DALEX:
Descriptive mAchine Learning EXplanations

Przemysław Biecek

use R!

Please make sure that you have following packages

```
install.packages(c("DALEX",              "breakDown",
                   "ceterisParibus",     "live",
                   "randomForest",       "auditor"))
```

Materials for this workshop are available at

http://bit.do/DALEX

Associate Professor in Machine Learning
@ Warsaw University of Technology, PL

Interested in:
Explanations of Black-Box ML models
Molecular Oncology
Statistical Modelling
DataVis

15 years of experience in
Teaching, Using and Programming in R (3 books)

Head of MI2DataLab http://mi2.mini.pw.edu.pl/
Contact http://biecek.pl/

House rules

- Feel free to interrupt me if something is not clear
- We will have a coffee break in 90 minutes
- We will have 4 breaks for hands-on exercises (2 before and 2 after coffee break). Let me know if you need any help
- Collect your DALEX hex-sticker 

- Let me know if you have some ideas for new explainers (or you can leave an open issue on GitHub)

Do we need explanations 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 l
ety of classifi
of eye disease

SCIENTIFIC REPORTS

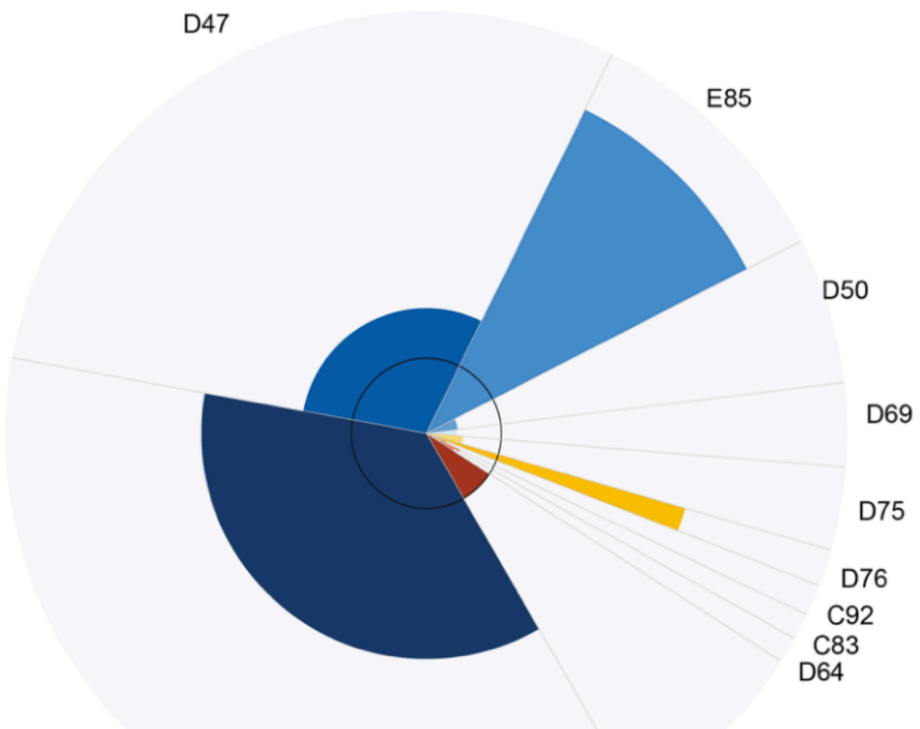
OPEN

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

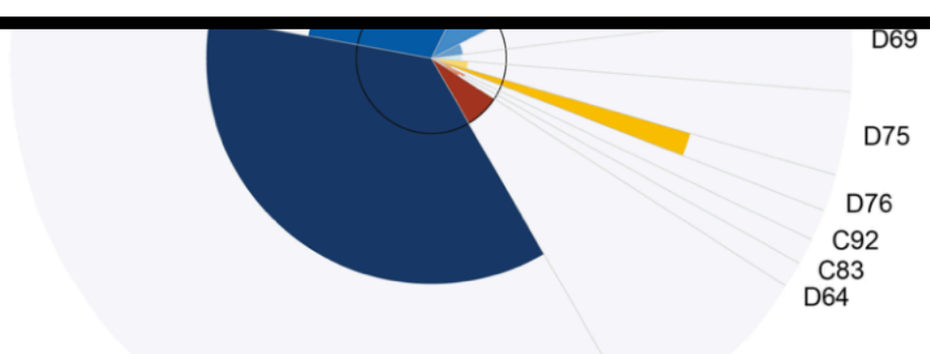
SCIENTIFIC REPORTS

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

Black box models - what they are?

Trade-off 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.

Interactions between input features
Even linear models with sparse input space may be hard to understand if they contains deep interactions (of the order three or higher).

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 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 an 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).

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 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 explanation for decisions that significantly affect an individual, particularly legally or financially. For example, a person who applies for is denied may ask for an explanation, which could be "Credit bureau X reports that you declared bankruptcy last year; this is the main 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 regulations on algorithmic decision-making and a "right to explanation"

Bryce Goodman,[1]* Seth Flaxman,[2]

[1]Oxford Internet Institute, Oxford
1 St Giles', Oxford OX1 3LB, United Kingdom
[2]Department of Statistics, University of Oxford,
24-29 St Giles', Oxford OX1 3LB, United Kingdom

*To whom correspondence should be addressed; E-mail: flaxman@stats.ox.ac.uk.

Abstract

We summarize the potential impact that the European Union's new General Data Protection Regulation will have on the routine use of machine learning algorithms. Slated to take effect as law across the EU in 2018, it will restrict automated individual decision-making (that is, algorithms that make decisions based on user-level predictors) which "significantly affect" users. The law will also effectively create a "right to explanation," whereby a user can ask for an explanation of an algorithmic decision that was made about them. We argue that while this law will pose large challenges for industry, it highlights opportunities for computer scientists to take the lead in designing algorithms and evaluation frameworks which avoid discrimination and enable explanation.

1 Introduction

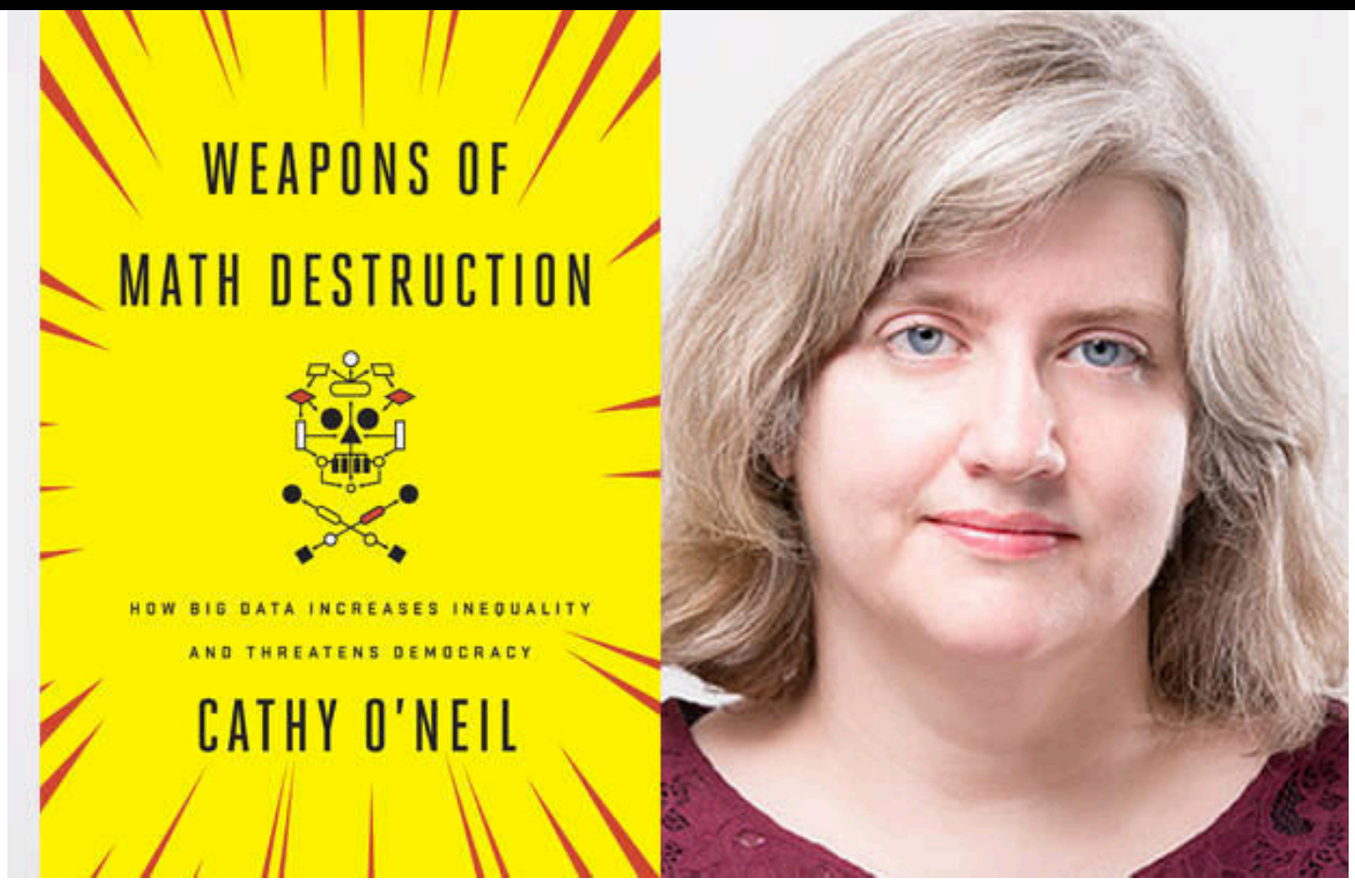
In April 2016, for the first time in over two decades, the European Parliament adopted a set of comprehensive regulations for the collection, storage and use of personal information, the General Data Protection Regulation (GDPR)[1] [26]. The new regulation has been described as a "Copernican Revolution" in data protection law, "seeking to shift its focus away from paper-based, bureaucratic requirements and towards compliance in practice, harmonization of the law, and individual empowerment" [22]. Much of the regulations are clearly aimed at perceived gaps and inconsistencies in the EU's current approach to data protection. This includes, for example, the codification of the "right to be forgotten" (Article 17), and regulations for foreign companies collecting data from European citizens (Article 44).

