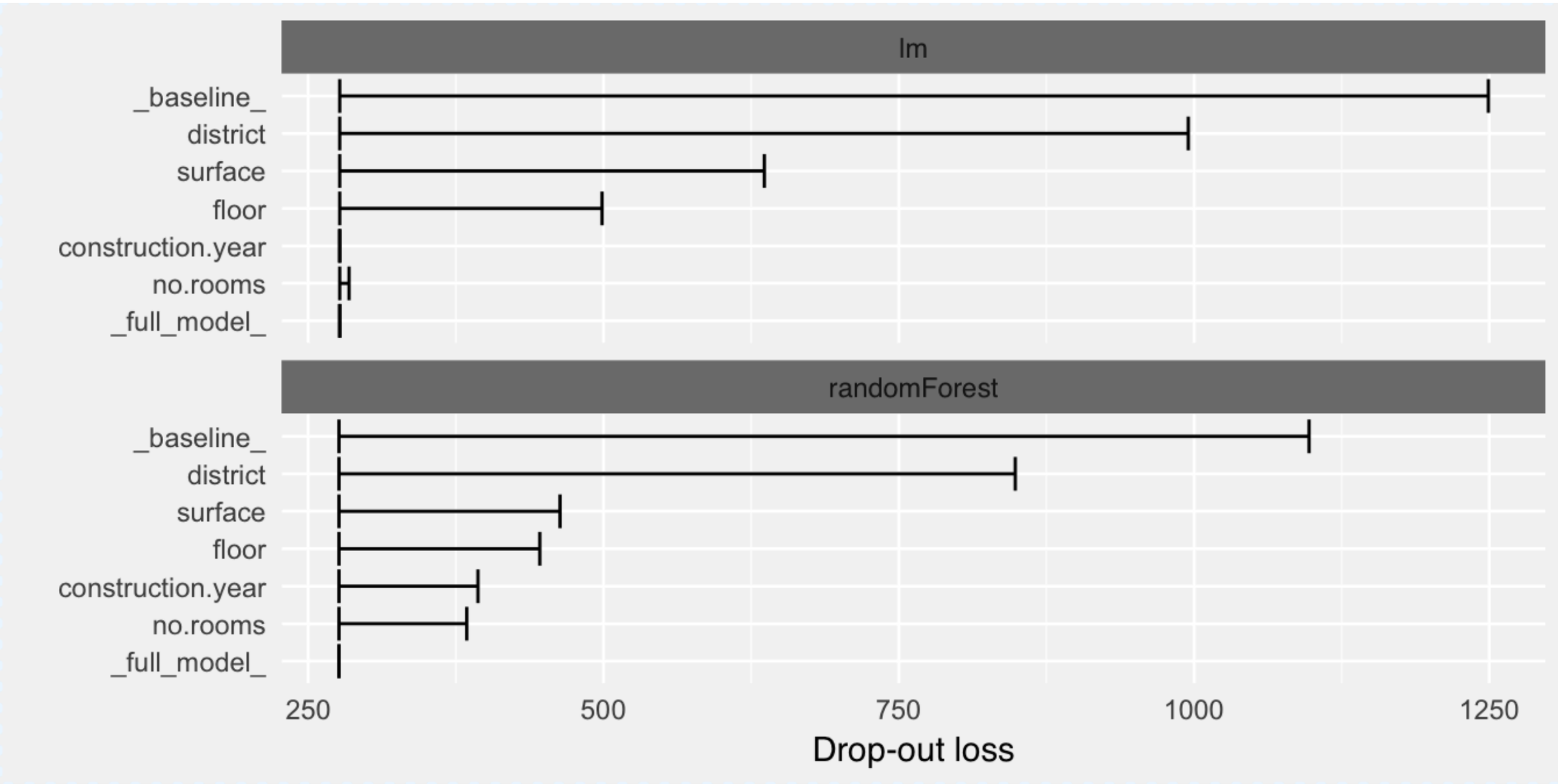
```
library(randomForest)
mod_rf <- randomForest(diabetes~.,
                       data=PimaIndiansDiabetes, ntree=100)
audit_rf <- audit(mod_rf)
plotHalfNormal(audit_rf)
```



4. Variable importance

Which variables are influential?

Which variables are influential?



