explain! explain! explain!

Przemysław Biecek

The Good, the Bad and the Ugly

SCIENTIFIC REPORTS

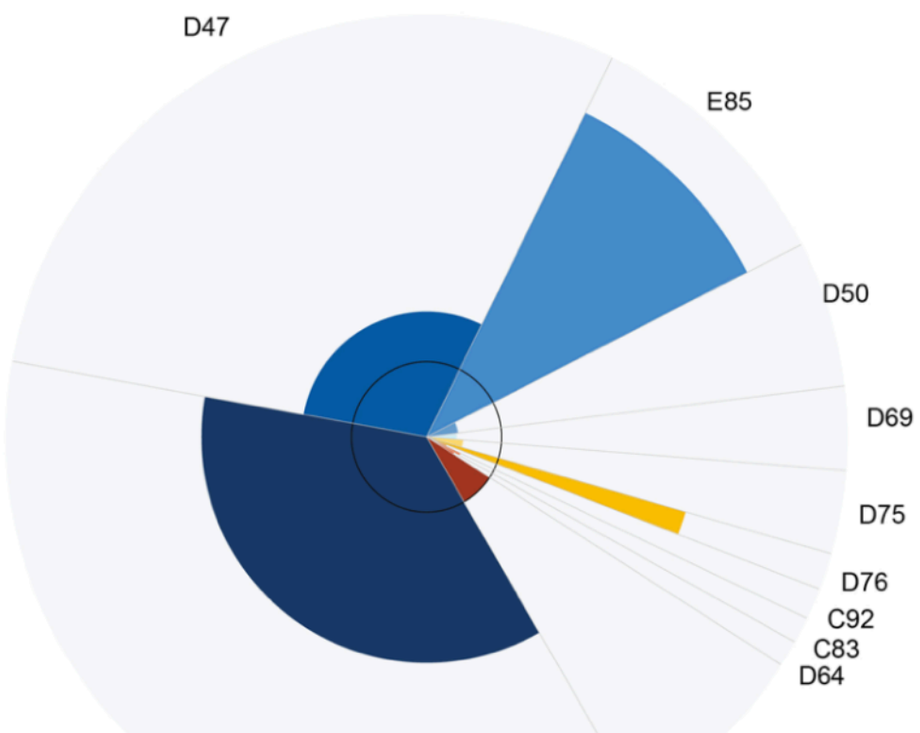
OPEN

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]

d: 2 August 2017
d: 14 December 2017
ed 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 d "fingerpr and indic clinical t specialis tests alo unprece



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

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 vascu-

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

Machine le
ety of classifi

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 Po
Greg S.

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

Traditiona
unning ex
e difficul
how that
rom 284,
isk factor
ears), ge
lood pres
he trained

gning an
can ofte
. Here, w
ed on dat
iovascula
within 3.2
), systoli
show tha
prediction

Risk str
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 vascu-

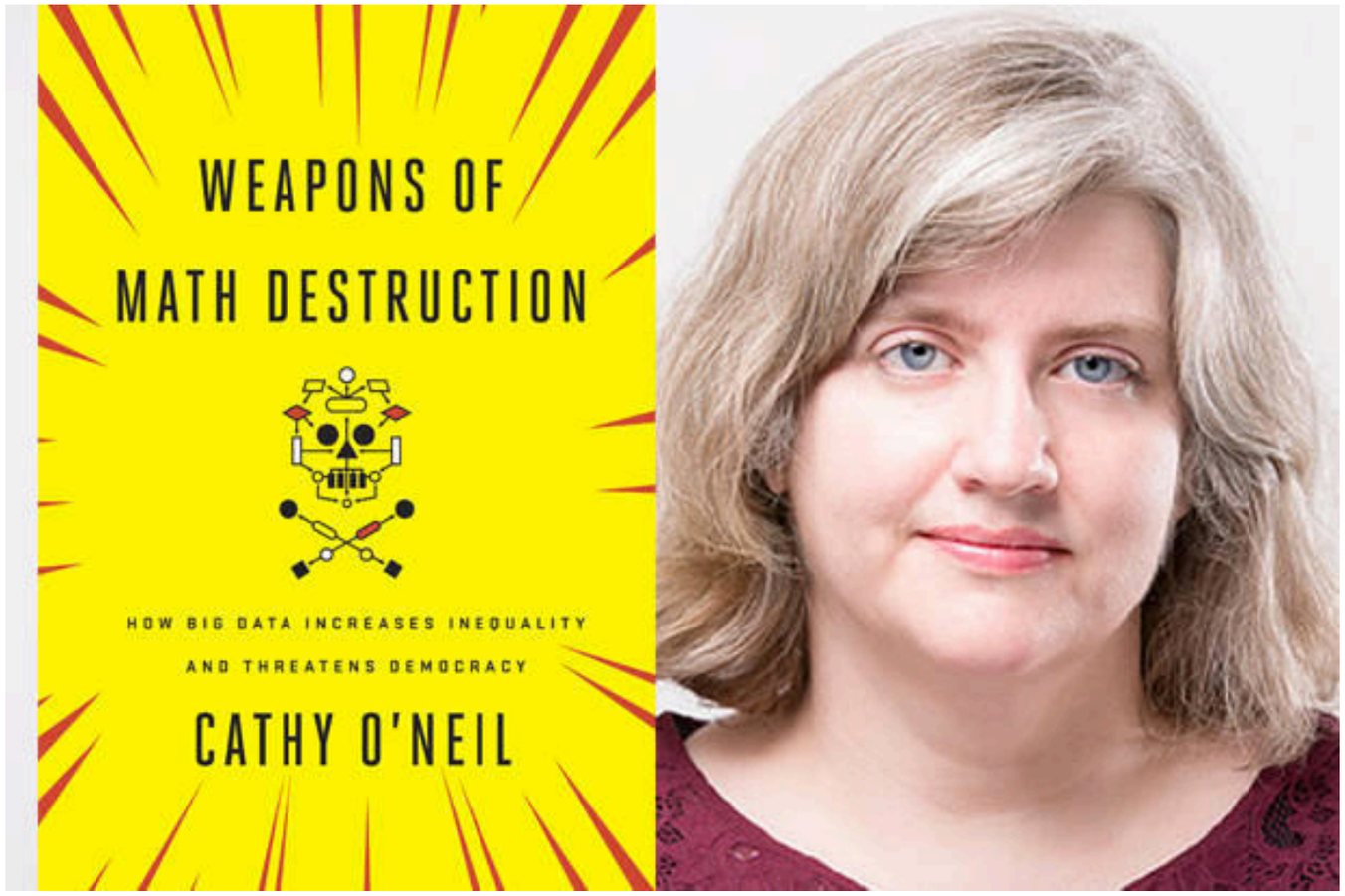
ect the sys
temic health of the cardiovascular system as well as future risk. Th
clinical utility of such features still requires further study. In thi
work, we demonstrate the extraction and quantification of multipl
cardiovascular risk factors from retinal images using deep learning

Machine learning has been leveraged for many years for a vari
ety of classification tasks, including the automated classificatio

Why do we need explanations for complex models?

Cathy O'Neil:

The era of blind faith in ~~big data~~ machine learning must end



- "You don't see a lot of skepticism," she says. "The algorithms are like shiny new toys that we can't resist using. We trust them so much that we project meaning on to them."
- Ultimately algorithms, according to O'Neil, reinforce discrimination and widen inequality, "using people's fear and trust of mathematics to prevent them from asking questions".

https://www.theguardian.com/books/2016/oct/27/cathy-oneil-weapons-of-math-destruction-algorithms-big-data

Why do we need explanations for complex models?

Article Talk Read Edit View history Search Wikipedia

Right to explanation

From Wikipedia, the free encyclopedia

In the regulation of algorithms, particularly artificial intelligence and its subfield of machine learning, a **right to explanation** (or **right to *an* explanation**) is a right to be given an explanation for an output of the algorithm. Such rights primarily refer to individual rights to be given an explanation for decisions that significantly affect an individual, particularly legally or financially. For example, a person who applies for a loan and is denied may ask for an explanation, which could be "Credit bureau X reports that you declared bankruptcy last year; this is the main factor in considering you too likely to default, and thus we will not give you the loan you applied for."

Some such legal rights already exist, while the scope of a general "right to explanation" is a matter of ongoing debate.

Contents [hide]

Why do we need explanations for complex models?

European Union [edit]

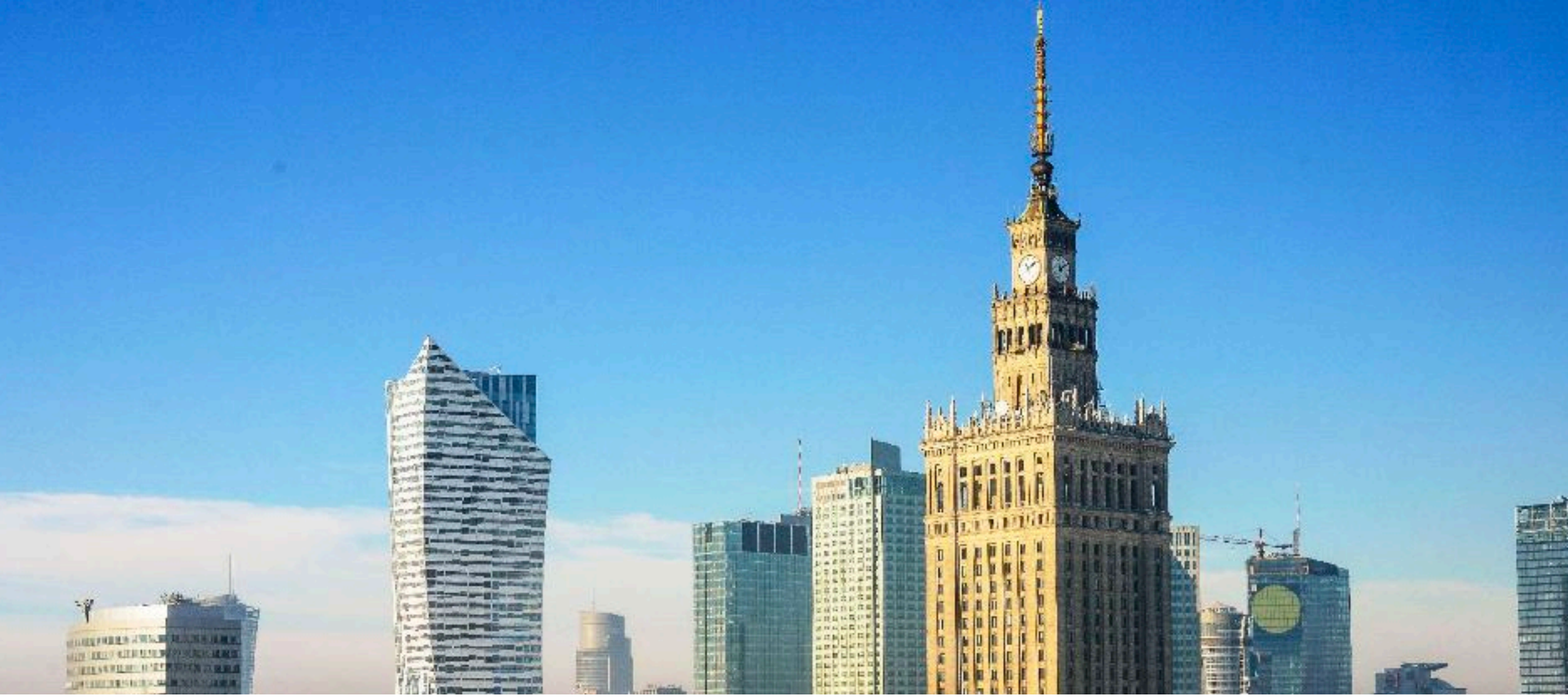
The European Union General Data Protection Regulation (enacted 2016, taking effect 2018), extends the automated decision-making rights in the 1995 Data Protection Directive to provide a legally disputed form of a right to an explanation, stated as such in Recital 71: "[the data subject should have] the right ... to obtain an explanation of the decision reached". In full:

> The data subject should have the right not to be subject to a decision, which may include a measure, evaluating personal aspects relating to him or her which is based solely on automated processing and which produces legal effects concerning him or her or similarly significantly affects him or her, such as automatic refusal of an online credit application or e-recruiting practices without any human intervention.
>
> ...
>
> In any case, such processing should be subject to suitable safeguards, which should include specific information to the data subject and the right to obtain human intervention, to express his or her point of view, to obtain an explanation of the decision reached after such assessment and to challenge the decision.

However, the extent to which the regulations themselves provide a "right to explanation" is heavily debated.[4][5] There are two main strands of criticism. There are significant legal issues with the right as found in Article 22 — as recitals are not binding, and the right to an explanation is not mentioned in the binding articles of the text, having been removed during the legislative process.[6] In addition, there are significant restrictions on the types of automated decisions that are covered — which must be both "solely" based on automated processing, and have legal or similarly significant effect — which limits their applicability in many of the cases of algorithmic controversy that have been picked up in the media.[7]