However, while the bulk of language deals with how data is collected and stored, the regulation contains

Why do we need explanations for complex models?

Cathy O'Neil:

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

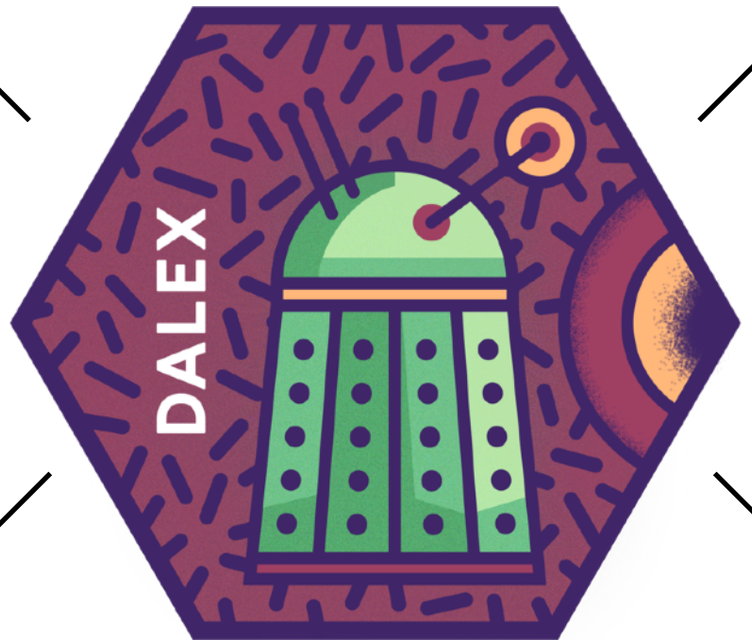


- "You don't see a lot of skepticism," she says. "The algorithms are like shiny new toys that we can't resist using. We trust them so much that we project meaning on to them."
- Ultimately algorithms, according to O'Neil, reinforce discrimination and widen inequality, "using people's fear and trust of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

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

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

How good is the predictive model?



Which variables are the most important in general?

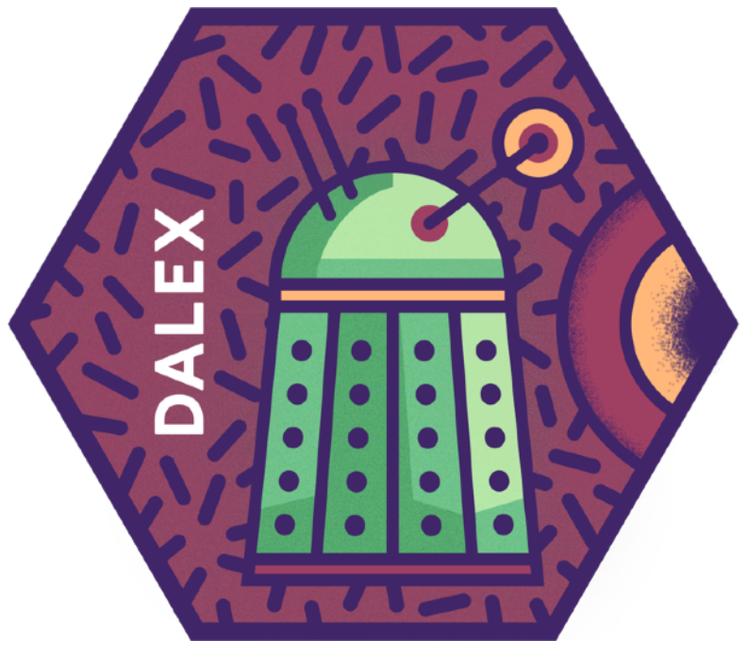
How good is the model fit?

Which variables influence the single prediction?

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

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

Structure Explainers
package: randomForest



Model Management
package: **archivist**

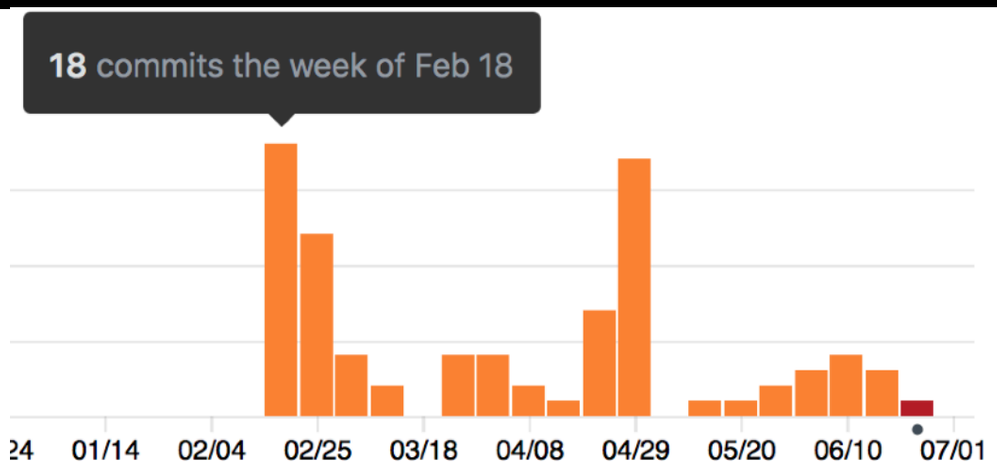
Model Diagnostic Tools
packages: **auditor**, ggfortify

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

Model Predictions Explainers
packages: **breakDown**, **live**, ceterisParibus
shapleyr, lime

DALEX is very young

First commit: 2/18/2018



Active development.
DALEX is meant to be
an *unified interface* to model explainers it's just a glue.

DALEX invasion

- Talk: Complexity Institute @ NTU Singapore, March 2018
- Talk: SER meeting in Warsaw, April 2018
- Workshop: eRum conference, May 2018
- Workshop: STWUR meeting in Wrocław, June 2018
- Workshop: Why R? conference in Wrocław, July 2018
- Workshop: useR! conference in Brisbane, July 2018
- ...

Model specific

or

Model agnostic?

randomForestExplainer
What's in the forest?