Which variables are influential?

$$\hat{e}_{\text{orig}}(f) := \frac{1}{n}\sum_{i=1}^{n} L(f, \mathbf{Z}_{[i,\cdot]}) = \frac{1}{n}\sum_{i=1}^{n} L\{f, (\mathbf{y}_{[i]}, \mathbf{X}_{1[i,\cdot]}, \mathbf{X}_{2[i,\cdot]})\} = \hat{\mathbb{E}} L(f, Z),$$

$$\begin{aligned}\hat{e}_{\text{switch}}(f) &:= \frac{1}{n(n-1)}\sum_{i=1}^{n}\sum_{j\neq i} h_f(\mathbf{Z}_{[i,\cdot]}, \mathbf{Z}_{[j,\cdot]}) \\ &= \frac{1}{n(n-1)}\sum_{i=1}^{n}\sum_{j\neq i} L\{f, (\mathbf{y}_{[j]}, \mathbf{X}_{1[i,\cdot]}, \mathbf{X}_{2[j,\cdot]})\}.\end{aligned}$$

$$\text{Model Class Reliance} = \frac{\hat{e}_{\text{switch}}(f)}{\hat{e}_{\text{orig}}(f)},$$

Model Class Reliance: Variable Importance Measures for any Machine Learning Model Class, from the "Rashomon" Perspective
Aaron Fisher, Cynthia Rudin, Francesca Dominici

DALEX: : variable_importance() - Explanations for Variable Importance

Basics

Black-box models, like random forest or extreme gradient boosting machines, are commonly used due to their high performance. They are very flexible, what often results in high accuracy.

The problem is, that due to their complicated structure it is hard to understand which variables were the most influential for a particular model prediction

Variable Importance Explainers are designed to assess the influence of a single variable on the final model accuracy.

How does it work?
The concept is straightforward, calculate how much we will loose on accuracy if a selected variable is perturbed.

For a selected loss function the model loss is calculated for the trained model applied to the original dataset against the original target. Then the drop in model loss is calculated for each variable in the dataset. It is calculated as the loos for a model applied to a dataset with selected column being permuted.

The variable importance plots shows the initial loss of the trained model, size of the additional losses that come from permutations of selected variables and the 'baseline' which is the loss for permuted model responses.

It is useful when different model are compared, since on a single plot we see the initial model performance and also drops that come from permutations of a single variable in the selected model.

References

- Molnar, Christoph. 2018. *Interpretable Machine Learning*. https://christophm.github.io/
- Breiman, Leo. 2001. *Random Forests.* Machine Learning 45 (1). Springer: 5–32.
- Fisher, Aaron, Cynthia Rudin, and Francesca Dominici. 2018. *Model Class Reliance: Variable Importance Measures for any Machine Learning Model Class, from the 'Rashomon' Perspective.* http://arxiv.org/abs/1801.01489.

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.

Here we are building a random forest model. The nice thing about this model is that it embeds some variable importance measure.

```
library("randomForest")
HR_rf_model <- randomForest(left~., data =
breakDown::HR_data, ntree = 100)
importance(HR_rf_model)
                      IncNodePurity
satisfaction_level       896.584411
last_evaluation          294.195269
number_project           496.397115
average_montly_hours     398.495343
```

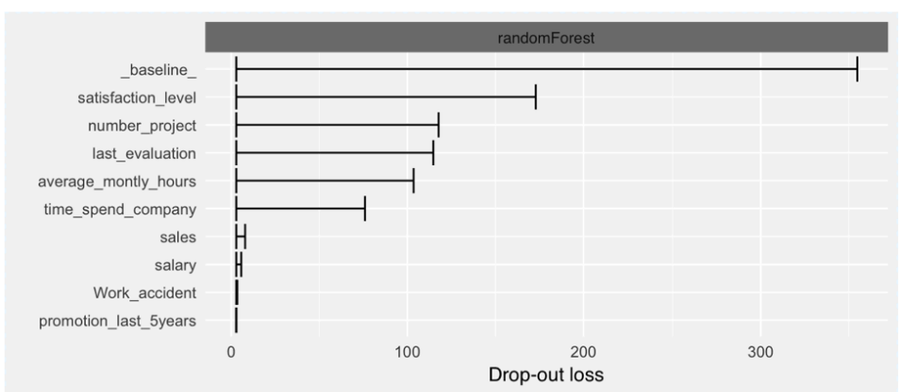
So, now let's build an explainer and use it to calculate our model agnostic variable importance measure with **variable_importance()** function.

```
explainer_rf  <- explain(HR_rf_model, data = HR_data,
                         y = HR_data$left)
vd_rf <- variable_importance(explainer_rf, type = "raw")
vd_rf
                variable dropout_loss        label
               _baseline_   335.556063 randomForest
     satisfaction_level   168.001733 randomForest
        last_evaluation   100.714534 randomForest
         number_project   100.568214 randomForest
    time_spend_company    90.349207 randomForest
```