Single prediction explainers

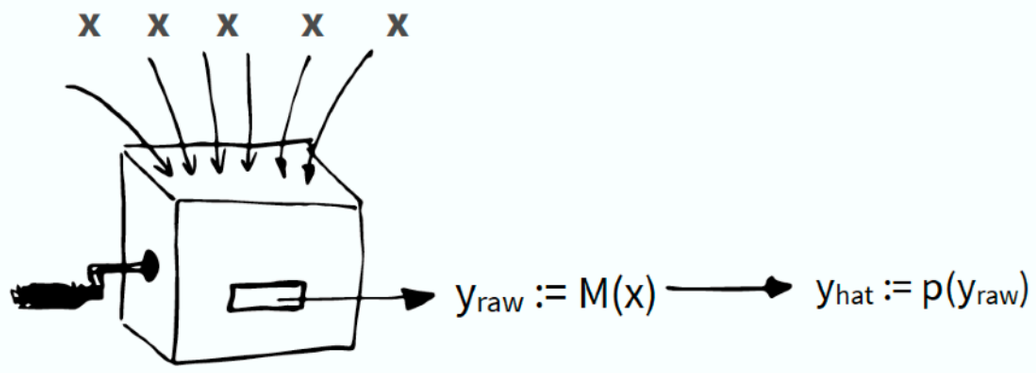
http://www.warszawa.pl/biznes/panorama-warszawy/galeria/dzis_palac_kultury_i_nauki/



```
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)



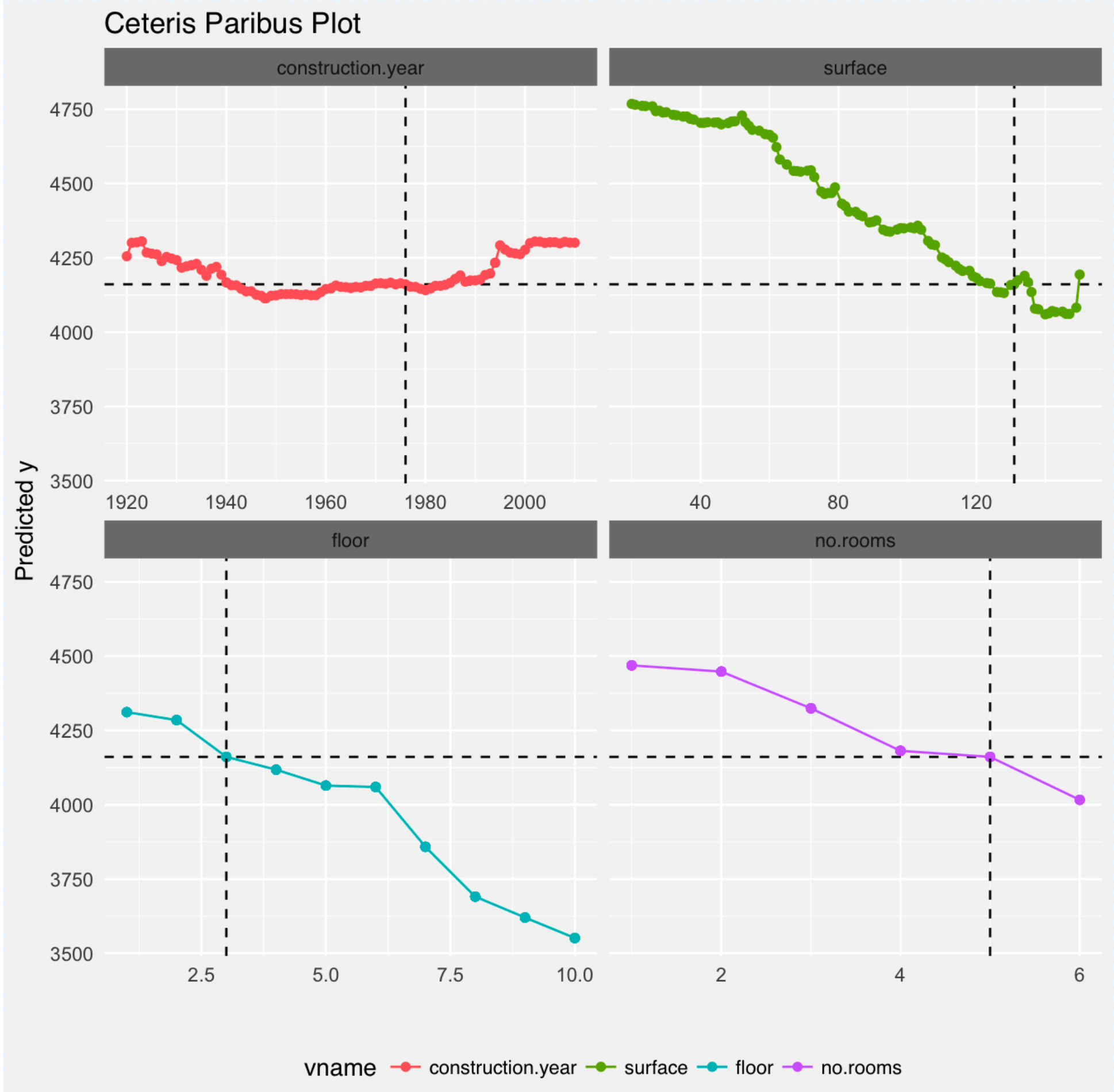
B)

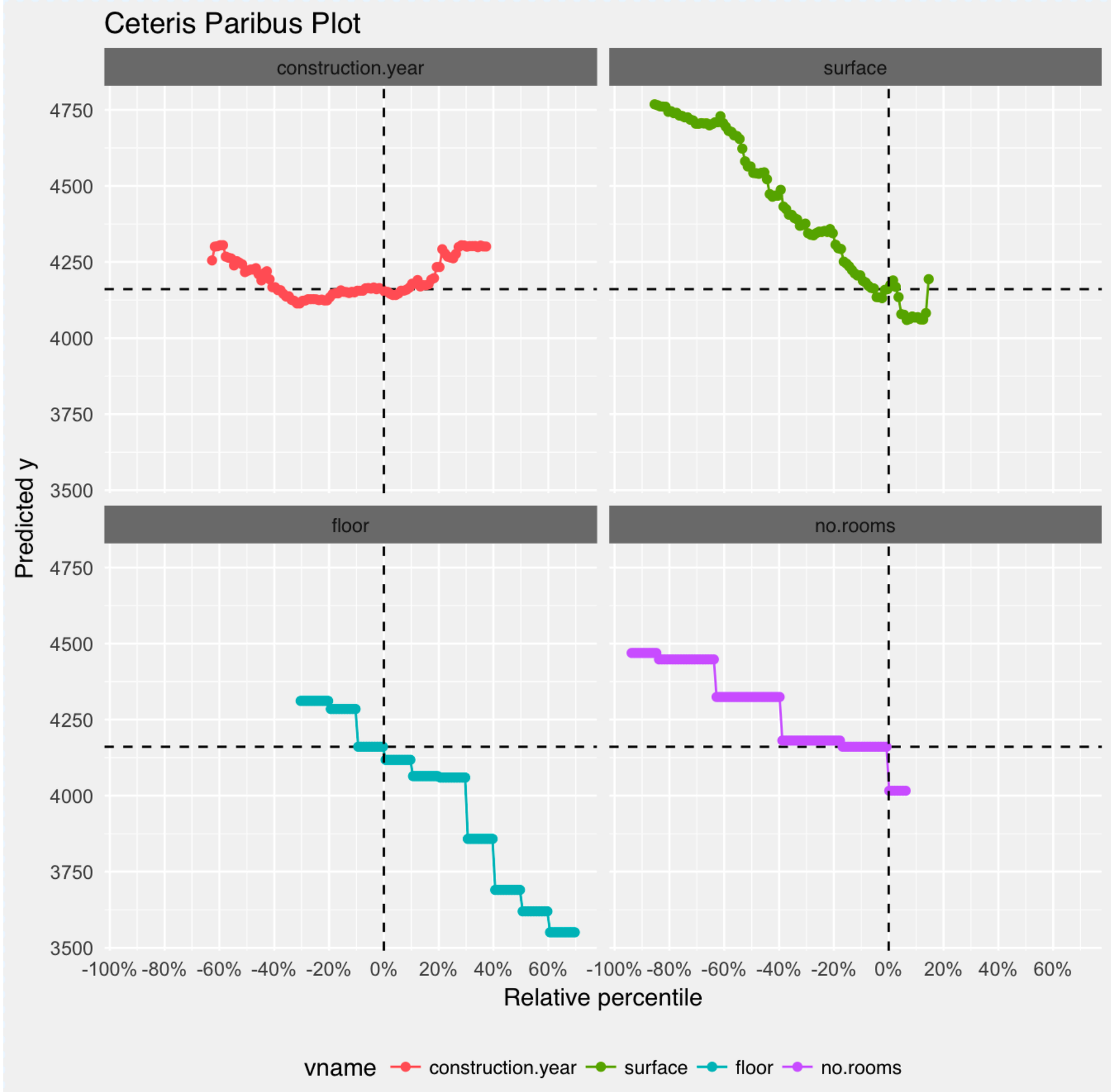
explain(model; data; y; predict_function; trans)

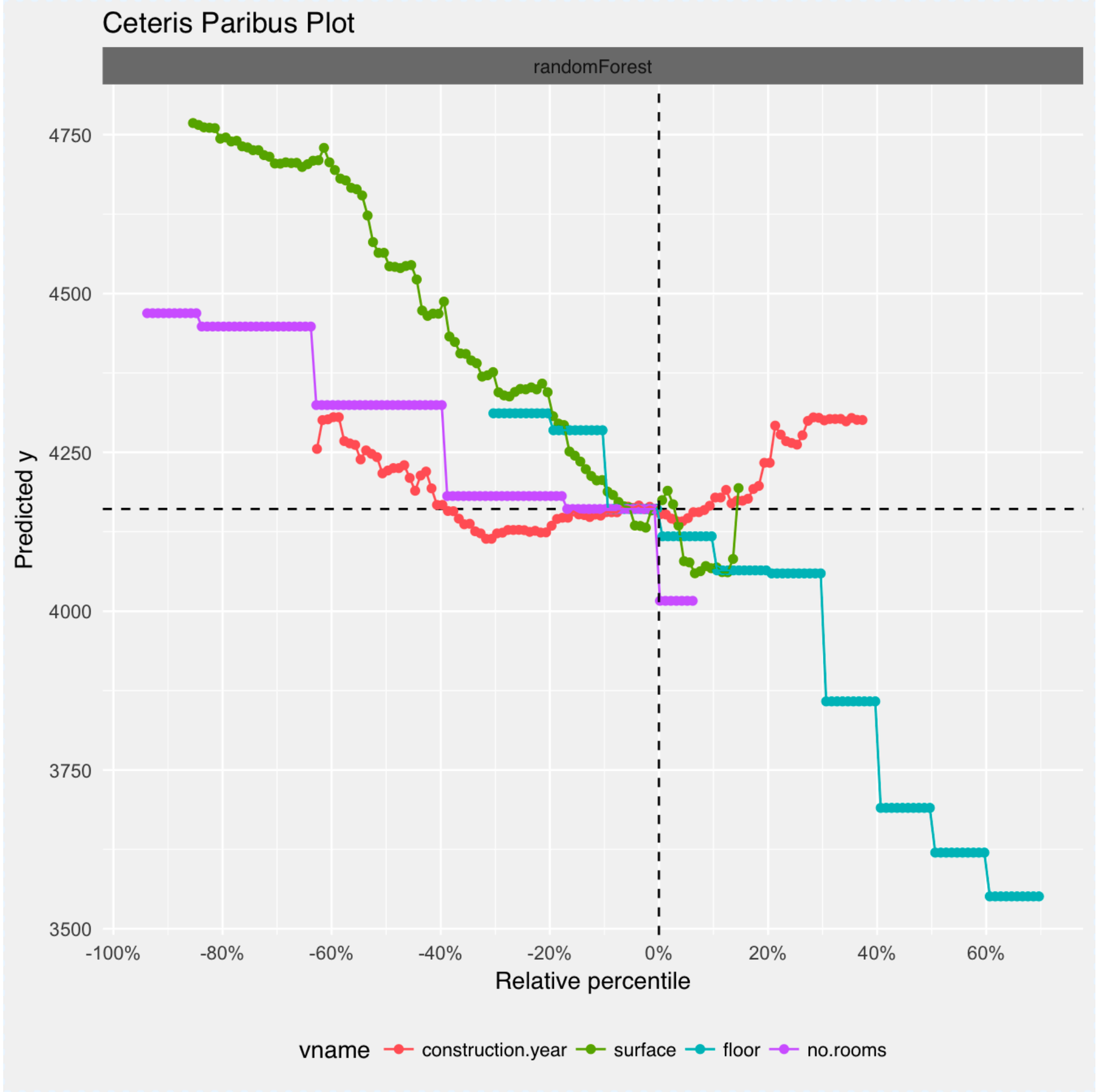
```
library("DALEX")
apartments_lm_model <- lm(m2.price ~ construction.year + surface + floor +
                          no.rooms + district, data = apartments)
library("randomForest")
set.seed(59)
apartments_rf_model <- randomForest(m2.price ~ construction.year + surface + floor
                  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

Case:
large (130 m2) flat, 5 rooms, 3 rd floor
from 1970

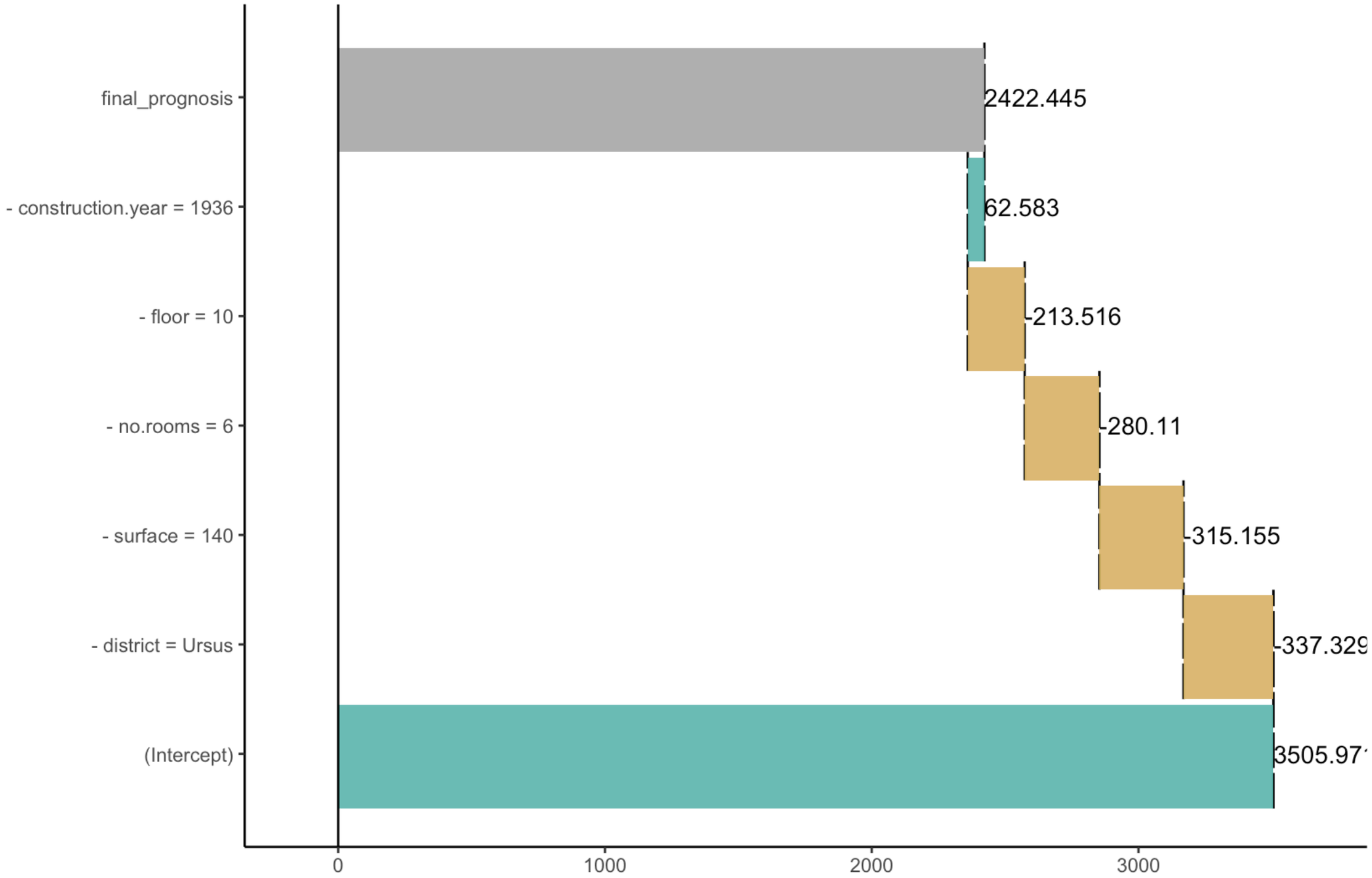
Prediction:
4200 E/m2


Ceteris Paribus Plot
construction.year
surface
floor
no.rooms
predicted y
4750
4500
4250
4000
3750
3500
1920
1940
1960
1980
2000
40
80
120
2.5
5.0
7.5
10.0
2
4
6
vname
construction.year
surface
floor
no.rooms