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

MI DATA LAB

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.

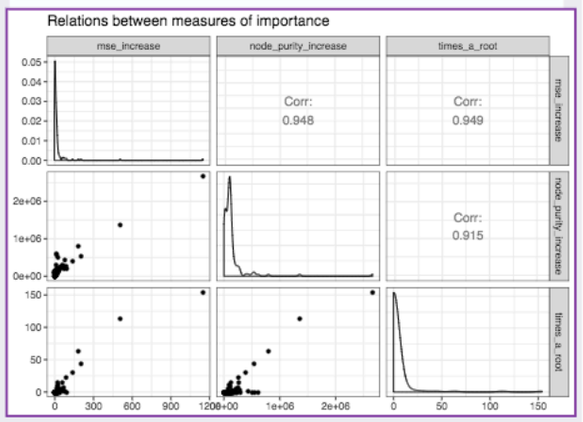
```
library(randomForest)
library(randomForestExplainer)

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)
```

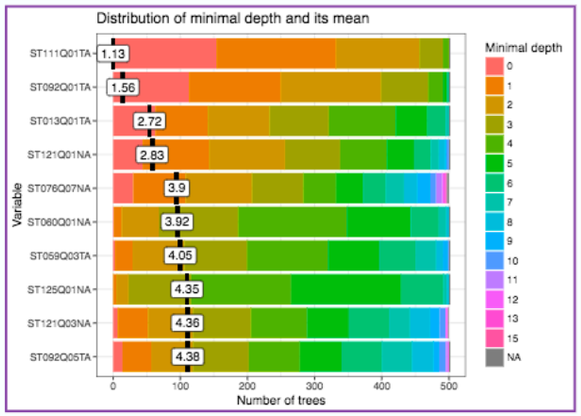


randomForestExplainer - Structure mining and visualisati

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.

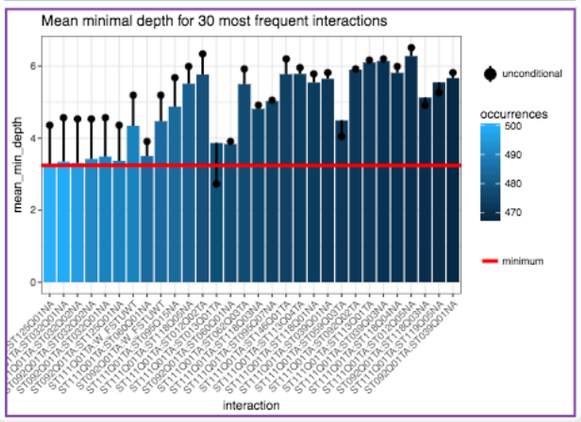
```
forest_frame <-
min_depth_distribution(forest)

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.

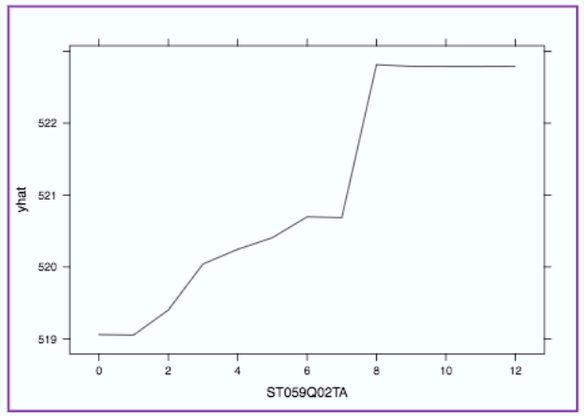
```
forest_interactions <-
min_depth_interactions(forest)
plot_min_depth_interactions(forest_inte
ractions)
```



Partial Dependence Plots

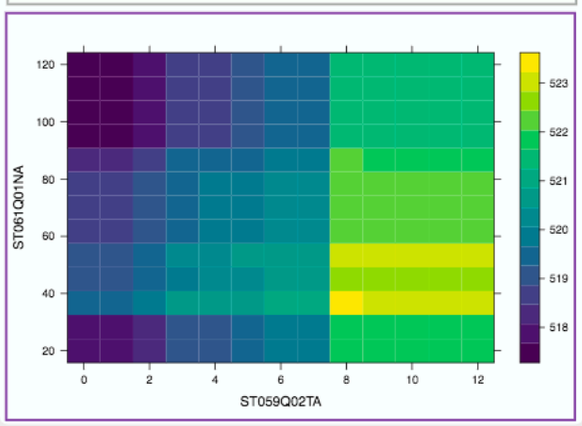
The **partial()** function from the **pdp** package calculates marginal relation between target variable and selected one or two independent variables. The relation can be noted with **lattice** graphical system with the **plotParial()** function or with **ggplot2** system with the **autoplot()** function.

```
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)
```



Lots of diagnostic tools for linear models

Some build-in tools for random forest and xgboost

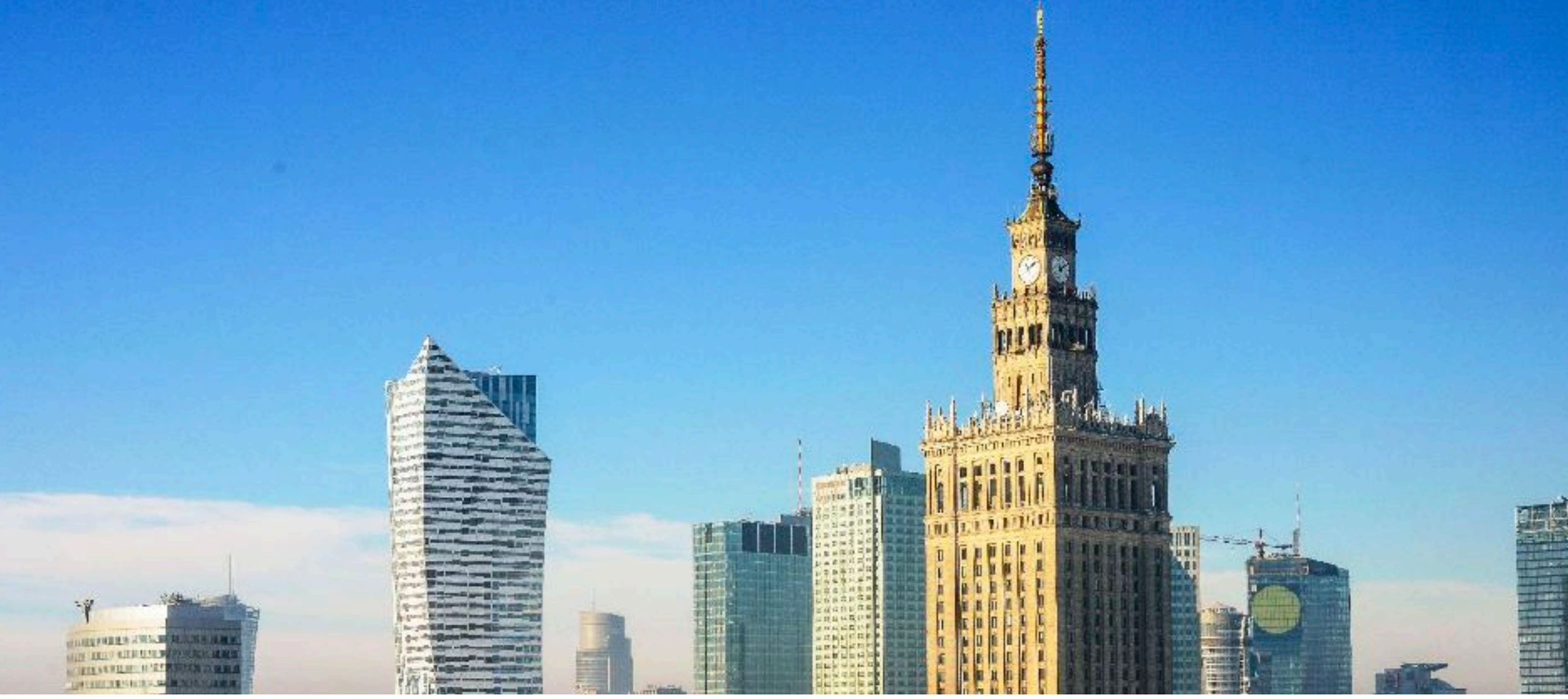
Some specialised tools like randomForestExplainer xgboostExplainer

DALEX is fully model agnostic

Use Case:

Predictive models for apartment prices

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



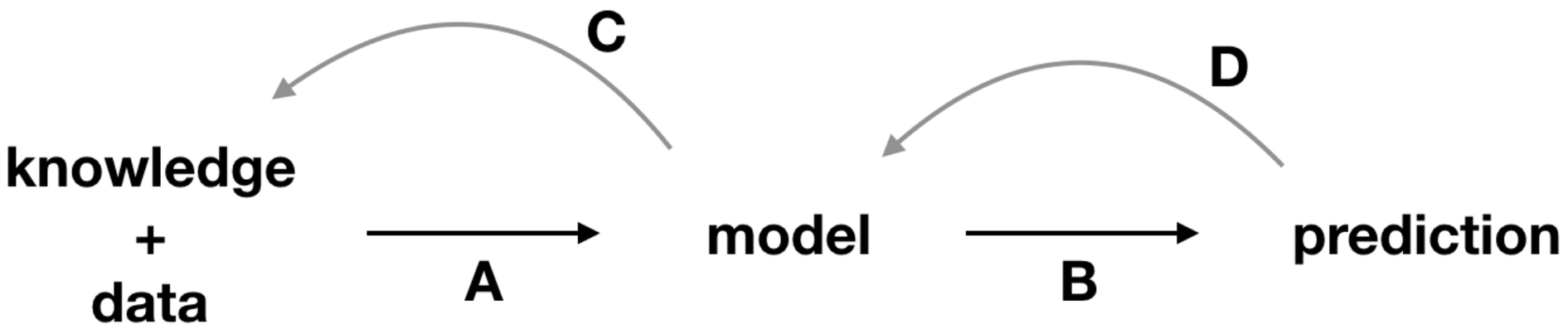
```
library("DALEX")
head(apartments)
```

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

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.*

Let's create two competing models