Values are different because different measures of importance are being calculated, but the order is more or less the same. Now we are ready to plot the variable importance with the generic **plot**() function.

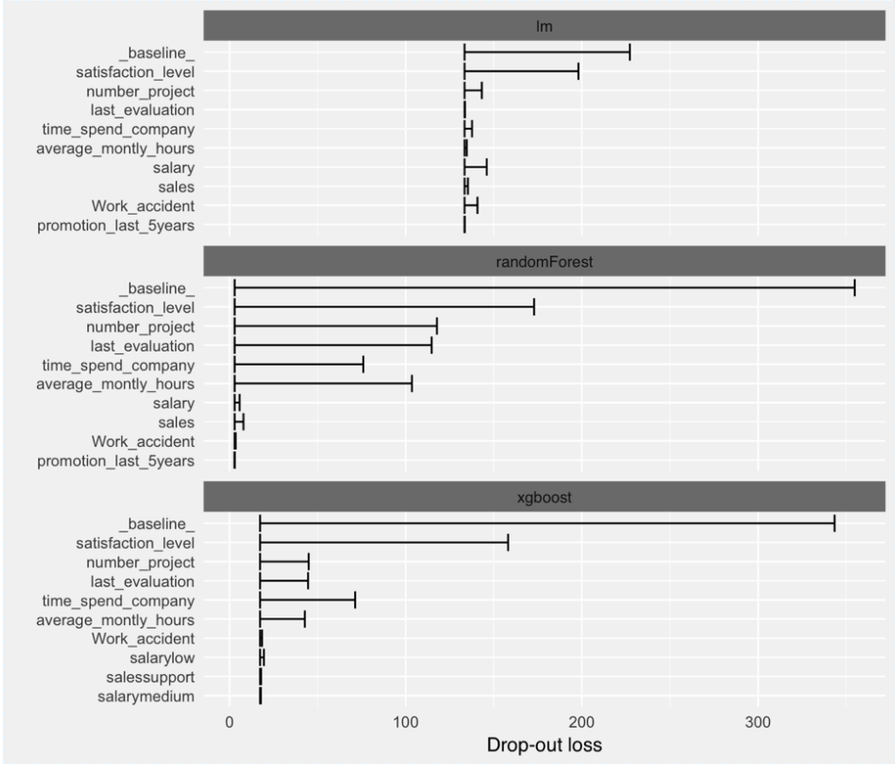
Note that in addition to *type="raw"* one can also use *"difference"* or *"ratio"* as suggested in the Fisher et all 2018 article.

```
plot(vd_rf)
```