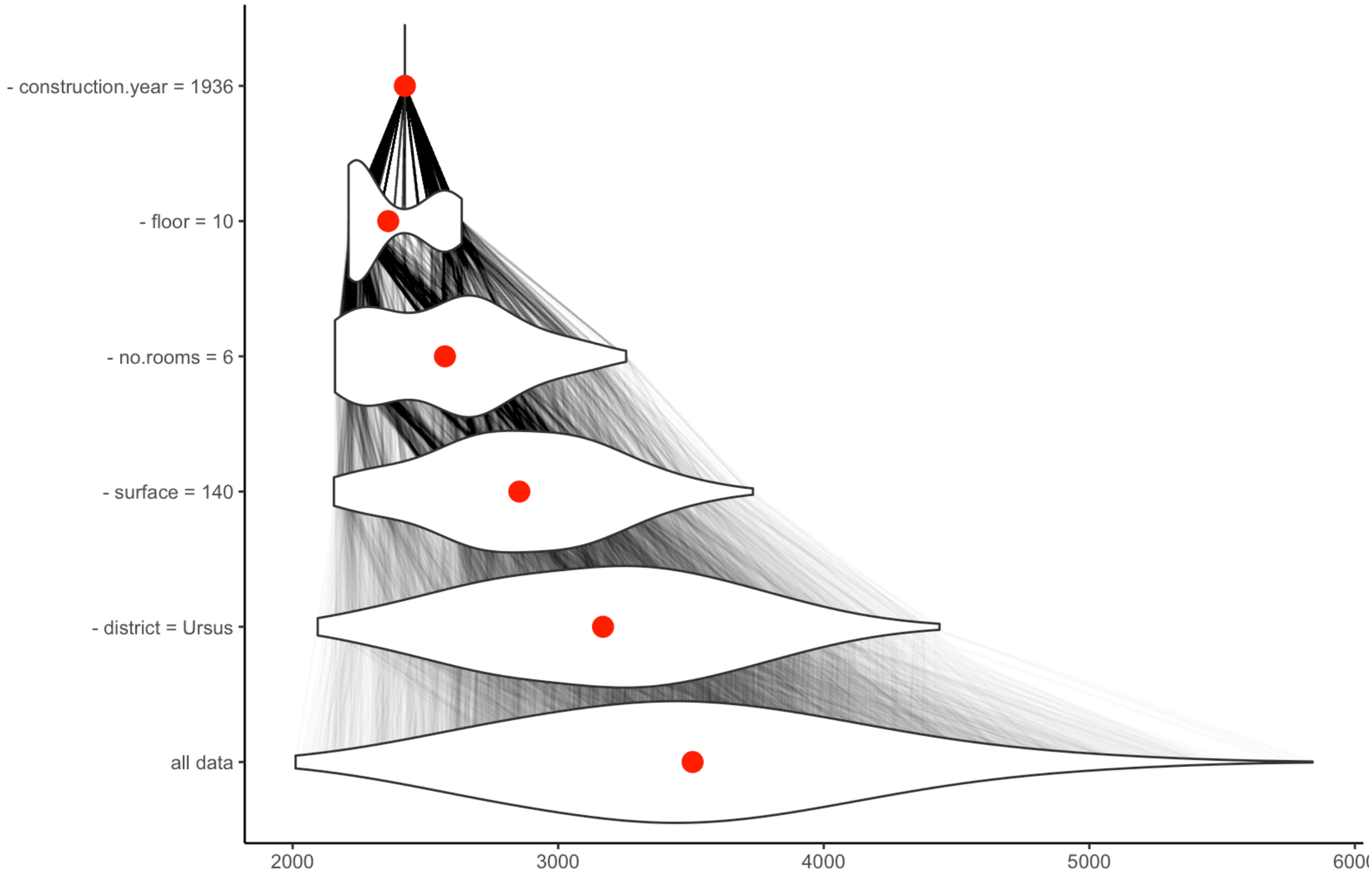

Ceteris Paribus Plot
construction.year
surface
floor
no.rooms
Predicted y
4750
4500
4250
4000
3750
3500
-100% -80% -60% -40% -20% 0% 20% 40% 60%
Relative percentile
vname
construction.year
surface
floor
no.rooms


Ceteris Paribus Plot
randomForest
4750
4500
4250
4000
3750
3500
Predicted y
-100%
-80%
-60%
-40%
-20%
0%
20%
40%
60%
Relative percentile
vname
construction.year
surface
floor
no.rooms

Break Down Plots



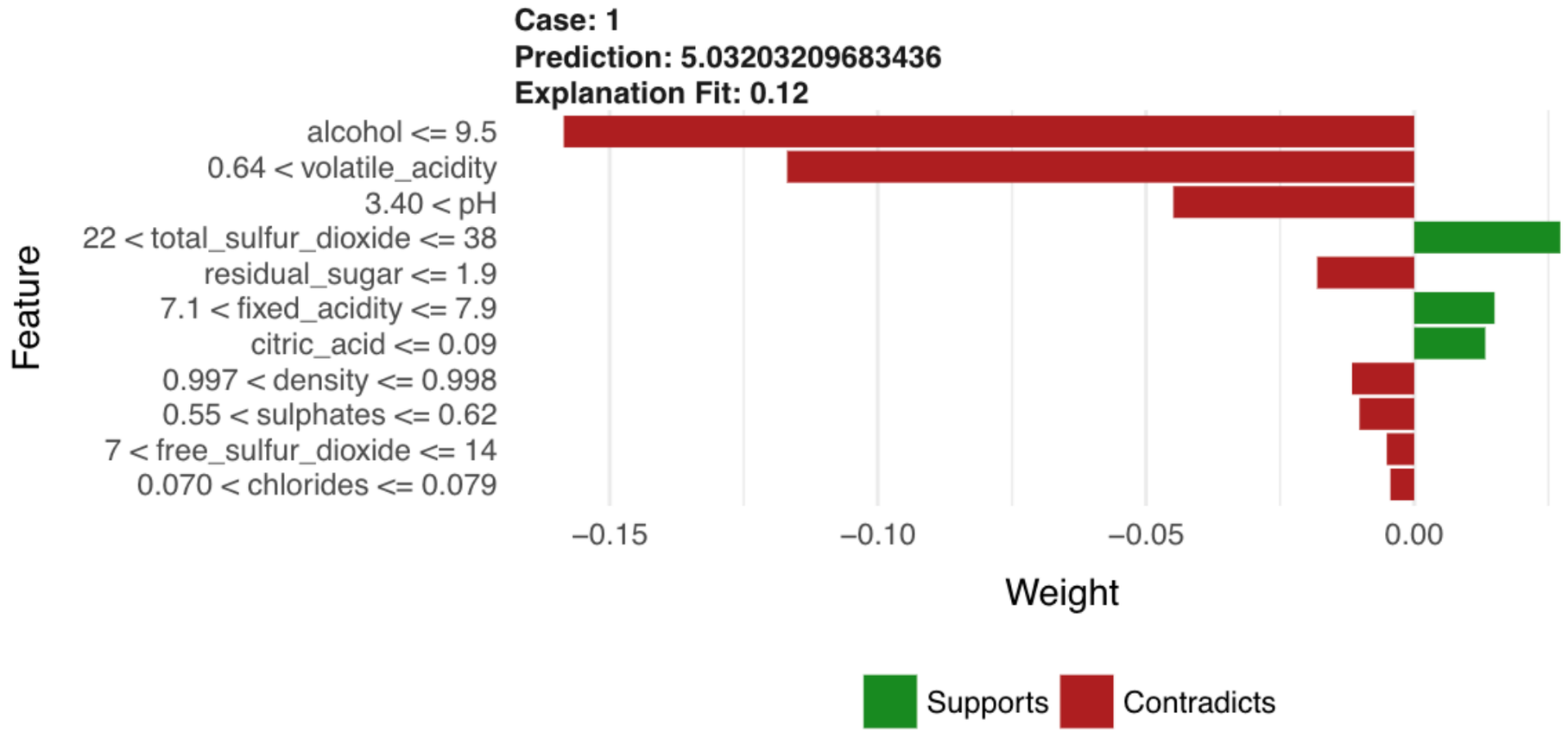
Break Down Plots

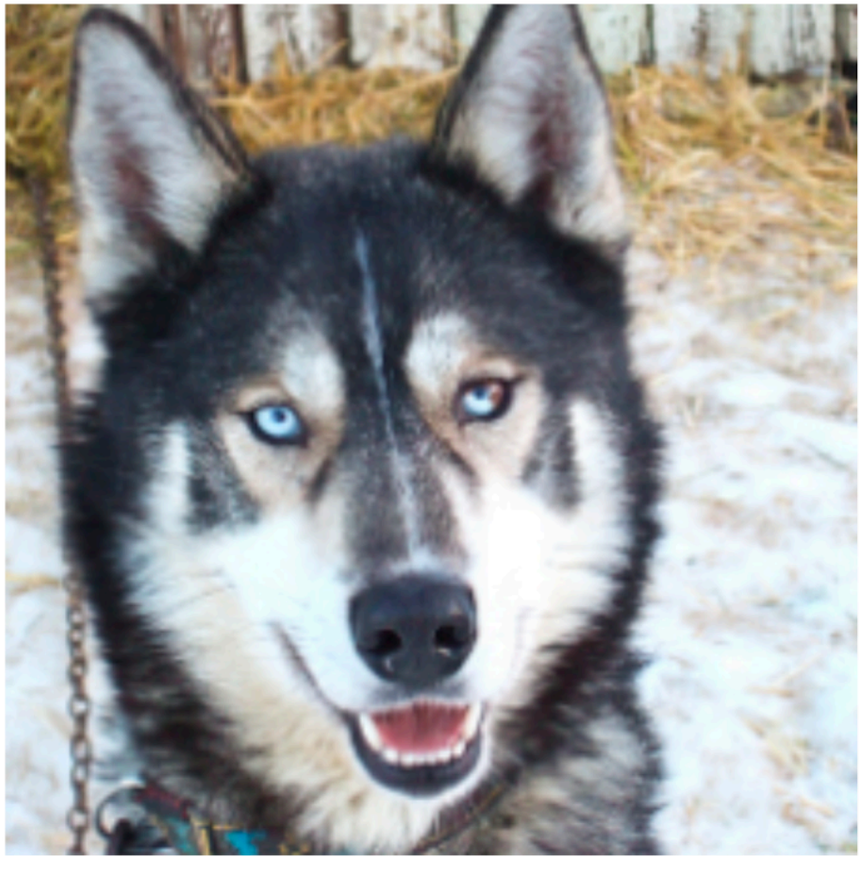


LIME

VS

breakDown





(a) Husky classified as wolf

(b) Explanation

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 forrandom 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.

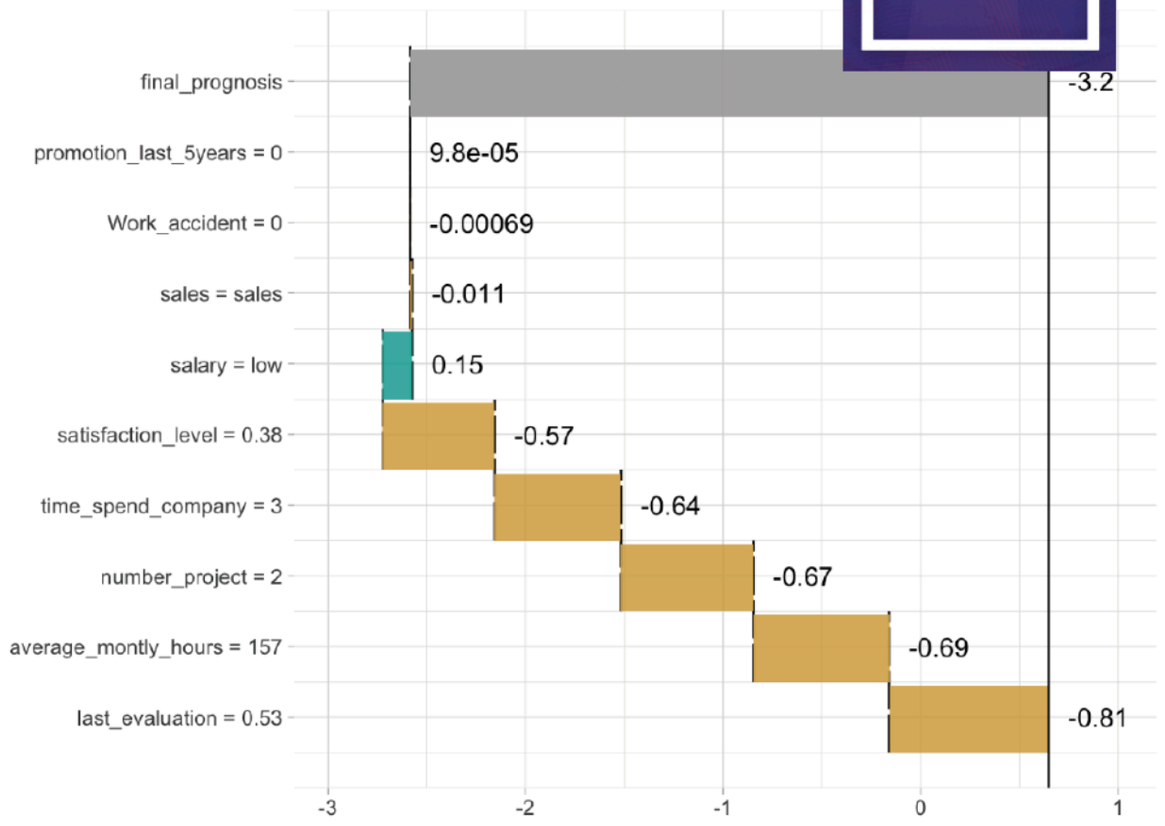
First, let's create a black-box model for prediction. Here we are using randomForest

```
library(live)
library(randomForest)

head(HR_data, 2)
#>  satisfaction_level last_evaluation
#> 1               0.38            0.53
#>  number_project average_montly_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
```

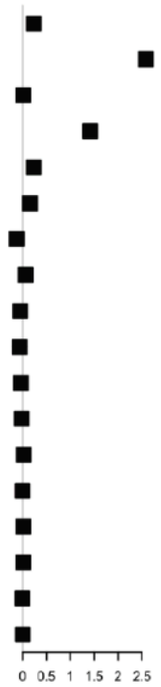
Then 1) create the adjacency (function **sample_locally**), 2) fit the white box (function **fit_explanations**) and 3) plot the white box.