```
> library("DALEX")
> apartments_lm_model <- lm(m2.price ~ construction.year + surface + floor +
+                             no.rooms + district, data = apartments)
>
> library("randomForest")
> set.seed(3)
> apartments_rf_model <- randomForest(m2.price ~ construction.year + surfac
+                                     no.rooms + district, data = apartmen
```

Let's create two competing models

```
> library("DALEX")
> apartments_lm_model <- lm(m2.price ~ construction.year + surface + floor +
+                             no.rooms + district, data = apartments)
>
> library("randomForest")
> set.seed(3)
> apartments_rf_model <- randomForest(m2.price ~ construction.year + surfac
+                                     no.rooms + district, data = apartmen
>
> predicted_mi2_lm <- predict(apartments_lm_model, apartmentsTest)
> sqrt(mean((predicted_mi2_lm - apartmentsTest$m2.price)^2))
[1] 283.0865
>
> predicted_mi2_rf <- predict(apartments_rf_model, apartmentsTest)
> sqrt(mean((predicted_mi2_rf - apartmentsTest$m2.price)^2))
[1] 283.3479
```

Let's create two competing models

```
> library("DALEX")
> apartments_lm_model <- lm(m2.price ~ construction.year + surface + floor +
+                             no.rooms + district, data = apartments)
>
> library("randomForest")
> set.seed(3)
> apartments_rf_model <- randomForest(m2.price ~ construction.year + surface
+                                     no.rooms + district, data = apartmen
>
> predicted_mi2_lm <- predict(apartments_lm_model, apartmentsTest)
> sqrt(mean((predicted_mi2_lm - apartmentsTest$m2.price)^2))
[1] 283.0865
>
> predicted_mi2_rf <- predict(apartments_rf_model, apartmentsTest)
> sqrt(mean((predicted_mi2_rf - apartmentsTest$m2.price)^2))
[1] 283.3479
```

Which one is better?

Agenda

How to understand a black-box model?

Choose the right visual explainer in 2.875 simple steps

1. Want to understand a model or a single prediction?