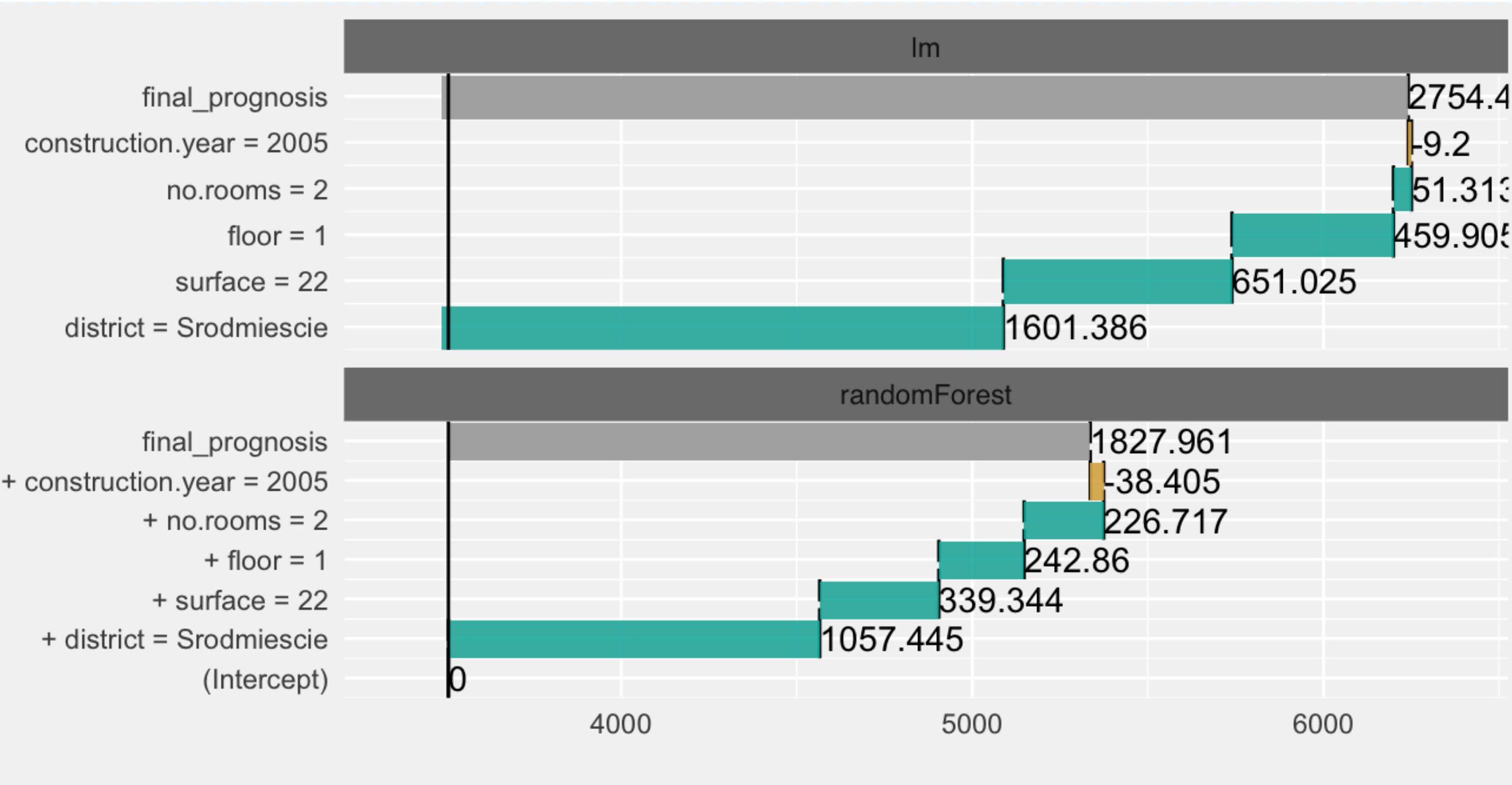
A very useful feature of the DALEX package is the capability to overlay responses from different models in a single plot.
Below we present results for a Random Forest, GLM and XGBoost.

```
library("xgboost")
mm_train <- model.matrix(left~.-1, HR_data)
data_train <- xgb.DMatrix(model_martix_train,
                label = HR_data$left)
param <- list(max_depth = 2,
      objective = "binary:logistic", eval_metric = "auc")
HR_xgb <- xgb.train(param, data_train, nrounds = 50)
ex_xgb <- explain(HR_xgb,  data = mm_train,
                  y = HR_data$left, label = "xgboost")
vd_xgb <- variable_importance(ex_xgb, type = "raw")

HR_glm <- glm(left~., data=HR_data, family = "binomial")
ex_glm <- explain(HR_glm, data=HR_data, y = HR_data$left)
logit <- function(x) exp(x)/(1+exp(x))
vd_glm <- variable_importance(ex_glm, type = "raw")
plot(vd_rf, vd_glm, vd_xgb)
```