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,])
```



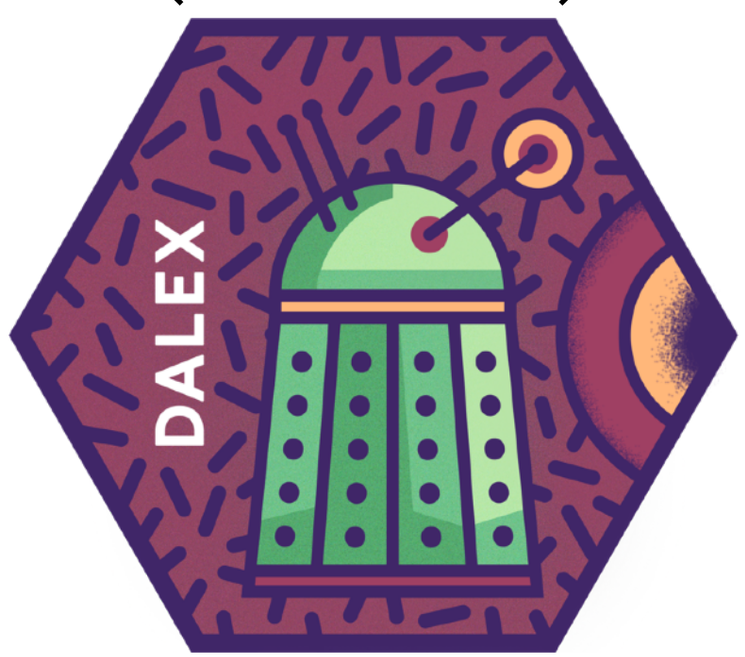
Variable	Observed	Estimate	Lower	Upper
time_spend_company	3	0.23	0.23	0.24
last_evaluation	0.53	2.58	2.5	2.66
average_montly_hours	157	0.01	0.01	0.01
satisfaction_level	0.38	1.41	1.35	1.48
number_project	2	0.23	0.22	0.24
salarylow		0.15	0.08	0.22
salessales		-0.13	-0.21	-0.05
Work_accident	0	0.06	0.01	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	0.09



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

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

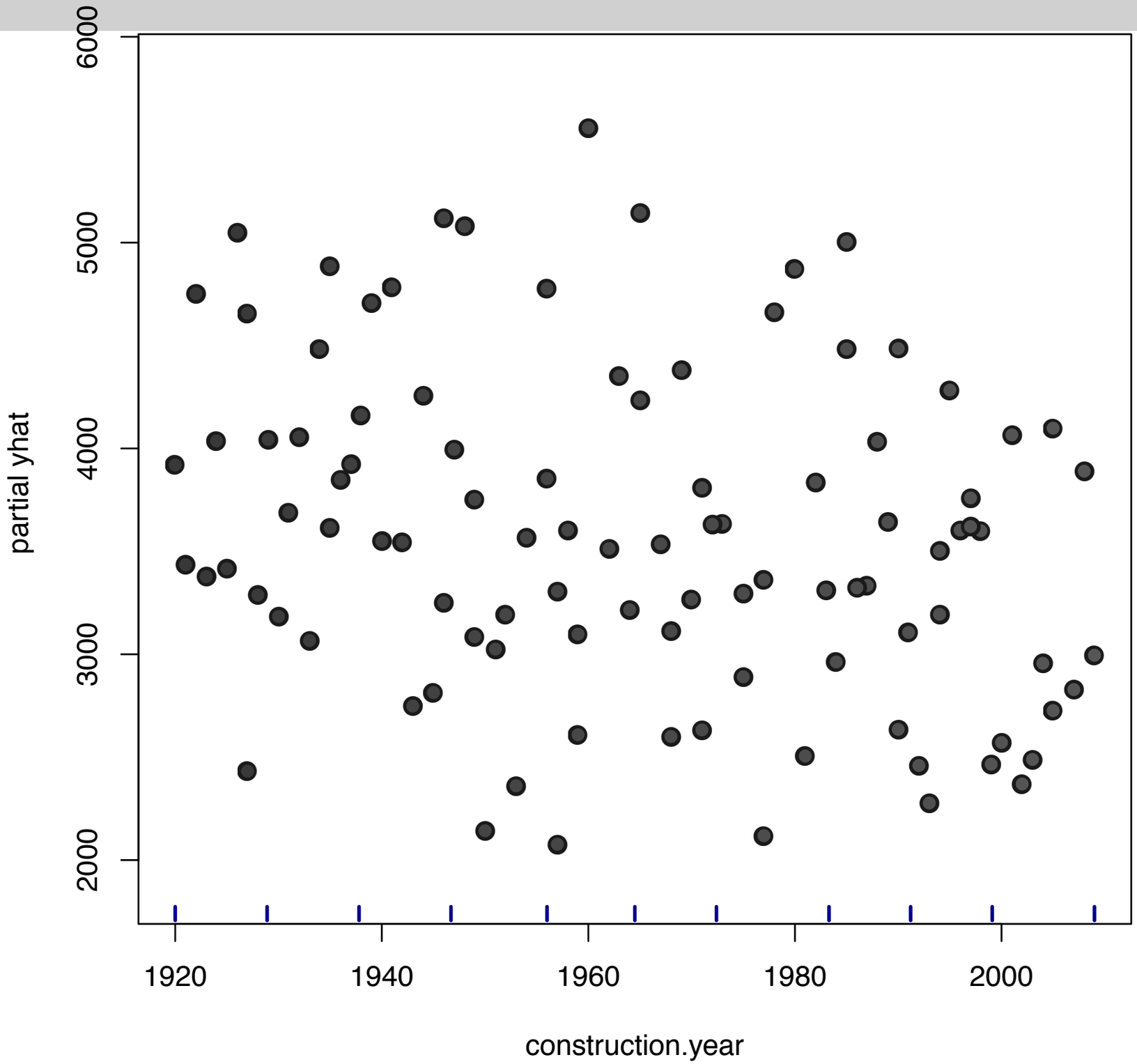
Model Predictions Explainers
package: **breakDown**, **live**, shapleyr, lime

A Tale of Two Models

Accuracy as a single number is not enough!

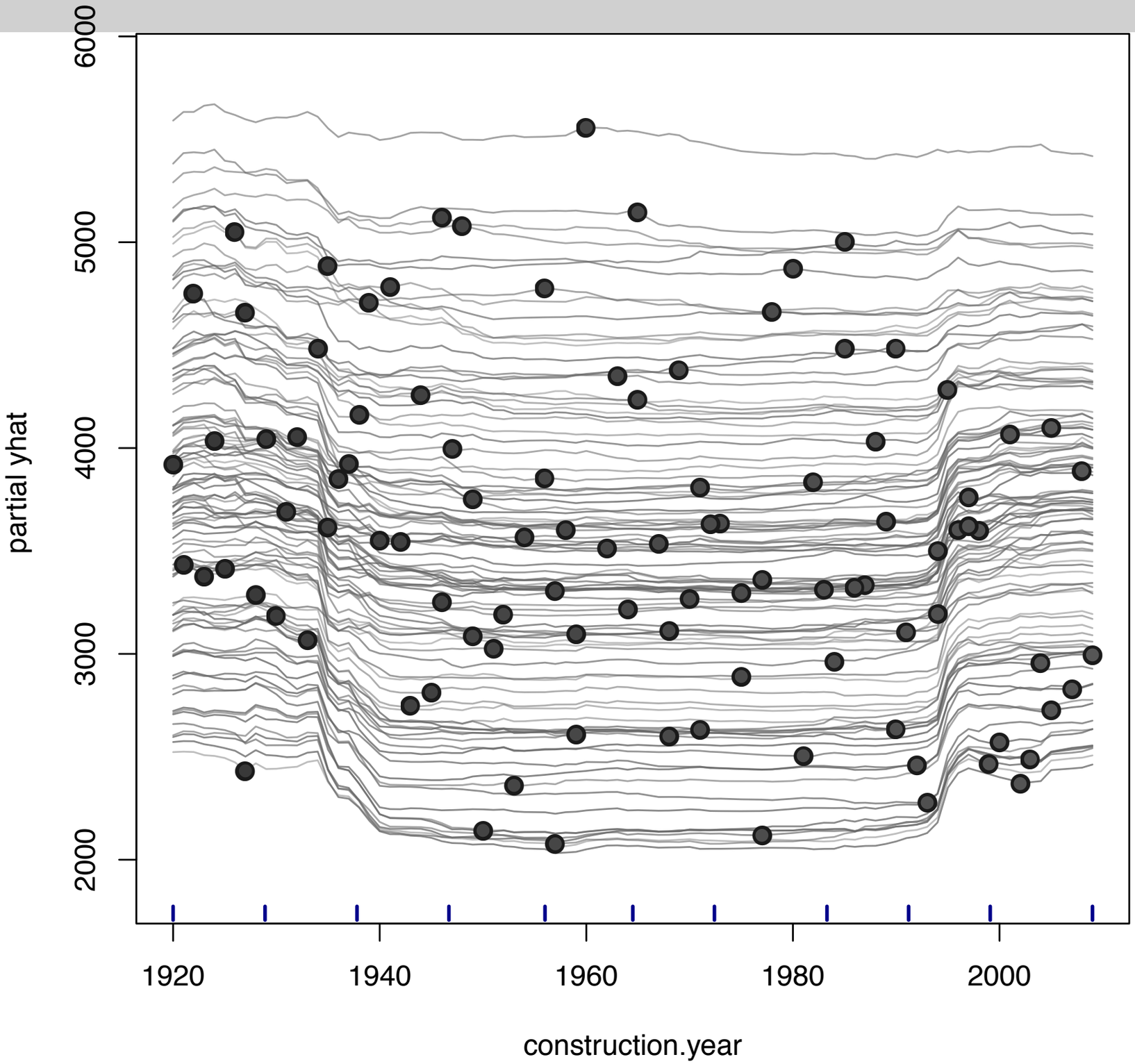
```
# root mean square
predicted_mi2_lm <- predict(apartments_lm_model, apartmentsTest)
sqrt(mean((predicted_mi2_lm - apartmentsTest$m2.price)^2))
## [1] 283.0865