- entire model
- prediction for a single observation

2. Is it *how to change it* or *why it happened*?
- interested in *what-if* scenarios
- how variables affected this single prediction

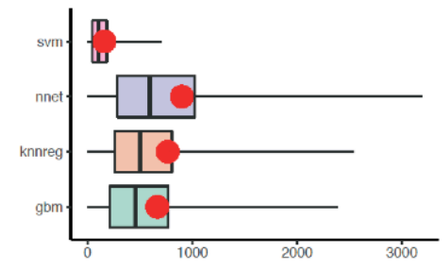
3. Variable attribution or importance?
- decompose prediction (breakDown, Shapley)
- identify key features (live, LIME)

3. Evaluate performance or validate fit?
- compare models performance
- audit residuals and goodness of fit

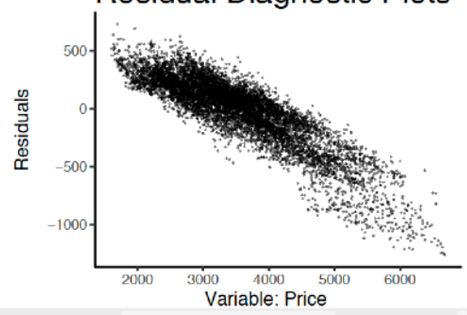
2. Interested in model performance or structure?
- how good is the model
- how does it work

3. Which variable are you interested in?
- all
- a categorical
- a continuous

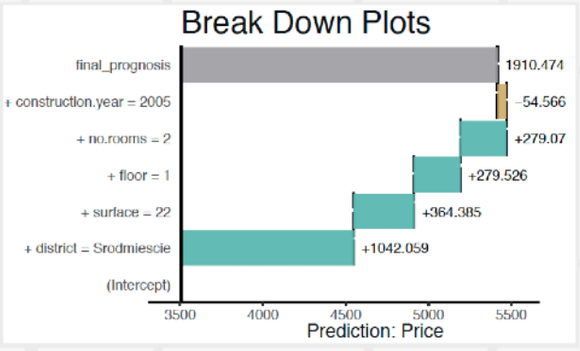
Model Performance Plots



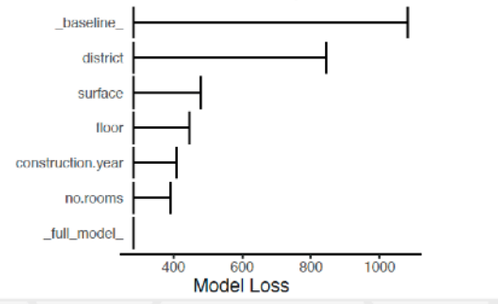
Residual Diagnostic Plots



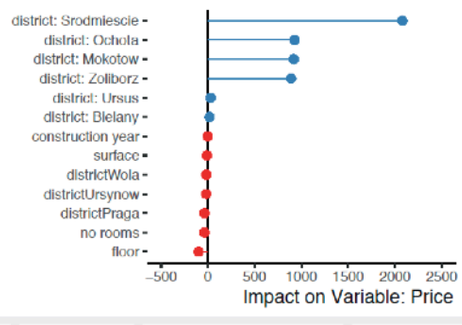
Break Down Plots



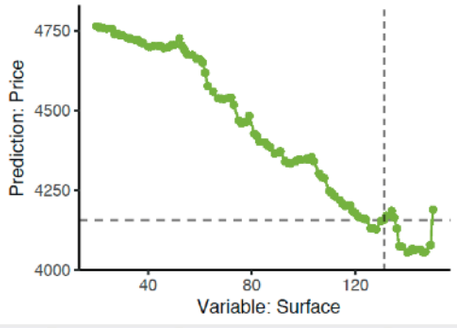
Variable Importance Plots



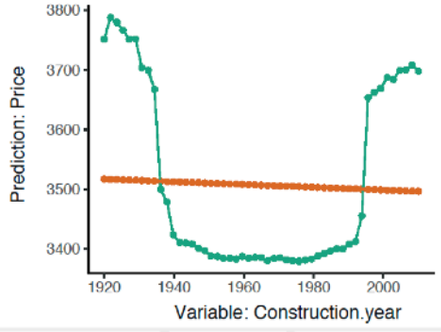
Local Variable Importance



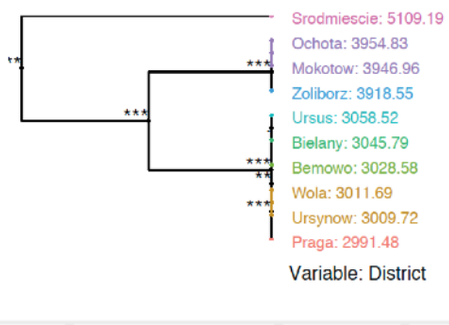
Ceteris Paribus Plots



Partial Dependency Plots



Merging Path Plots



Find more at:
https://github.com/pbiecek/DALEX

DALEX architecture

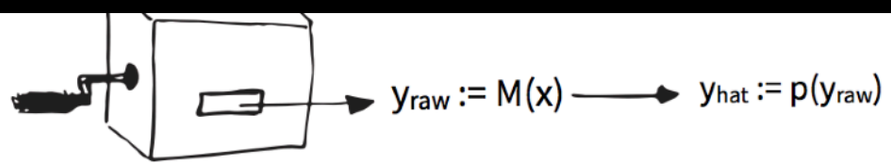
Wrap model

Create explainer

Plot explainer

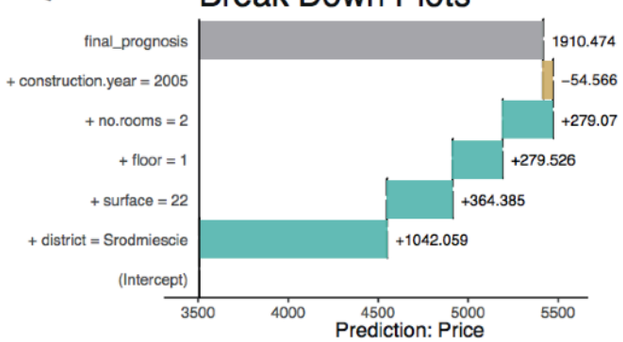
DALEX architecture

A) model

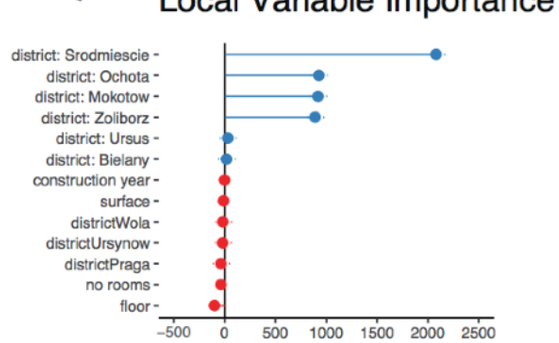


B) explain(model; data; y; predict_function)

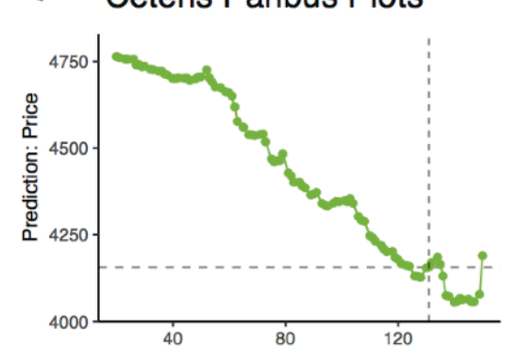
C)



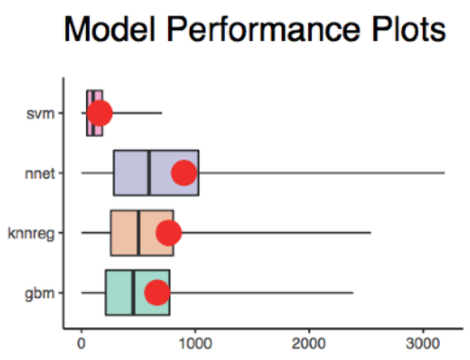
D)



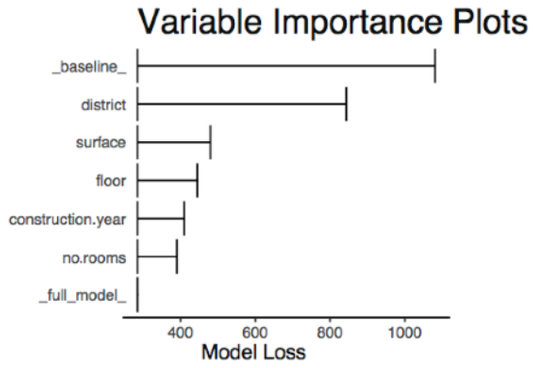
E)



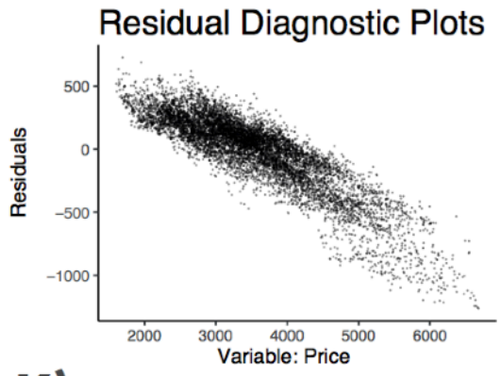
F)



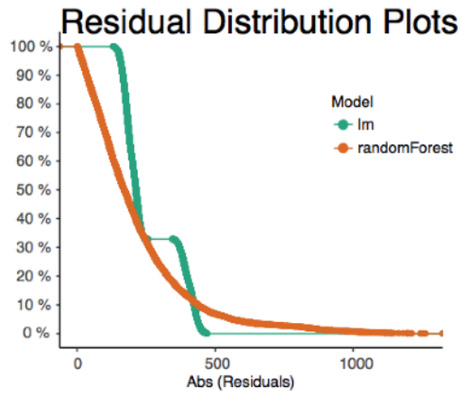
G)



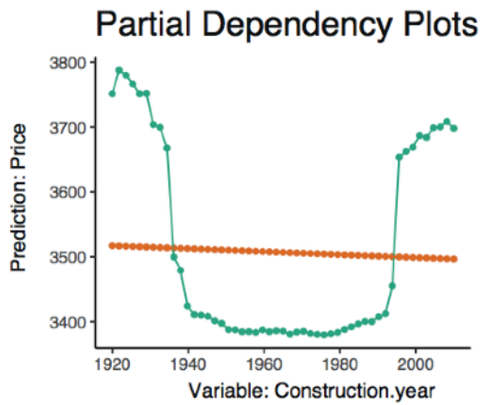
H)



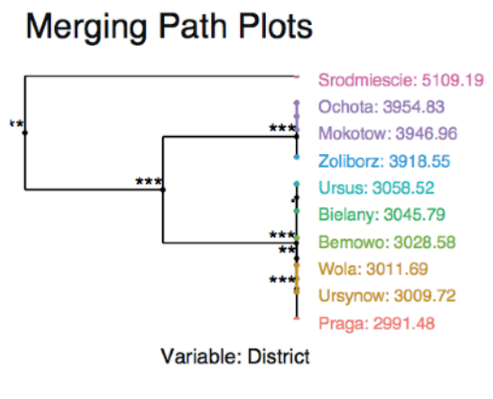
I)



J)



K)



Wrap model

↓

Create explainer

↓

Plot explainer

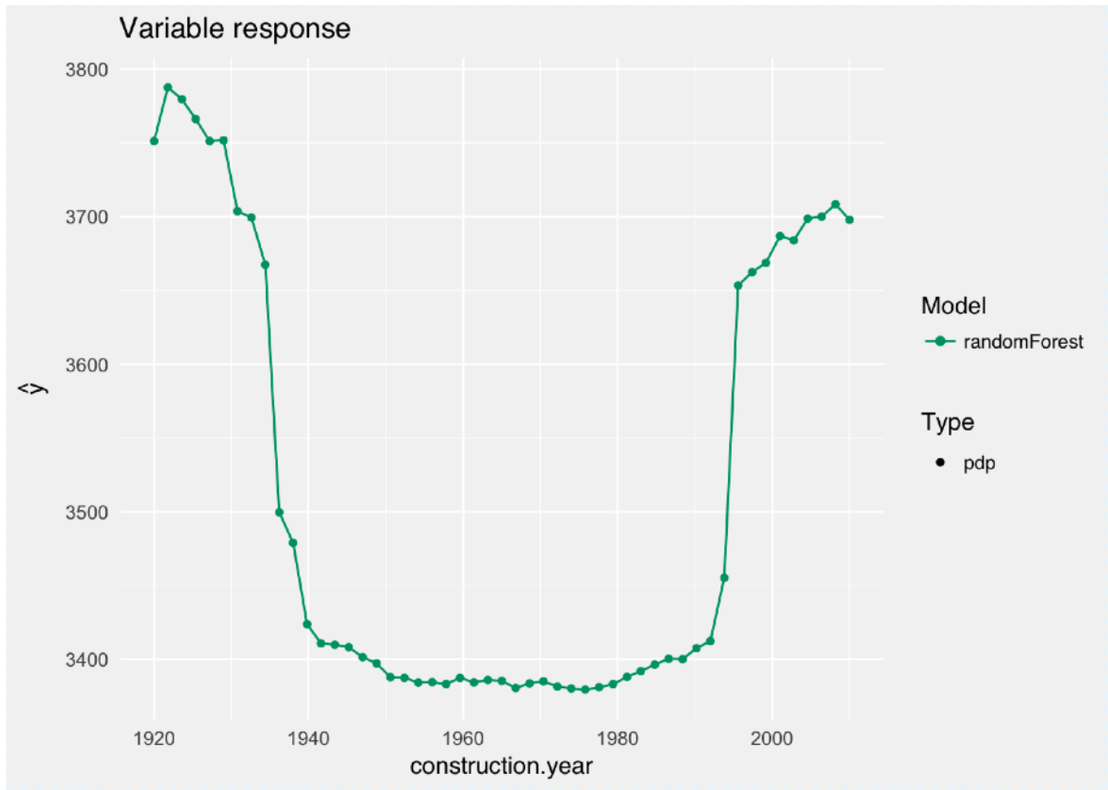
DALEX architecture