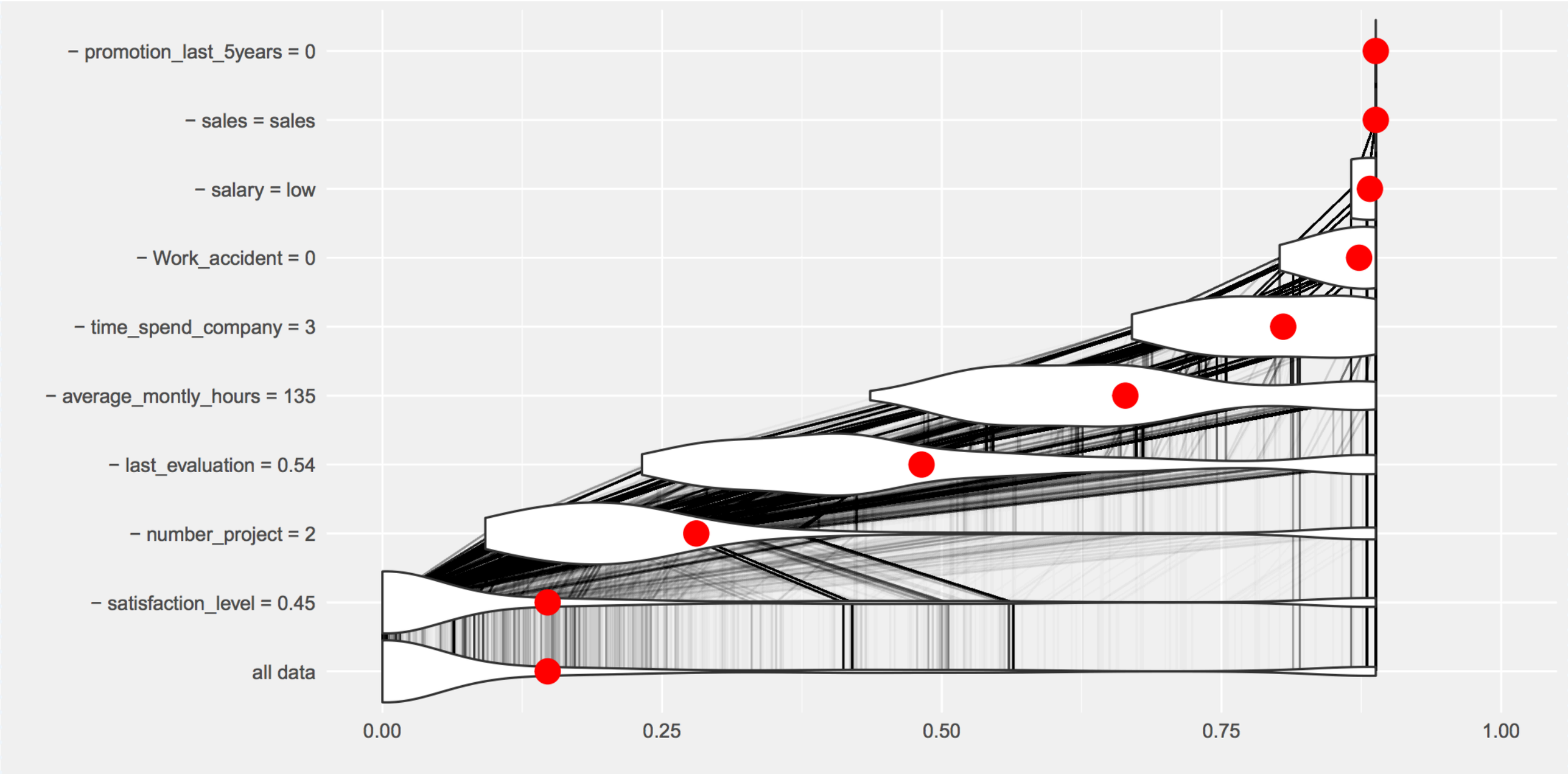


1. Prediction breakDown

Which features influence a single prediction?