# root mean square
predicted_mi2_rf <- predict(apartments_rf_model, apartmentsTest)
sqrt(mean((predicted_mi2_rf - apartmentsTest$m2.price)^2))
## [1] 283.3479
```

How a single variable affects the response?



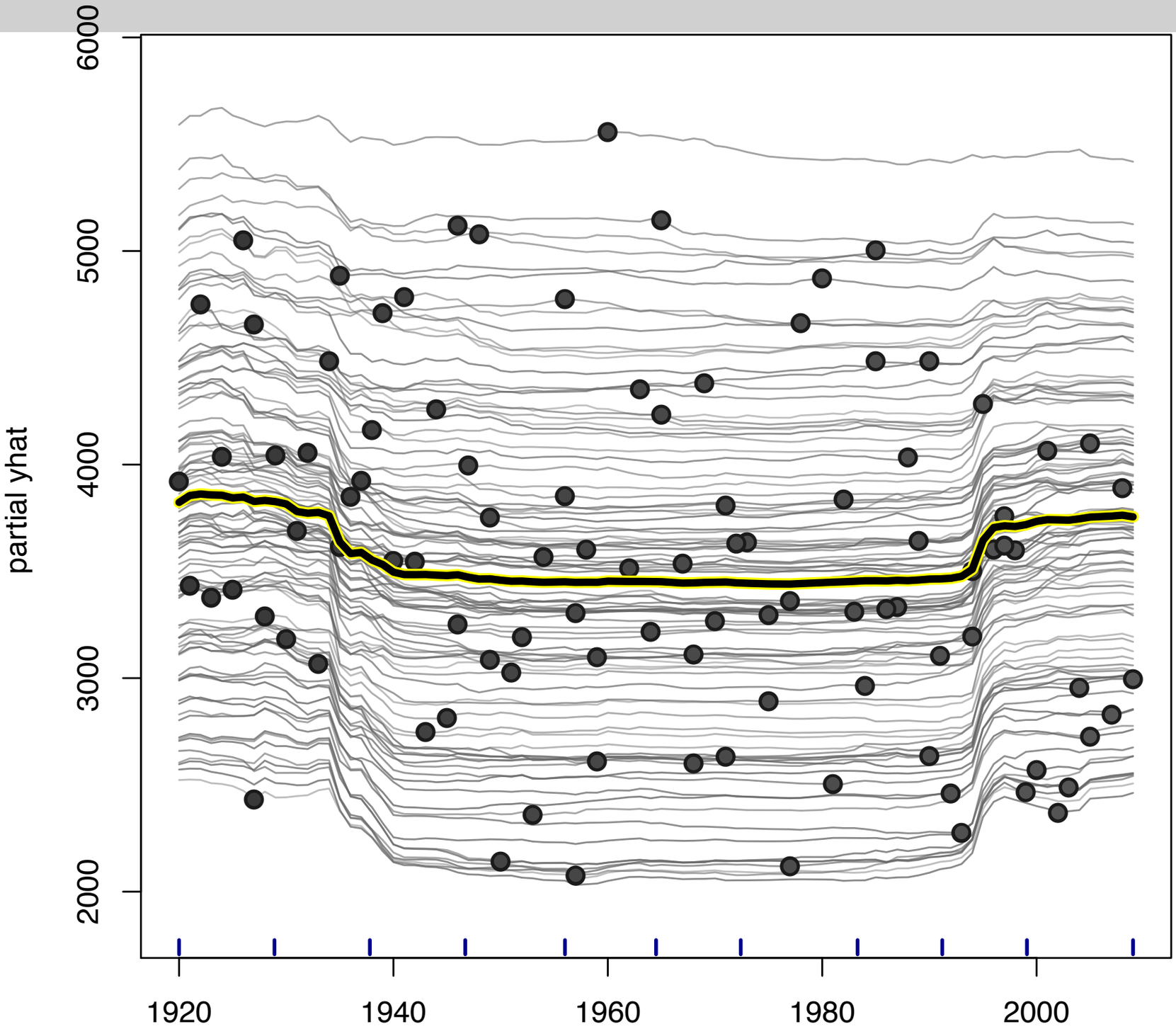
Predictions
vs
Single variable

How a single variable affects the response?



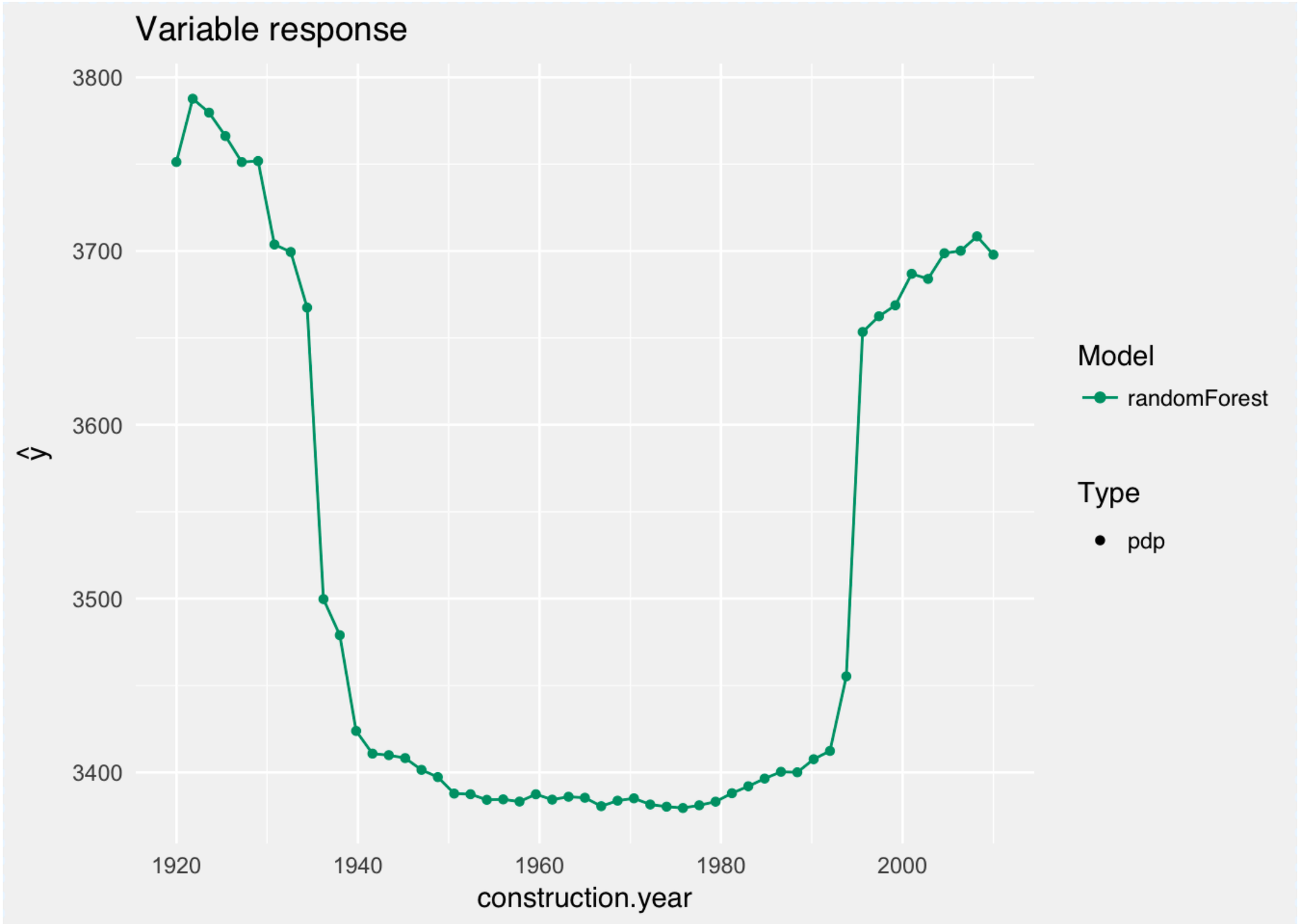
Individual Conditional Expectations

How a single variable affects the response?

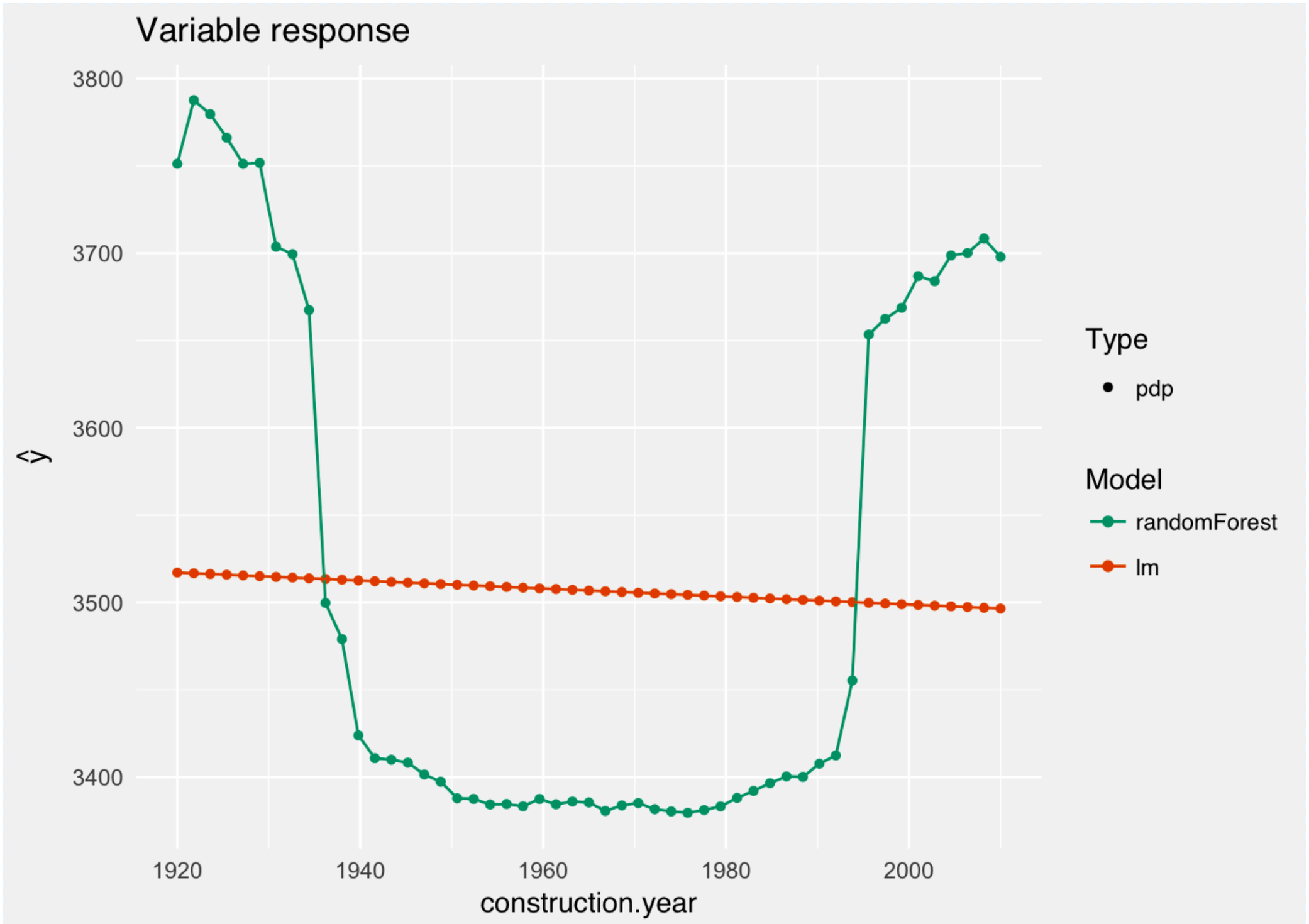


Partial Dependence Plot

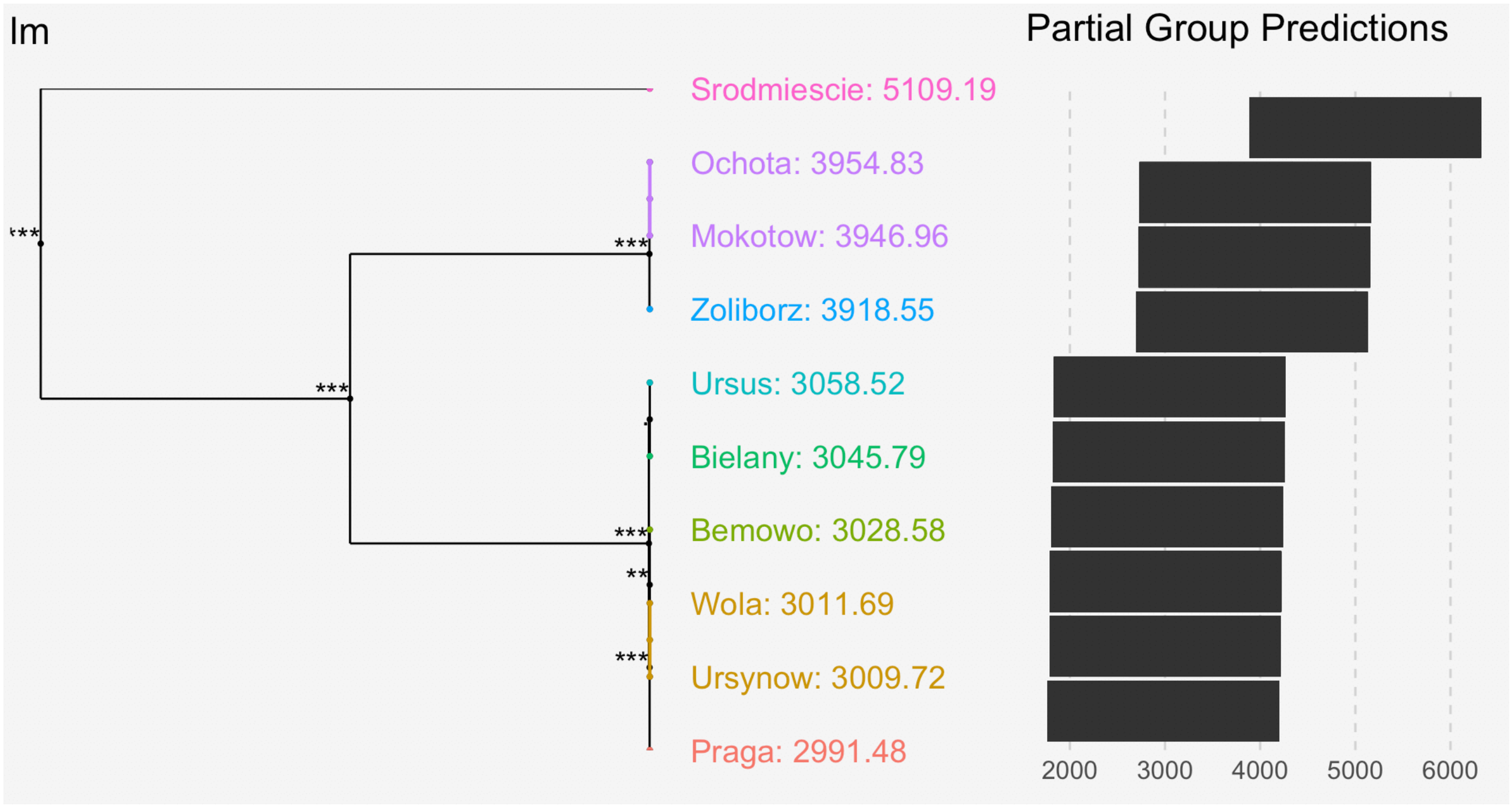
```r
sv_rf  <- single_variable(explainer_rf, variable =  "construction.year", type = "pdp")
plot(sv_rf)
```



```r
sv_lm  <- single_variable(explainer_lm, variable =  "construction.year", type = "pdp")

plot(sv_rf, sv_lm)
```



```r
svd_lm  <- single_variable(explainer_lm, variable = "district", type = "factor")

plot(svd_lm)
```



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

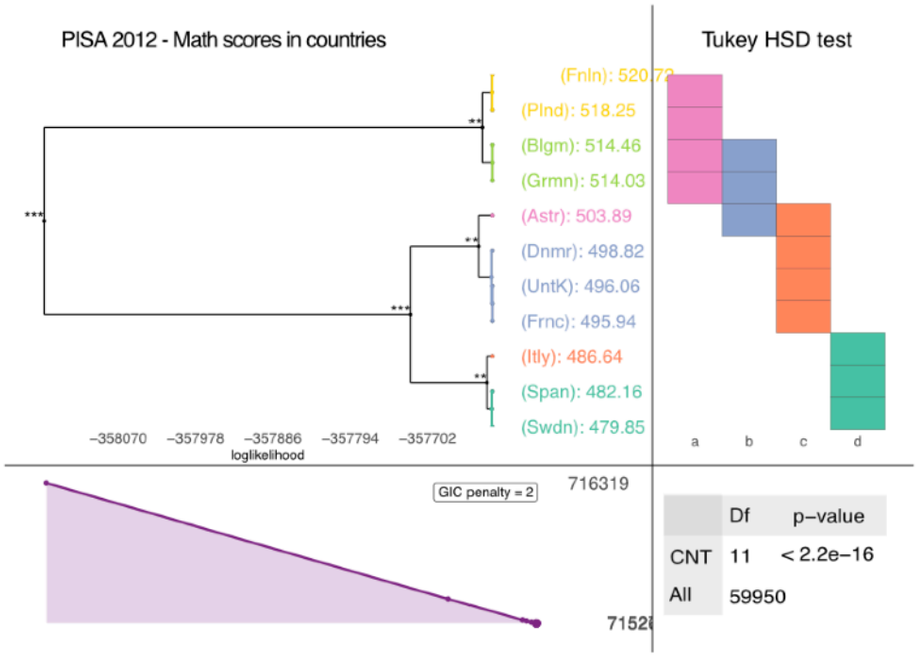
```
library(factorMerger)

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

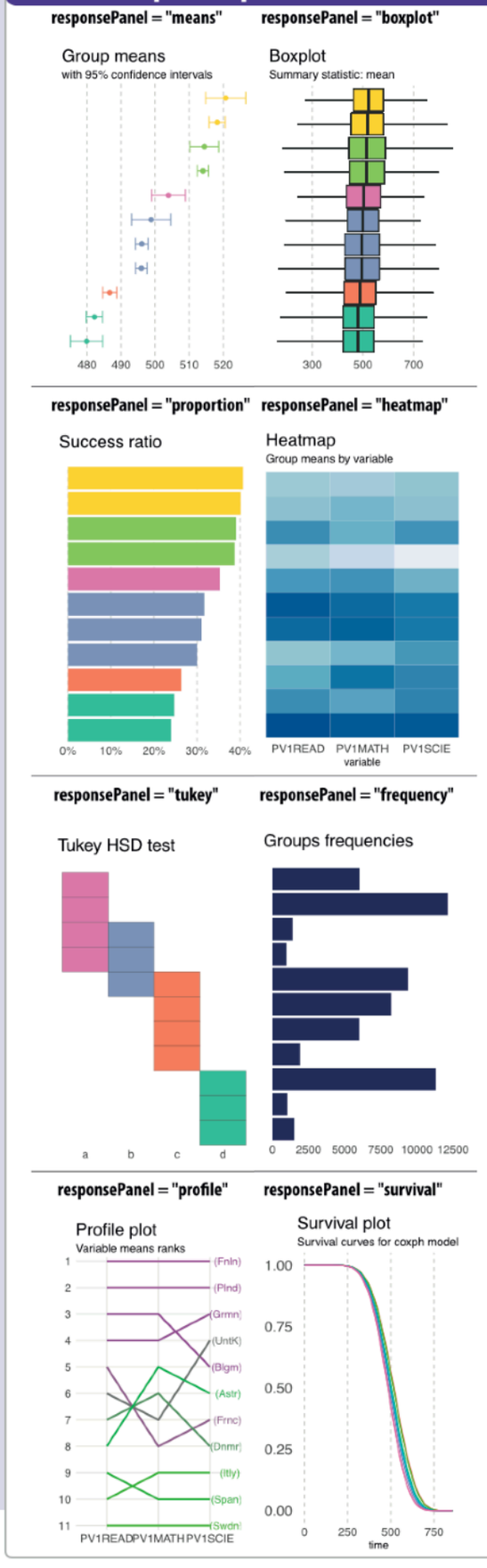
The **method** argument determines how pairs of groups are considered for grouping. The most computational demanding is *adaptive*. For larger number of groups one may consider only pairs of groups with closest averages or use the *fast-fixed* heuristic (fastest solution).