```
# wrap model
explainer_rf <- explain(apartments_rf_model,
                        data = apartmentsTest[,2:6],
                        y    = apartmentsTest$m2.price)
# create explainer
sv_rf  <- single_variable(explainer_rf,
                        variable = "construction.year",
                        type     = "pdp")
```

Wrap model

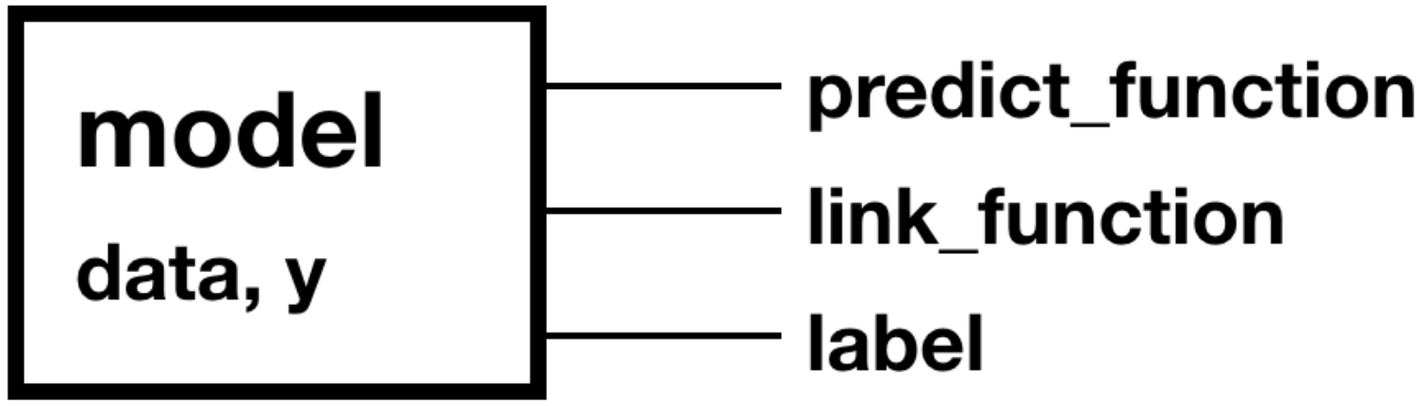
Create explainer

```
# plot explainer
plot(sv_rf)
```



Plot explainer

Why do we need a wrapper?



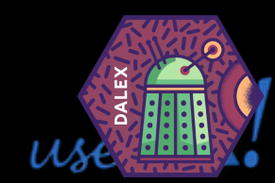