Which features influence a single prediction?



breakDown plots : : **visual explanations for lm/glm models**

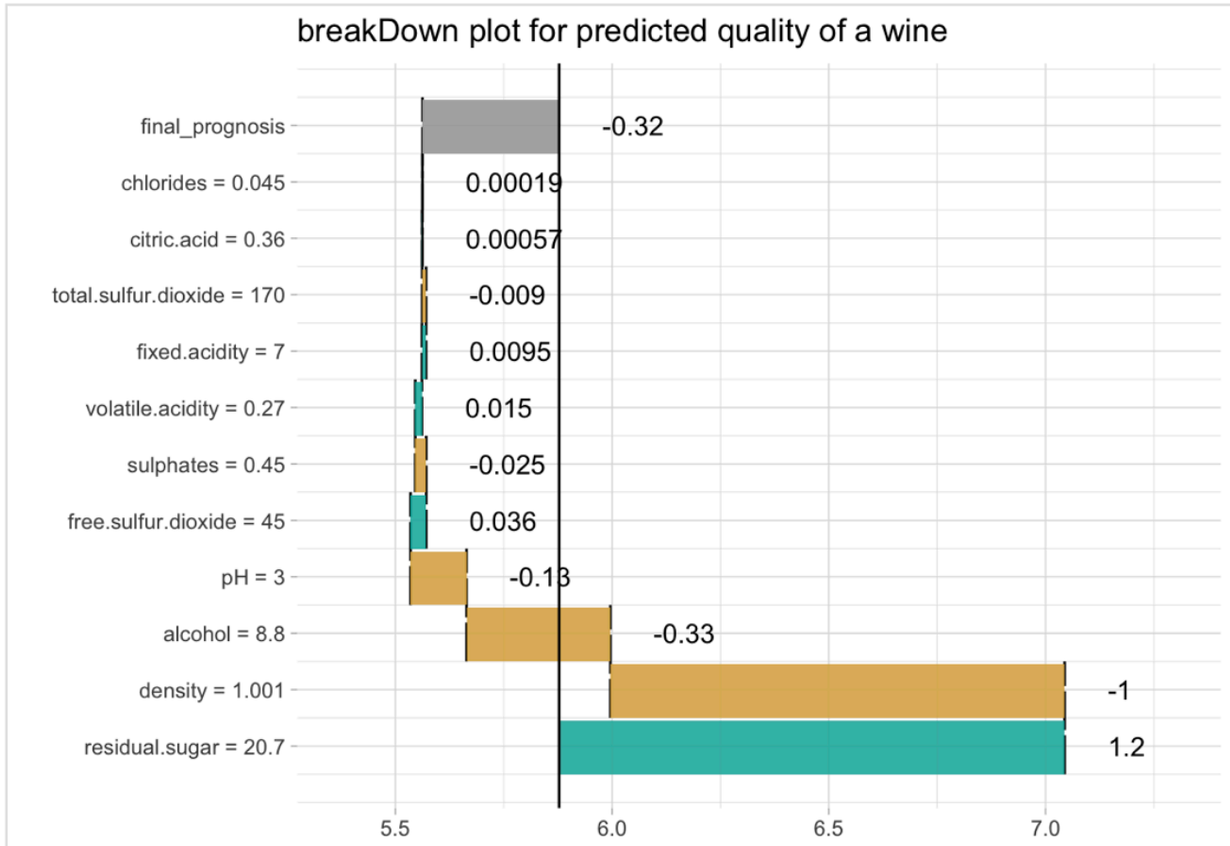
Linear model

Linear models are widely used in predictive modeling.
They have simple structure, which makes them easy to deploy or implement.
But models with many variables are hard to understand.

The **breakDown** plot explains the relation between variables and model prediction for a new observation.

Explanations are generated in three steps:
1) create model with **lm**() function
2) break down model predictions with the **broken()** function
3) plot the graphical summary with the generic **plot**() function.

```
library(breakDown)
library(ggplot2)
model <- lm(quality ~ ., data = wineQuality)
br <- broken(model, wineQuality[1,],
          baseline = "Intercept")
br
#>                                contribution
#> residual.sugar = 20.7               1.20000
#> density = 1.001                    -1.00000
#> alcohol = 8.8                      -0.33000
#> pH = 3                             -0.13000
#> free.sulfur.dioxide = 45            0.03600
#> sulphates = 0.45                   -0.02500
#> volatile.acidity = 0.27             0.01500
#> fixed.acidity = 7                   0.00950
#> total.sulfur.dioxide = 170         -0.00900
#> citric.acid = 0.36                  0.00057
#> chlorides = 0.045                   0.00019
#> final_prognosis                    -0.32000
#> baseline:  5.877909
plot(br)
```