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.

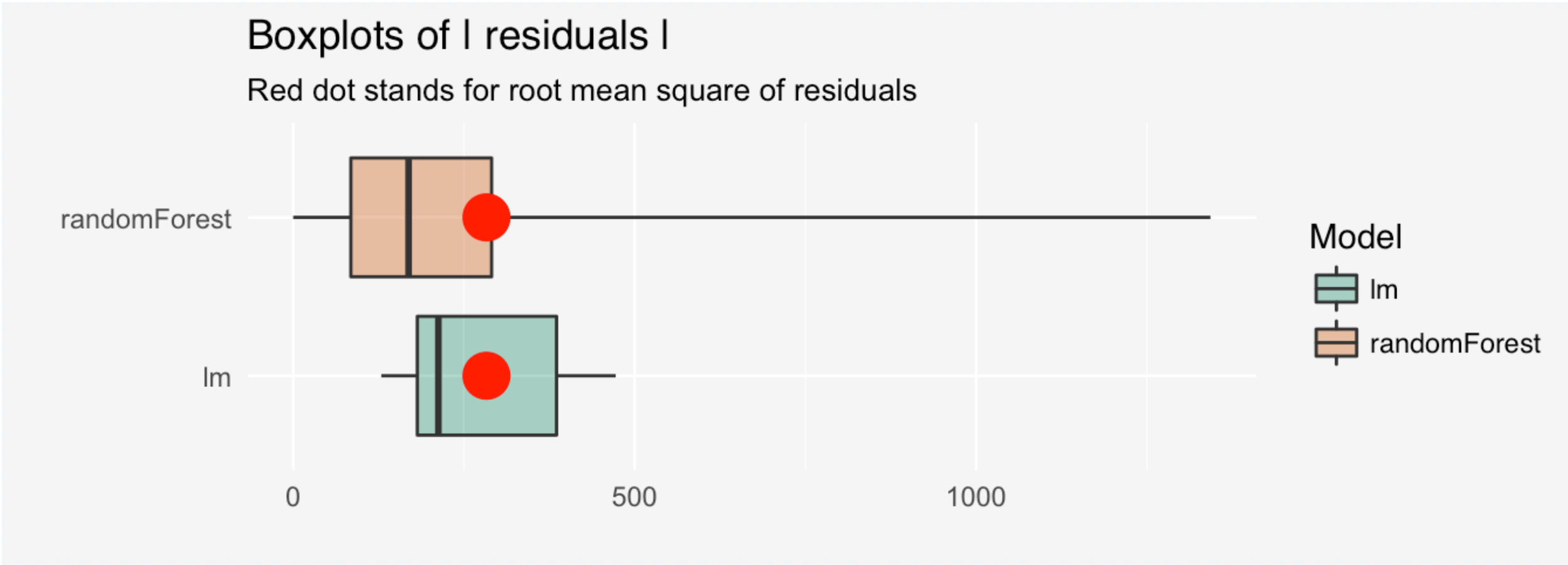
Argument	Summary
`family = "gaussian"`	For 1/n-dimensional Gaussian
`family = "binomial"`	For binomial regression
`family = "survival"`	For Cox regression

The response panel summaries

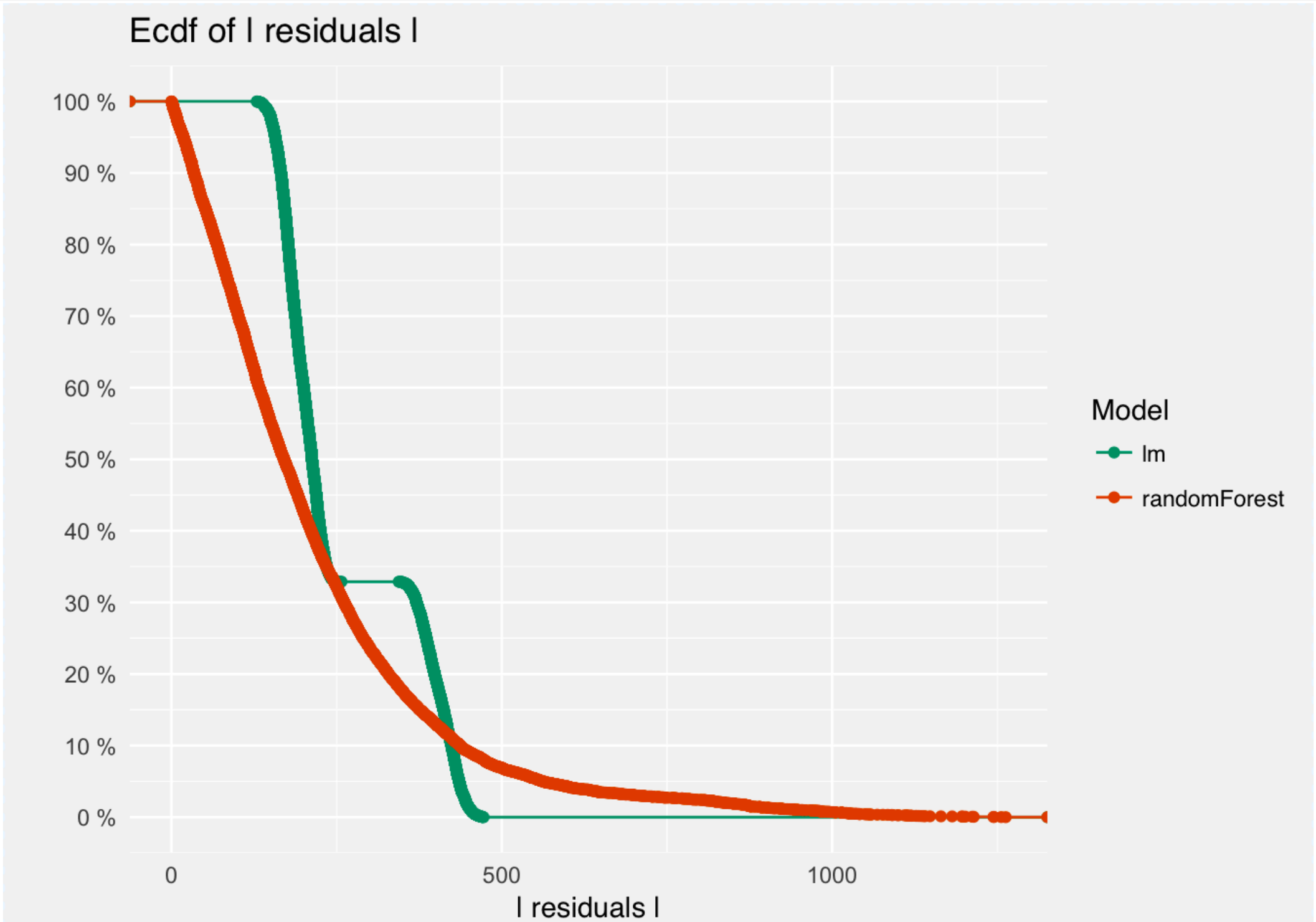


Accuracy as a single number is not enough!

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

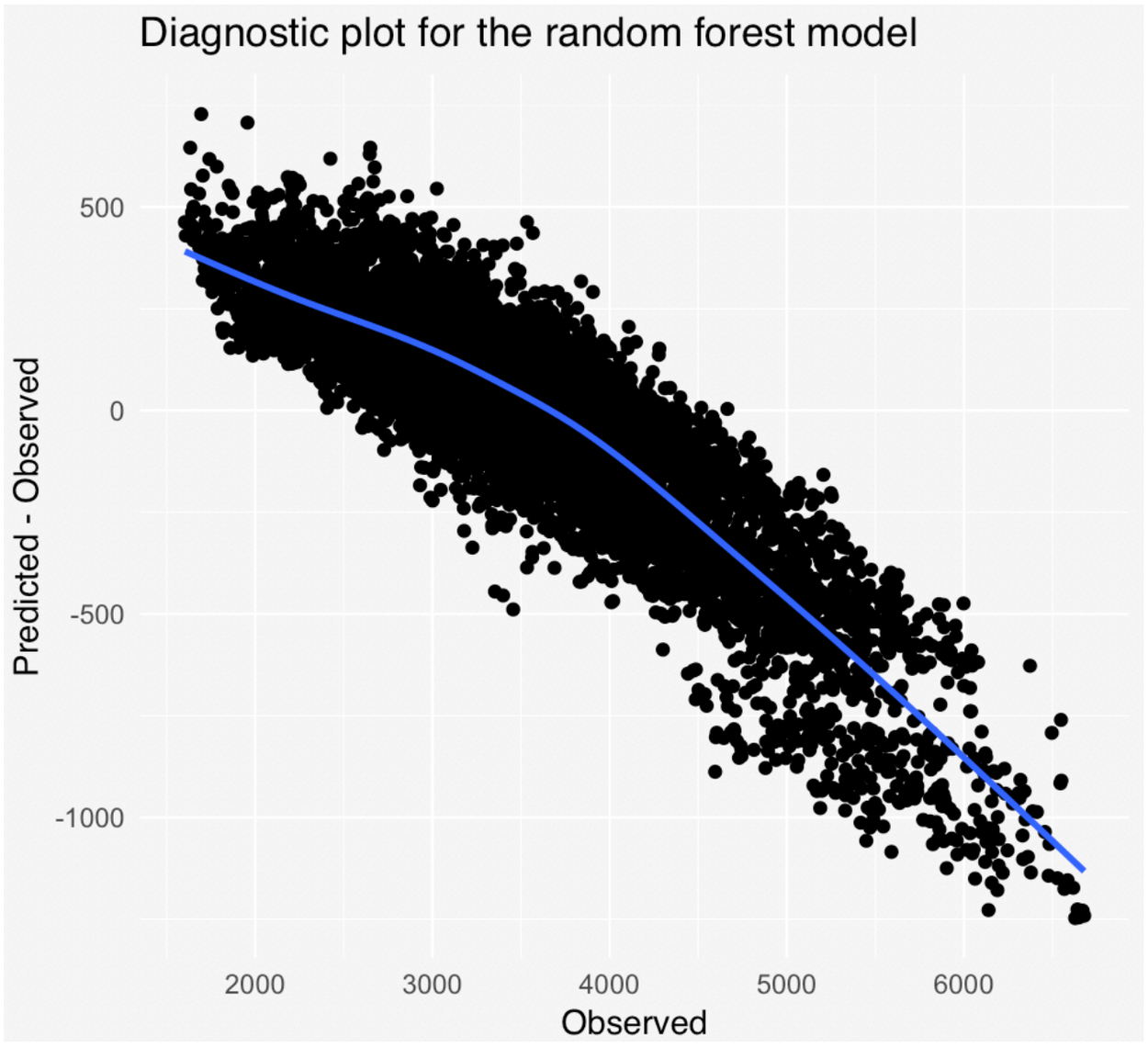


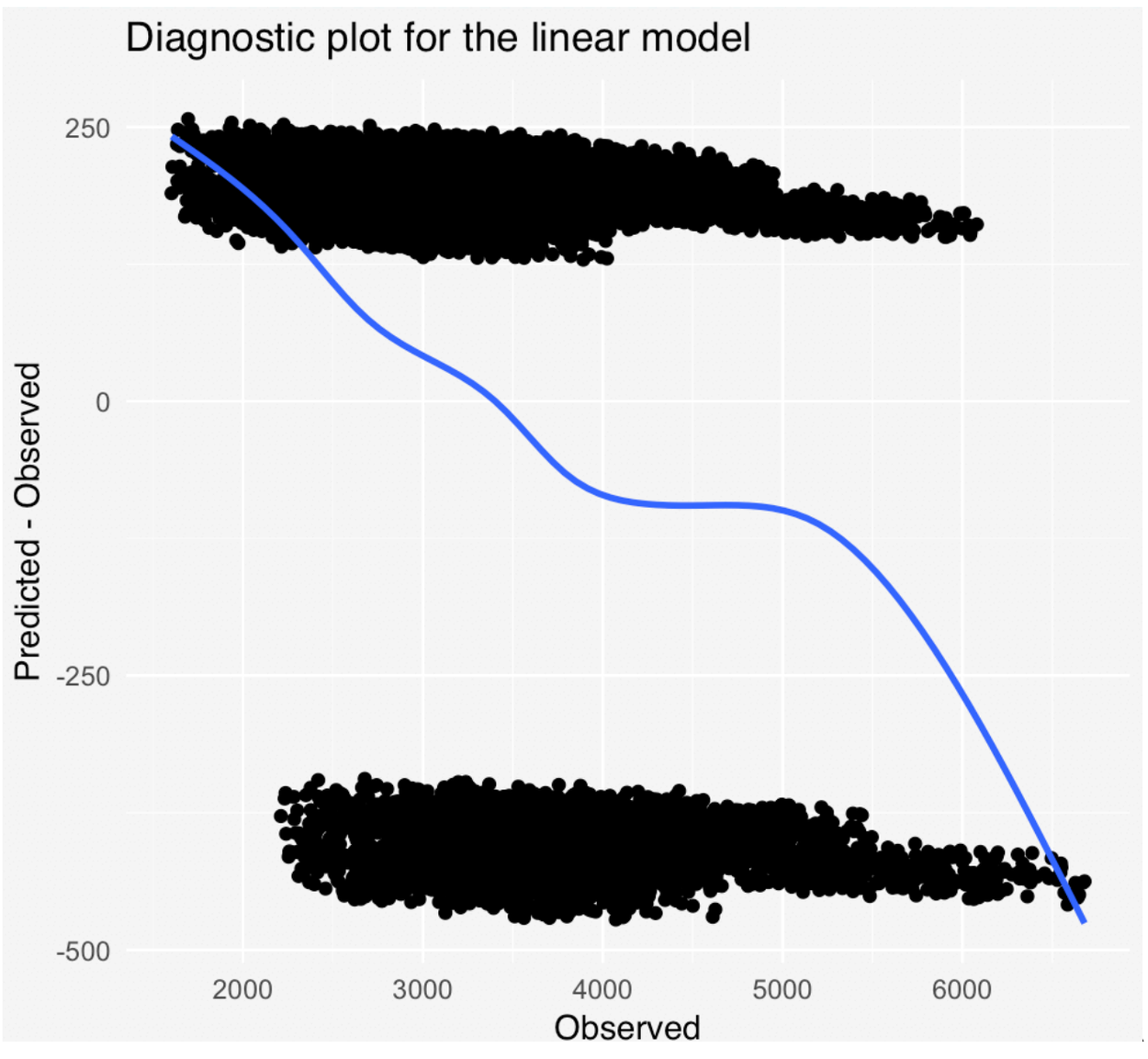
Accuracy as a single number is not enough!