```
explain(model, data, y, predict_function,
                link, ..., label)
```

Explainer encapsulates:

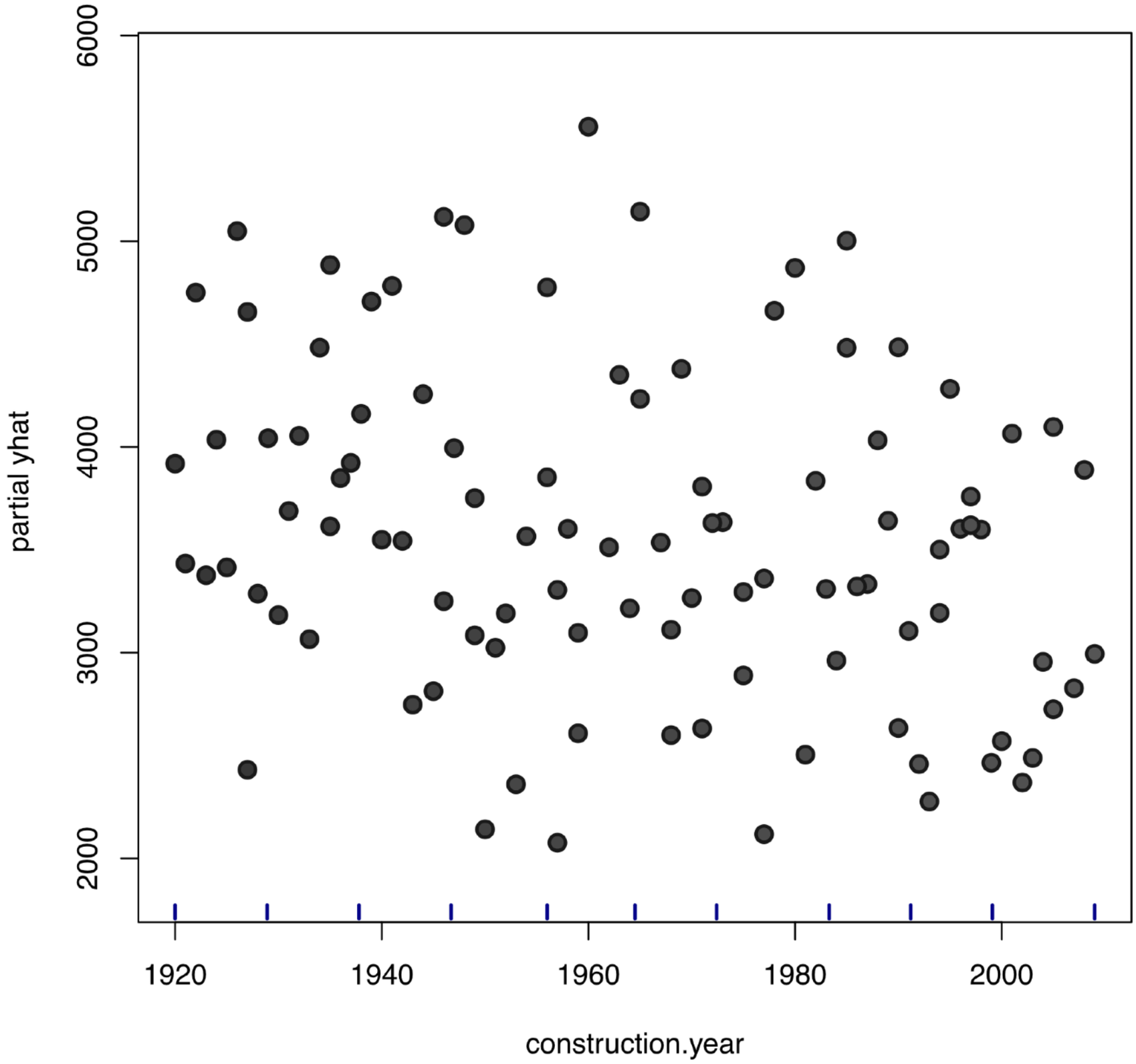
- model (any class)
- validation data and labels (y)
- predict interface that returns vector of scores
- other arguments that will be passed to predict_function
- model name

Model explainers
Continuous variable

How a single feature affects the model response?

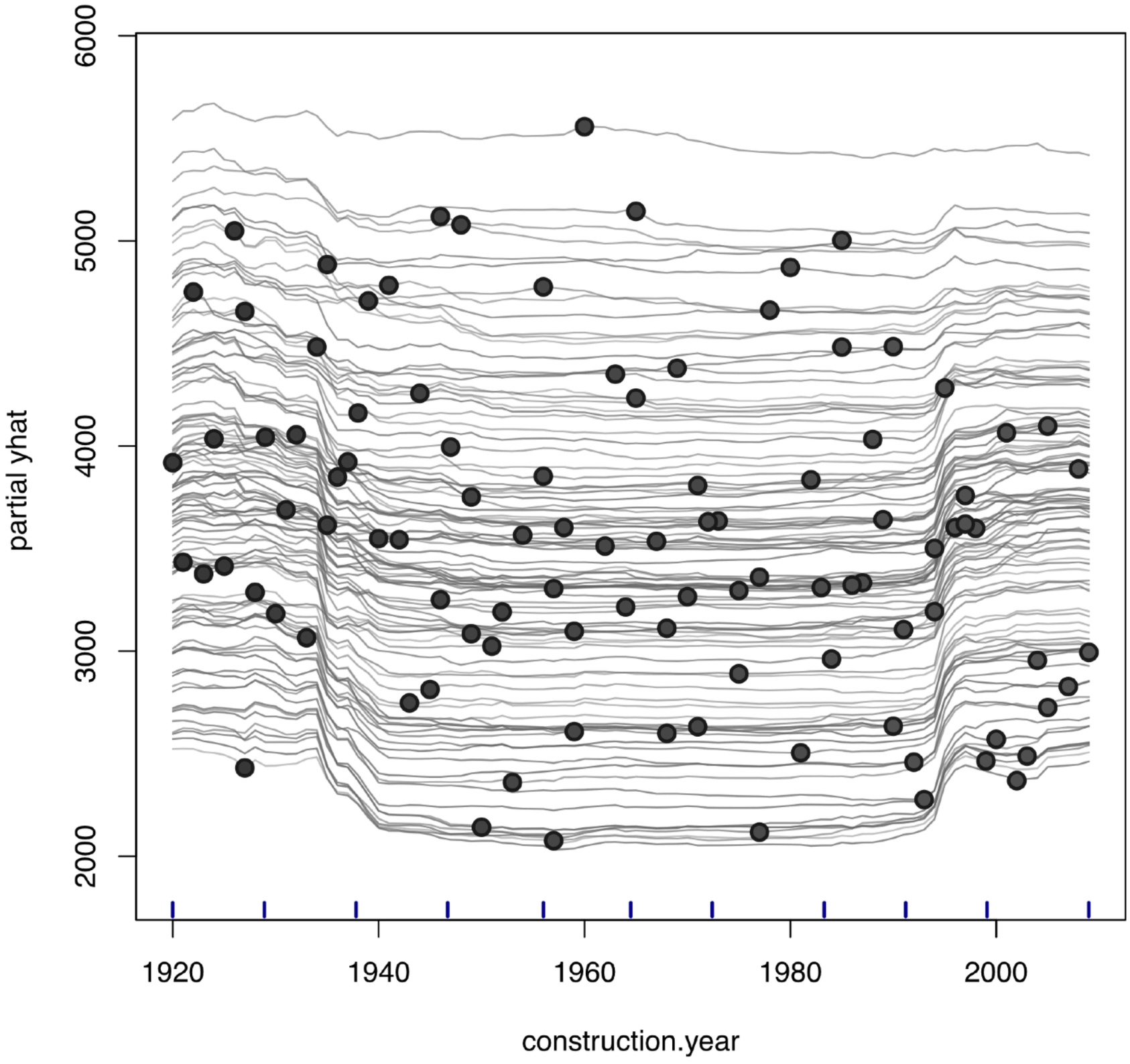


How a single variable affects the response?



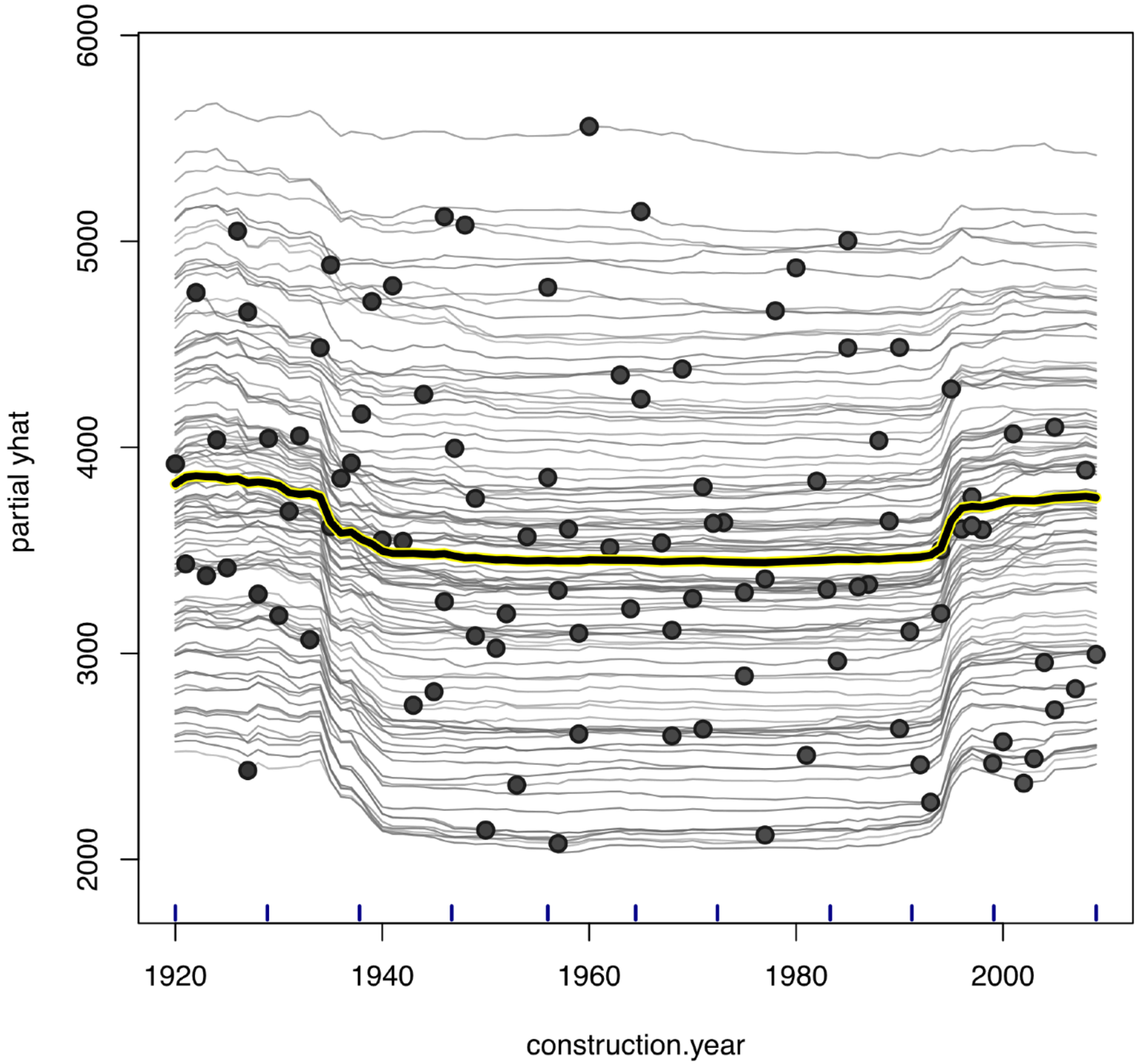
Predictions
vs
Single variable

How a single variable affects the response?



Individual
Conditional