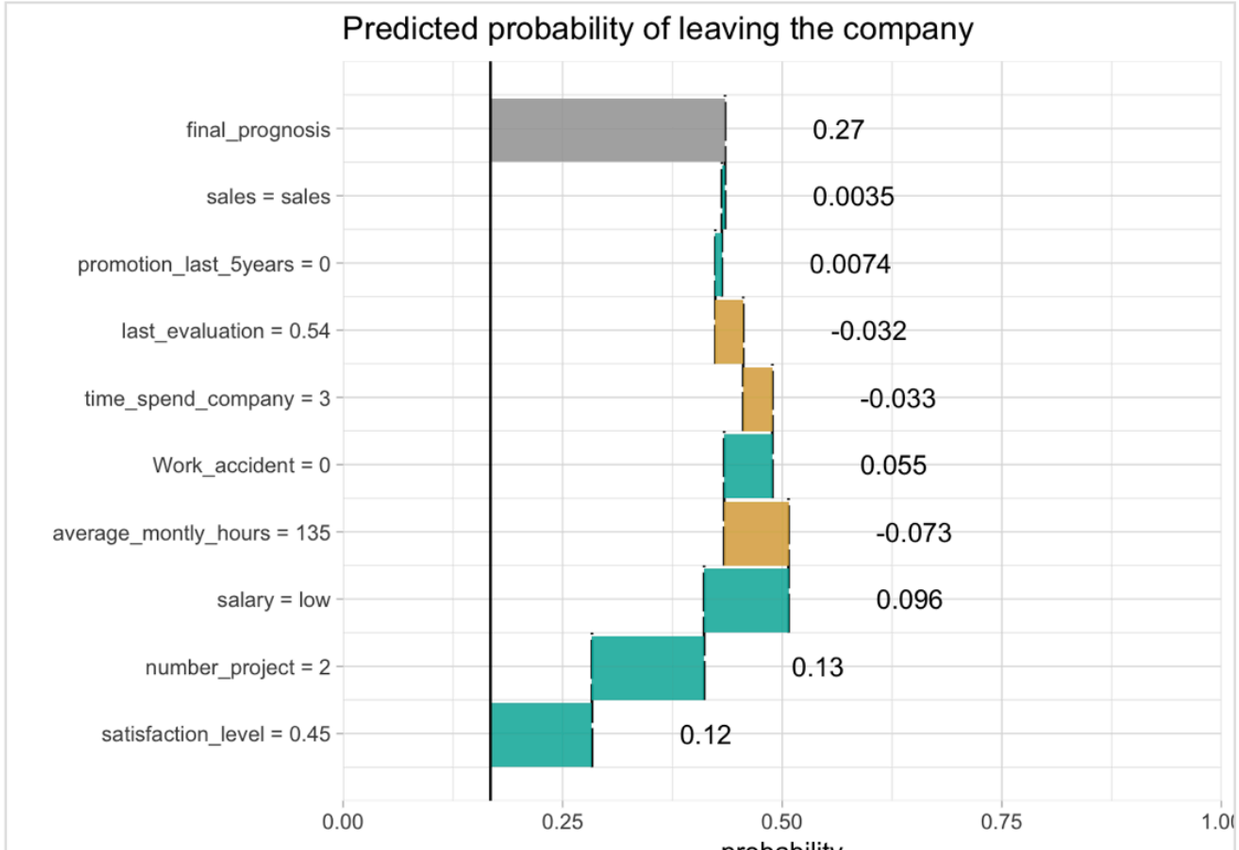
Logistic regression

breakDown plots may be also used to explain predictions from the logistic regression model.

On the OX axis one may present linear predictions (default) or use probit/logit transformation to present contributions of variables from the model. Use the trans= argument to define the transformation.

The baseline is presented by the vertical black line in the plot. One may set the baseline to 0 or to population average *(*use the *baseline = "intercept"* argument*)*.

```
library(breakDown)
library(ggplot2)
model <- glm(left~., data = HR_data,
                family = "binomial")
explain_1 <- broken(model, HR_data[11,],
                baseline = "intercept")
explain_1
#>                                contribution
#> satisfaction_level = 0.45             0.670
#> number_project = 2                    0.570
#> salary = low                          0.390
#> average_montly_hours = 135           -0.290
#> Work_accident = 0                     0.220
#> time_spend_company = 3               -0.130
#> last_evaluation = 0.54               -0.130
#> promotion_last_5years = 0             0.030
#> sales = sales                         0.014
#> final_prognosis                       1.300
#> baseline:  -1.601457
plot(explain_1, trans = function(x)
              exp(x)/(1+exp(x)))
```