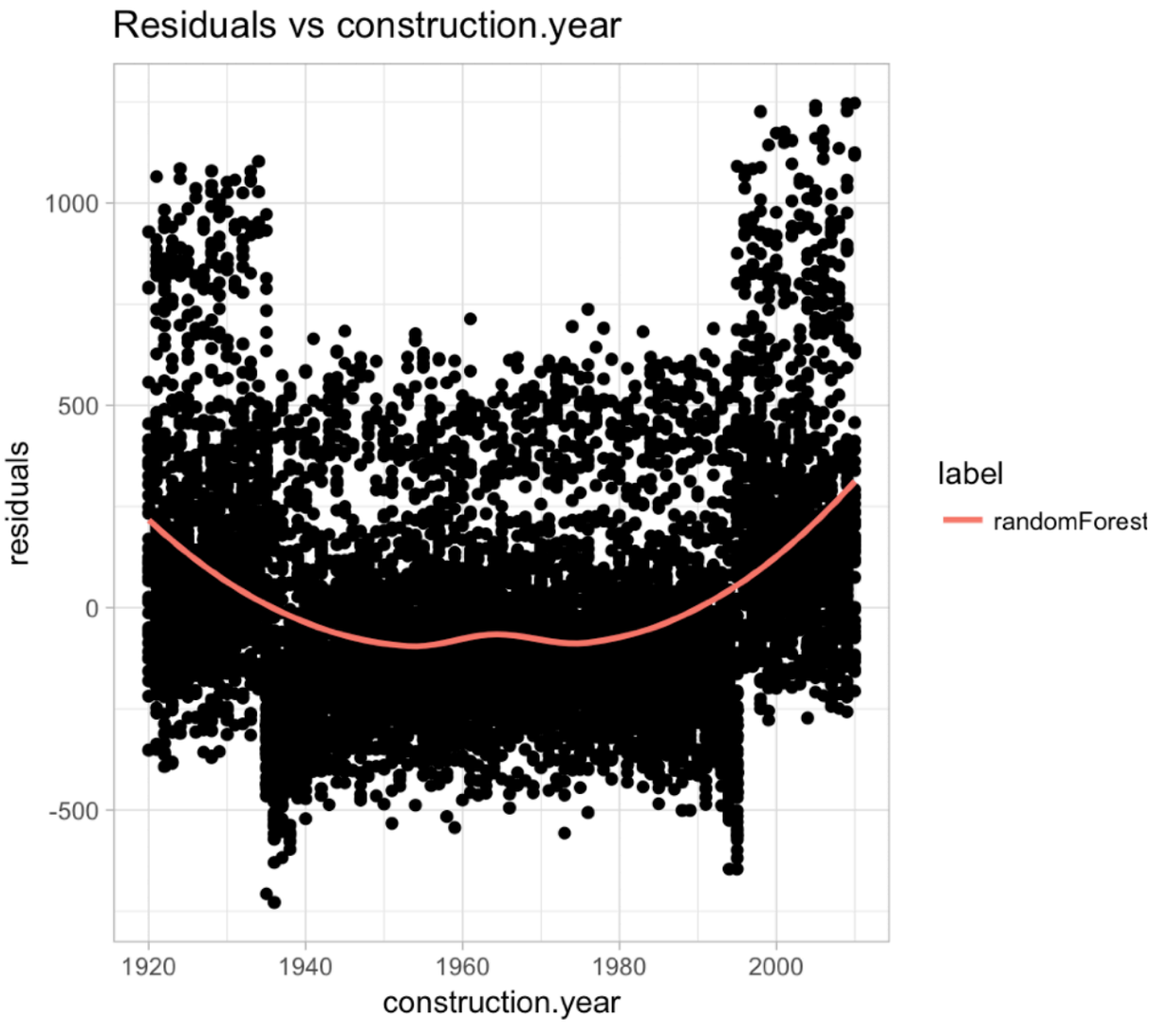
Accuracy as a single number is not enough!

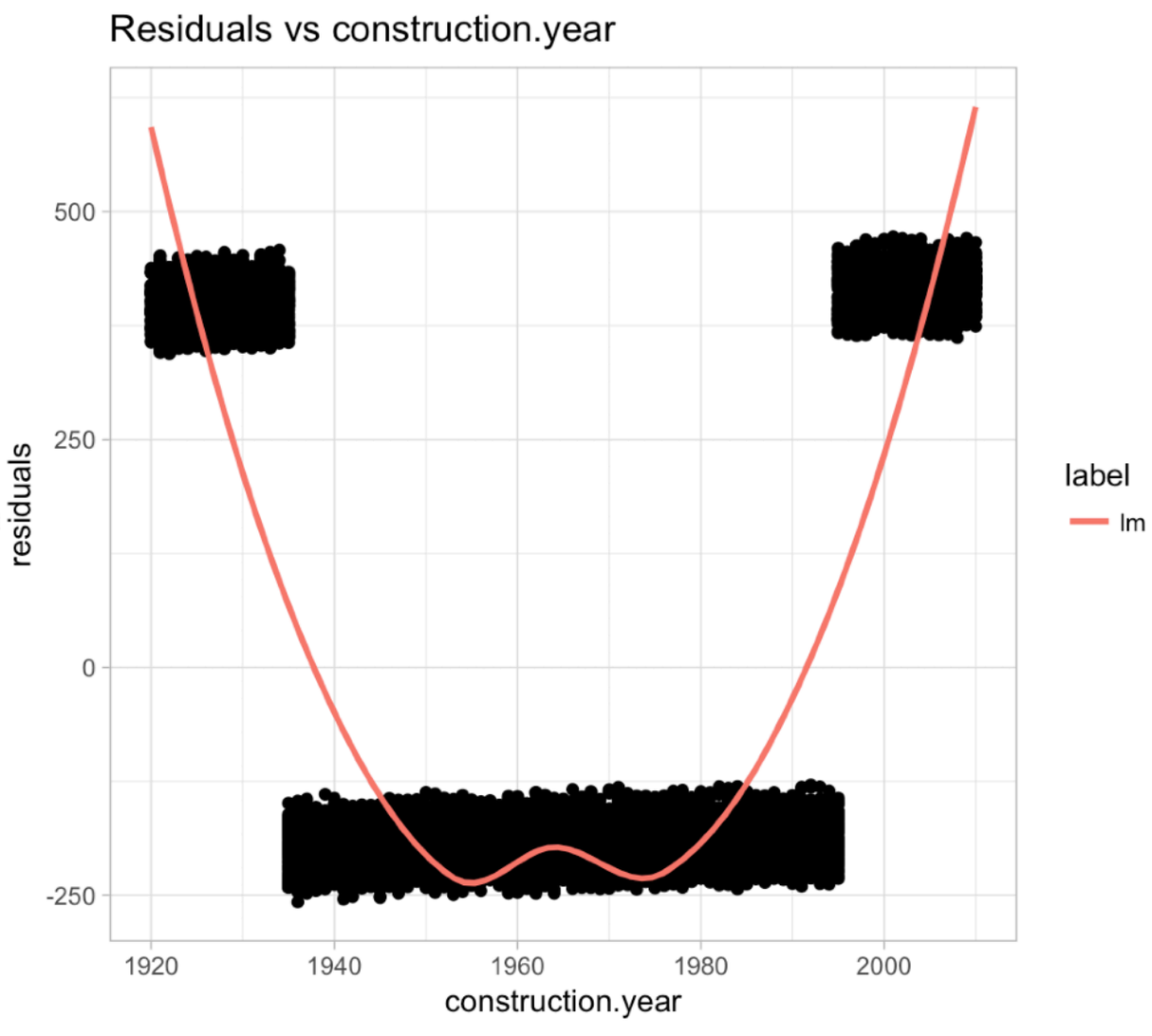
Always validate your model!





```r
library(auditor)
audit_rf <- audit(explainer_rf)
plotResidual(audit_rf, variable = "construction.year")

audit_lm <- audit(explainer_lm)
plotResidual(audit_lm, variable = "construction.year")
```





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

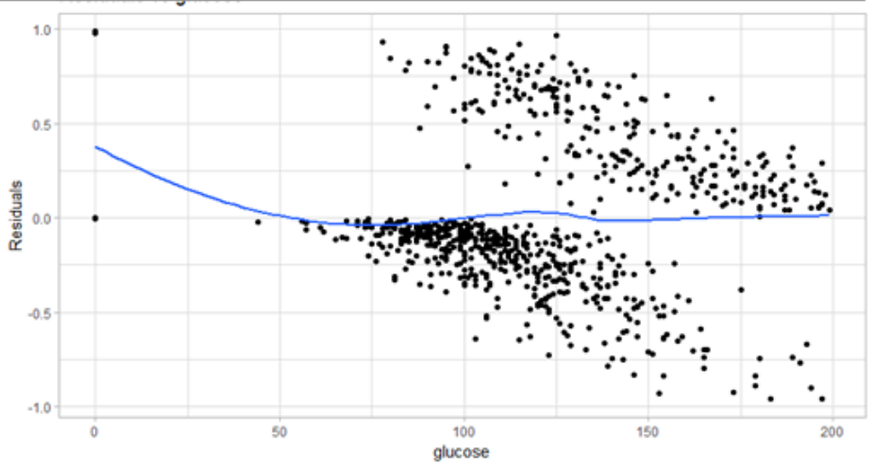
REFERNCES

- *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 different graphs.
By default it is drawn *Residuals vs variable plot*.

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

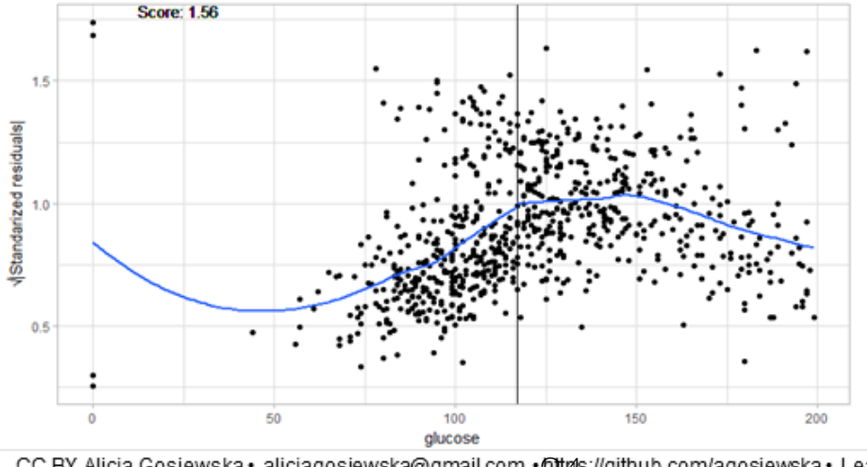


In the example above there were used response residuals, but it is possible to provide any predict and residual function.
To use non-defaulted functions to calculate the residuals and model prediction, you can use arguments **residual.function** and **predict.function** while using an audit function.

```
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")
```

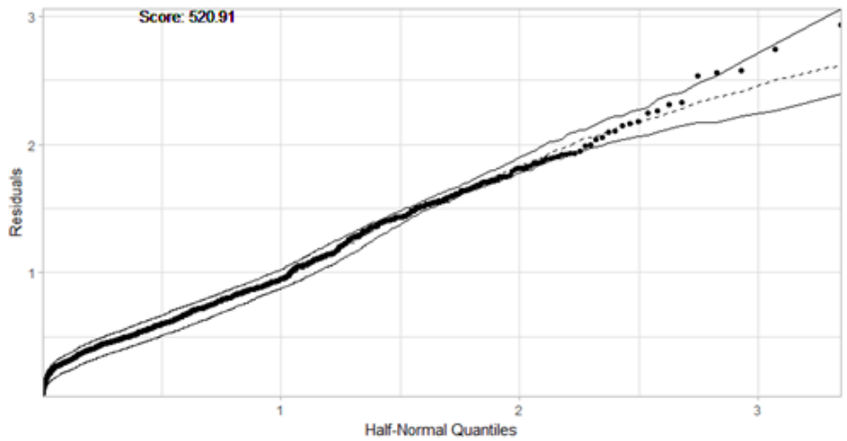


Half-Normal plots

The half-normal plot is a graphical method for comparing two probability distributions by plotting their quantiles against each other.
Points on the plot correspond to ordered absolute values of model residuals plotted against theoretical order statistics from a half-normal distribution.