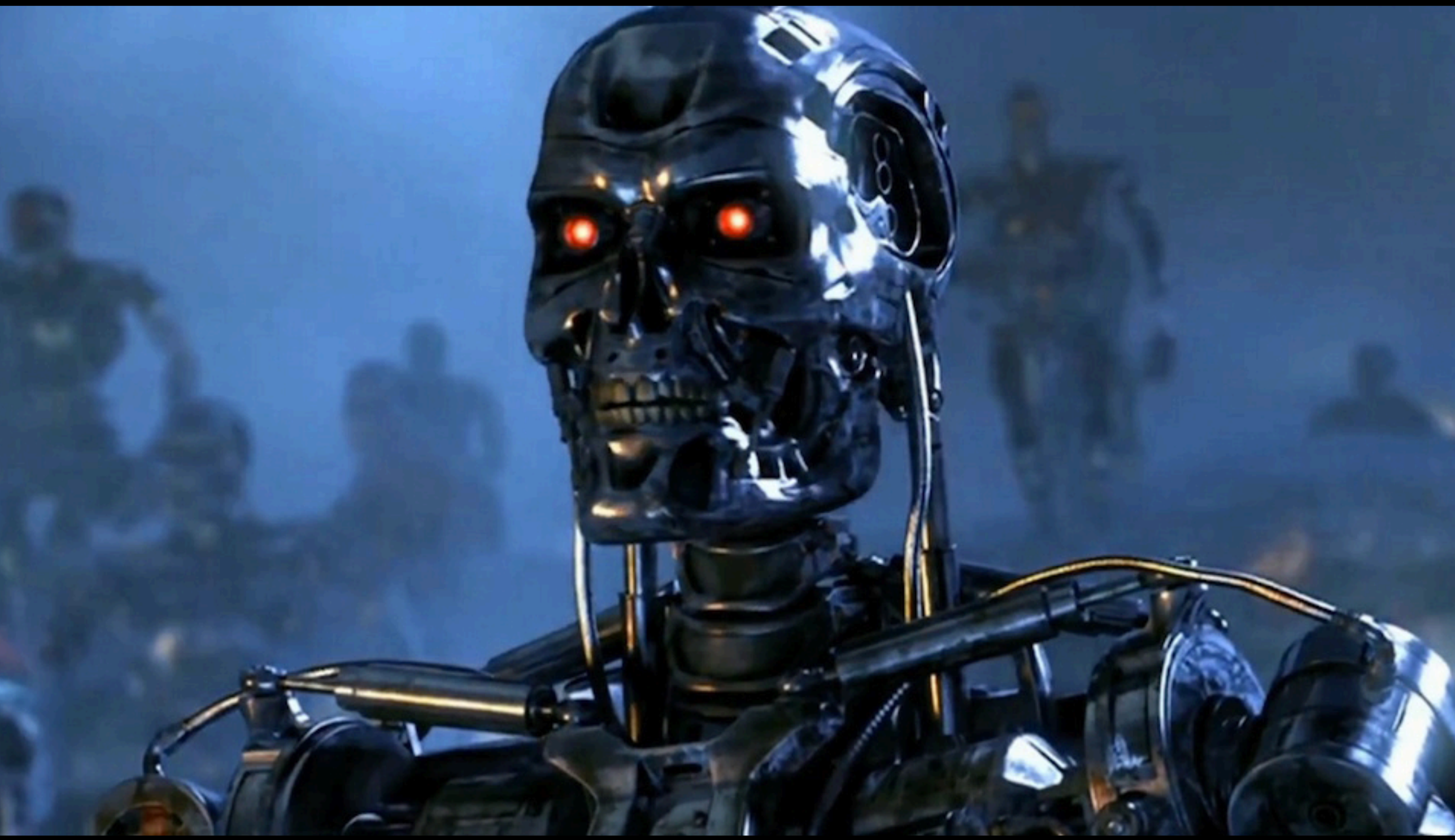


Model - Human interface

Hybrid AI + HI

How they will look like?

Machine Learning Models will replace humans



Machine Learning Models will empower humans