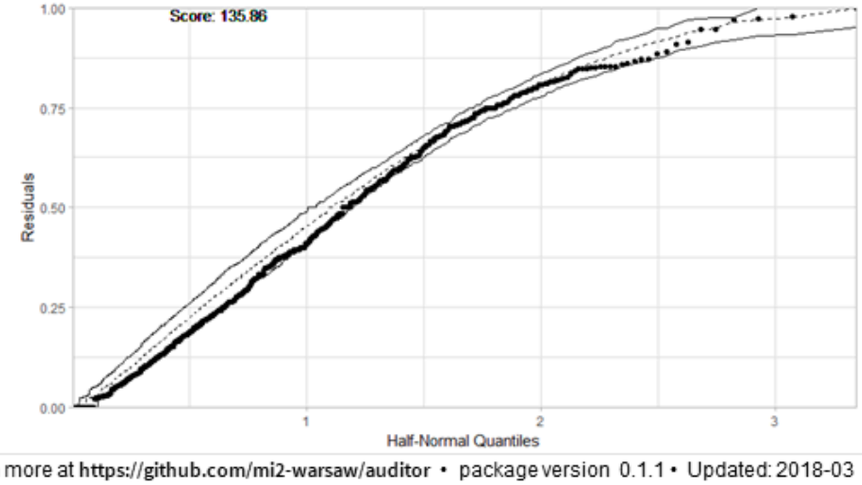
If residuals are from the normal distribution, they are close to a straight line. However, if they don't come from a normal distribution, they still show a certain trend. Simulated envelopes can be used to help verify the correctness of this trend. For a well-fitted model, residuals should lay within the envelope.

```
plotHalfNormal(audit_glm_dev, resid.type="response")
```



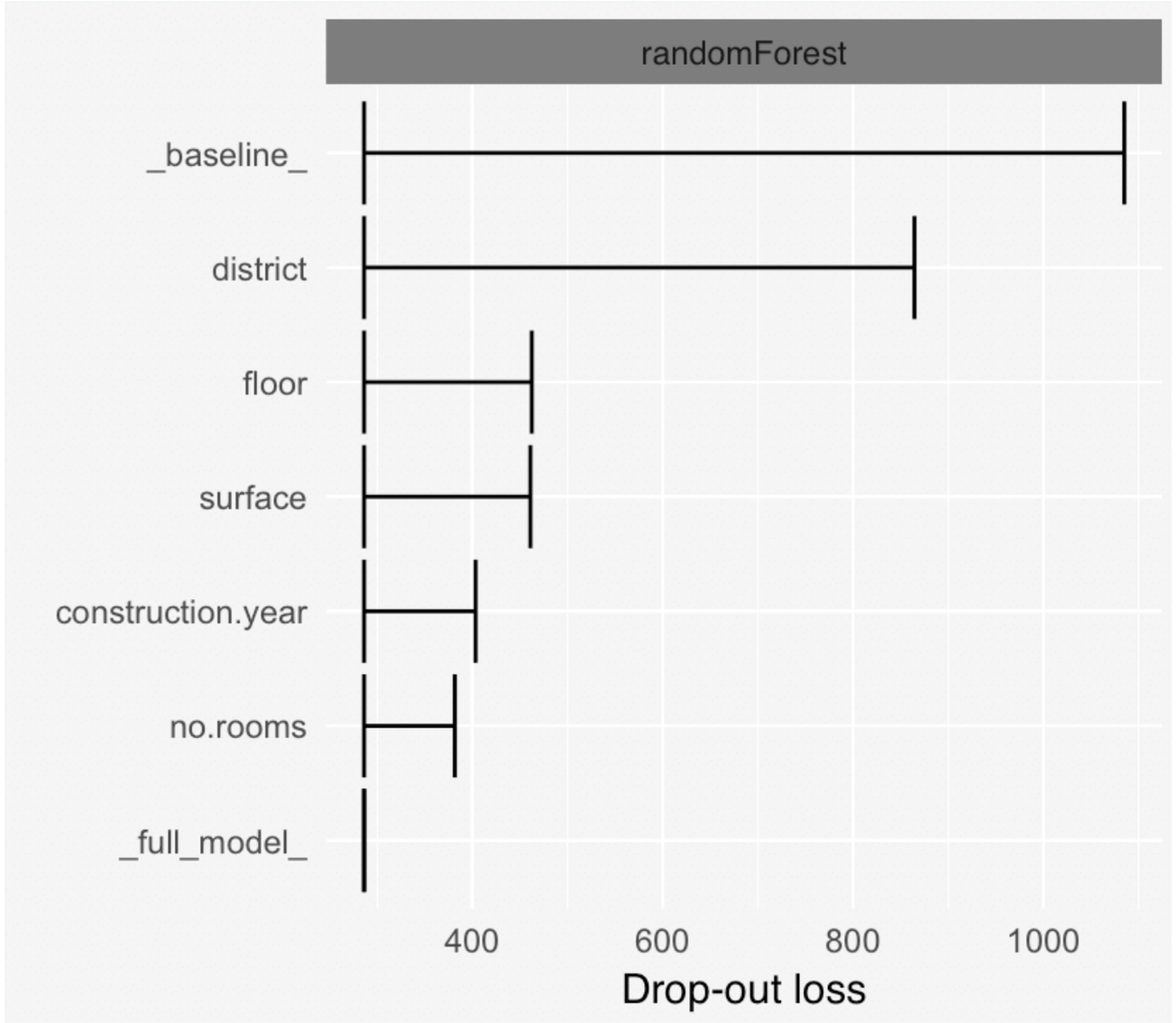
Half-Normal plots allow to check the model fit but comparing two models may be difficult.
A useful tool to compare goodness-of-fit of two models is score displayed on the plot. Score is a sum of logarithms of estimated PDF at each point. It is calculated on the basis of simulated data, so it may differ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

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



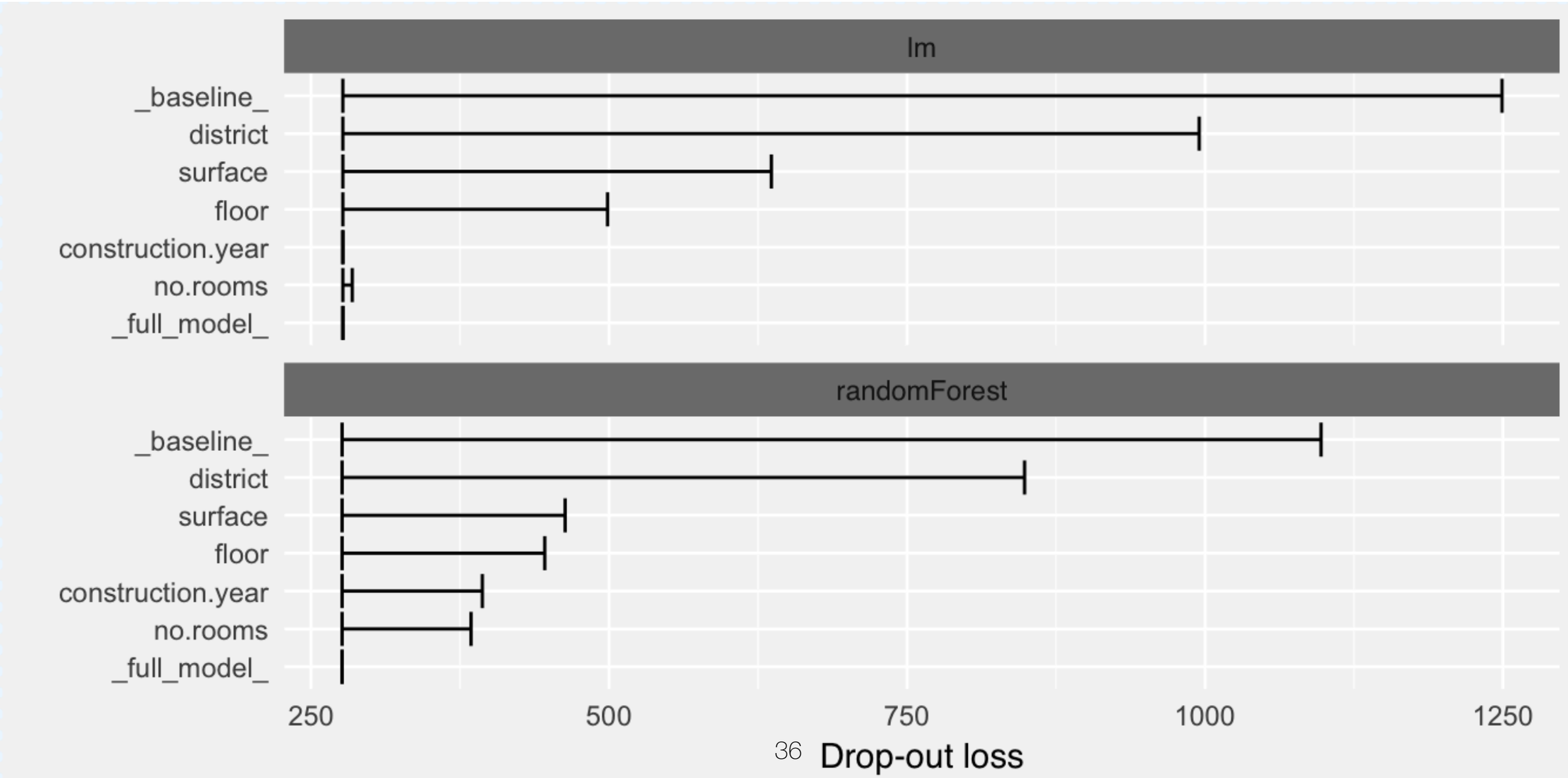
Which variables are influential for RF?

```
> vi_rf <- variable_importance(explainer_rf, loss_function = loss_root_mean_square)
> vi_rf
           variable dropout_loss         label
1      _full_model_     286.2676 randomForest
2          no.rooms     381.5975 randomForest
3 construction.year     403.4376 randomForest
4           surface     461.0018 randomForest
5             floor     462.3999 randomForest
6          district     864.4315 randomForest
7        _baseline_    1084.9218 randomForest
>
> plot(vi_rf)
```



Find 5 differences!

Such statistics may be directly compared across variables and across models!



randomForestExplainer
What's in the forest?

Aleksandra Paluszynska [aut, cre]
Przemysław Biecek [aut, ths]
University of Warsaw

randomForestExplainer - Structure mining and visualisation for Random Forests

Basics

The aim of the **randomForestExplainer** package is to support structure exploration and visualisation for a random forest model.

Once you have a model created with the **randomForest** package, use following functions to examine its structure.

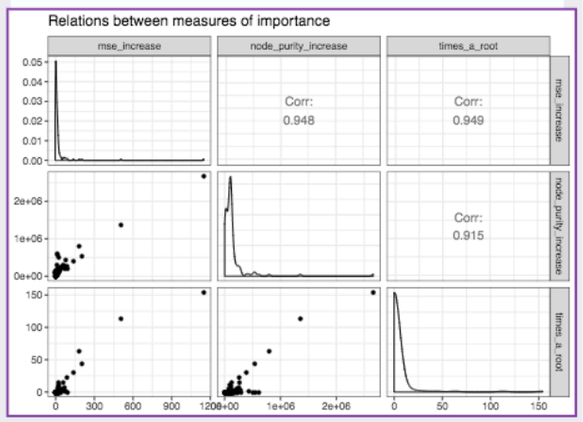
```
library(randomForest)
library(randomForestExplainer)

forest <- randomForest(PV1MATH~.,
data = pisa2015, localImp = TRUE)
```

Variable Importance

The **measure_importance()** function calculates different measures of importance for variables presented in the forest. Note that different variables are available for classification forests and regression forests.

Use the **plot_importance_ggpairs()** function to plot examine relations between selected measures.

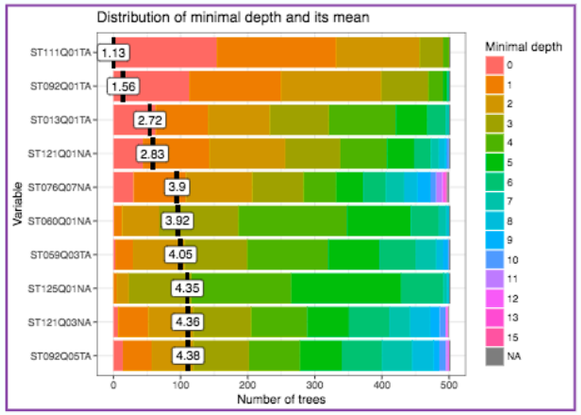
```
forest_stats <-
measure_importance(forest, measures =
c("mse_increase",
"node_purity_increase",
"times_a_root"))

plot_importance_ggpairs(forest_stats)
```



Variable Depth

The **min_depth_distribution()** function calculates distribution of minimal depth of given variable in all trees. Use the **plot_min_depth_distribution()** function to plot this distribution along with mean depths for variables. In general, the higher are variables the more influential they are.

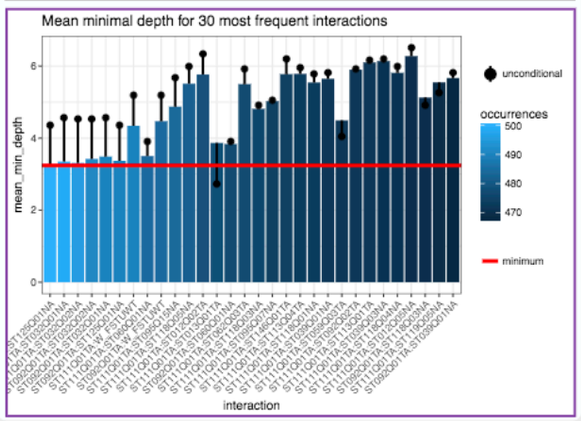
```
forest_frame <-
min_depth_distribution(forest)

plot_min_depth_distribution(forest_fram
e)
```



The **min_depth_interactions()** function calculates conditional depth of variables in subtrees rooted in the selected variable. Such statistic is useful to identify interactions of two variables.

Use the **plot_min_depth_interactions()** function to plot such statistics.

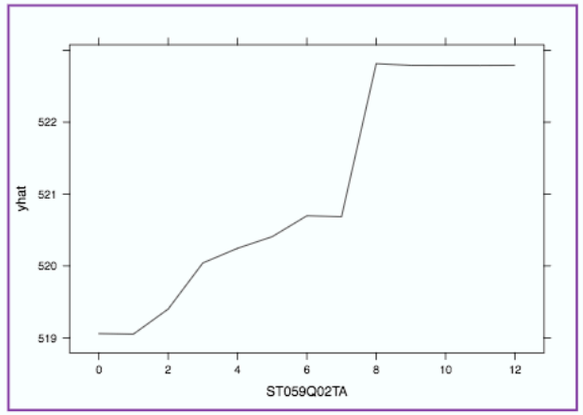
```
forest_interactions <-
min_depth_interactions(forest)
plot_min_depth_interactions(forest_inte
ractions)
```



Partial Dependence Plots

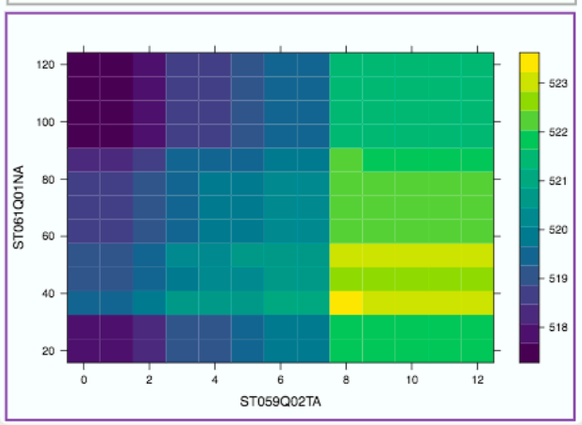
The **partial()** function from the **pdp** package calculates marginal relation between target variable and selected one or two independent variables. The relation can be noted with **lattice** graphical system with the **plotParial()** function or with **ggplot2** system with the **autoplot()** function.

```
library(pdp)

pdp1 <- partial(forest, "ST059Q02TA")
plotPartial(pdp1)
```



Variable depth and variable importance functions are useful in identification which variable/variables are worth watching, while the **pdp** plots are useful to understand the nature of the relation between target variable and variable/s of interest.

```
pdp2 <- partial(forest, c("ST059Q02TA",
"ST061Q01NA"))
plotPartial(pdp2)
```



Local Approximations

Based on LIME

Literature

ggRandomForests: Random Forests for Regression. John Ehrlinger (2016)

pdp: An R Package for Constructing Partial Dependence Plots. Brandon M. Greenwell (2017)

forestFloor: Forest Floor Visualizations of Random Forests. Soeren Welling, Hanne Refsgaard, Per Brockhoff, Line Clemmensen (2016)

Language for Human-Model Interaction