Expectations

How a single variable affects the response?



Partial Dependence Plot

How a single variable affects the response?

Formally, partial dependence may be defined as

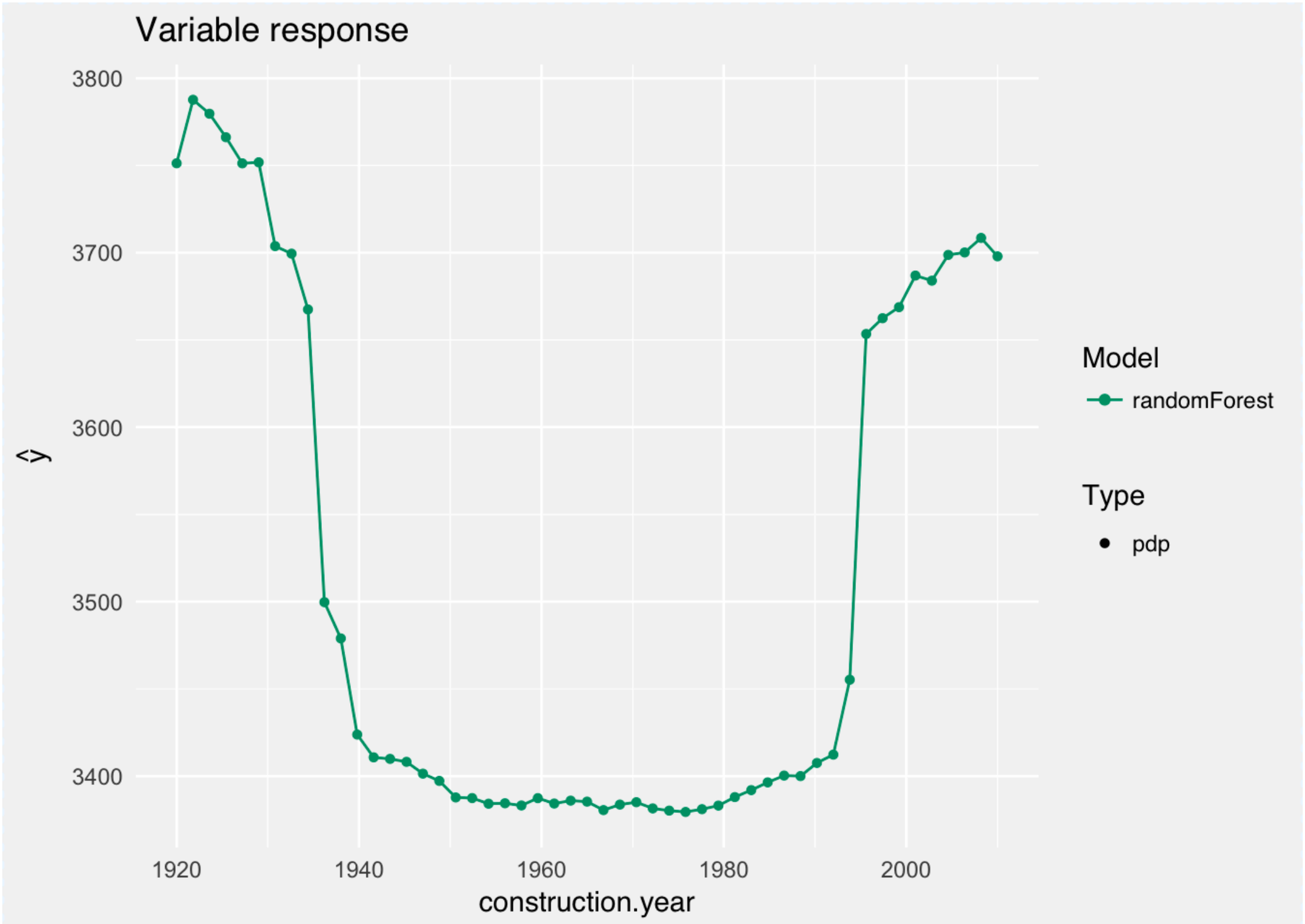
$$p_i(x_i) = E_{x_{-i}}[f(x^i, x^{-i}; \theta)]$$

and estimated as

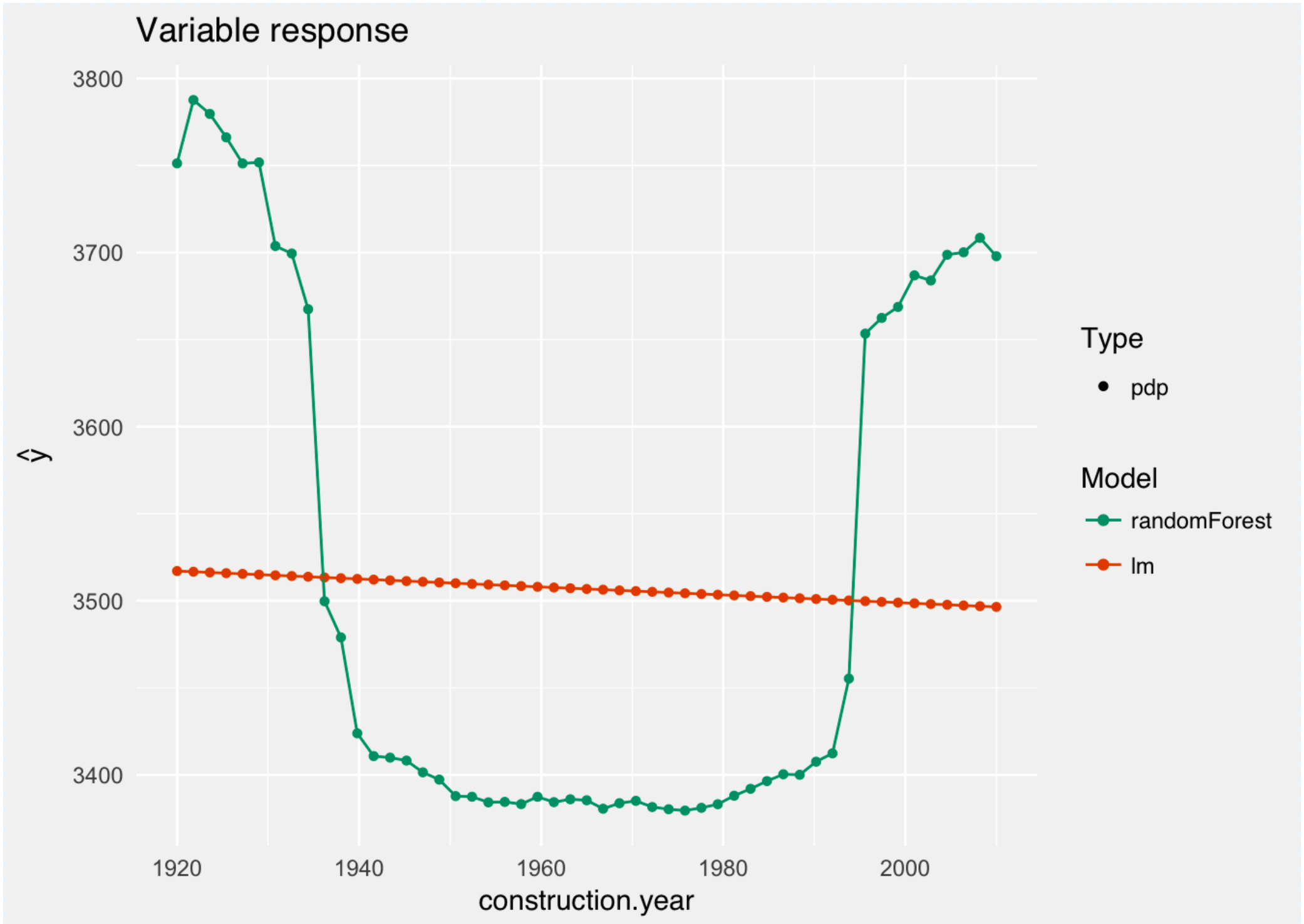
$$\hat{p}_i(x_i) = \frac{1}{n} \sum_{j=1}^{n} f(x_j^i, x_j^{-i}, \hat{\theta})$$

pdp: An R Package for Constructing Partial Dependence Plots.
Brandon M. Greenwell (2017)
https://journal.r-project.org/archive/2017/RJ-2017-016/index.html

```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)
```



No Silver Bullet

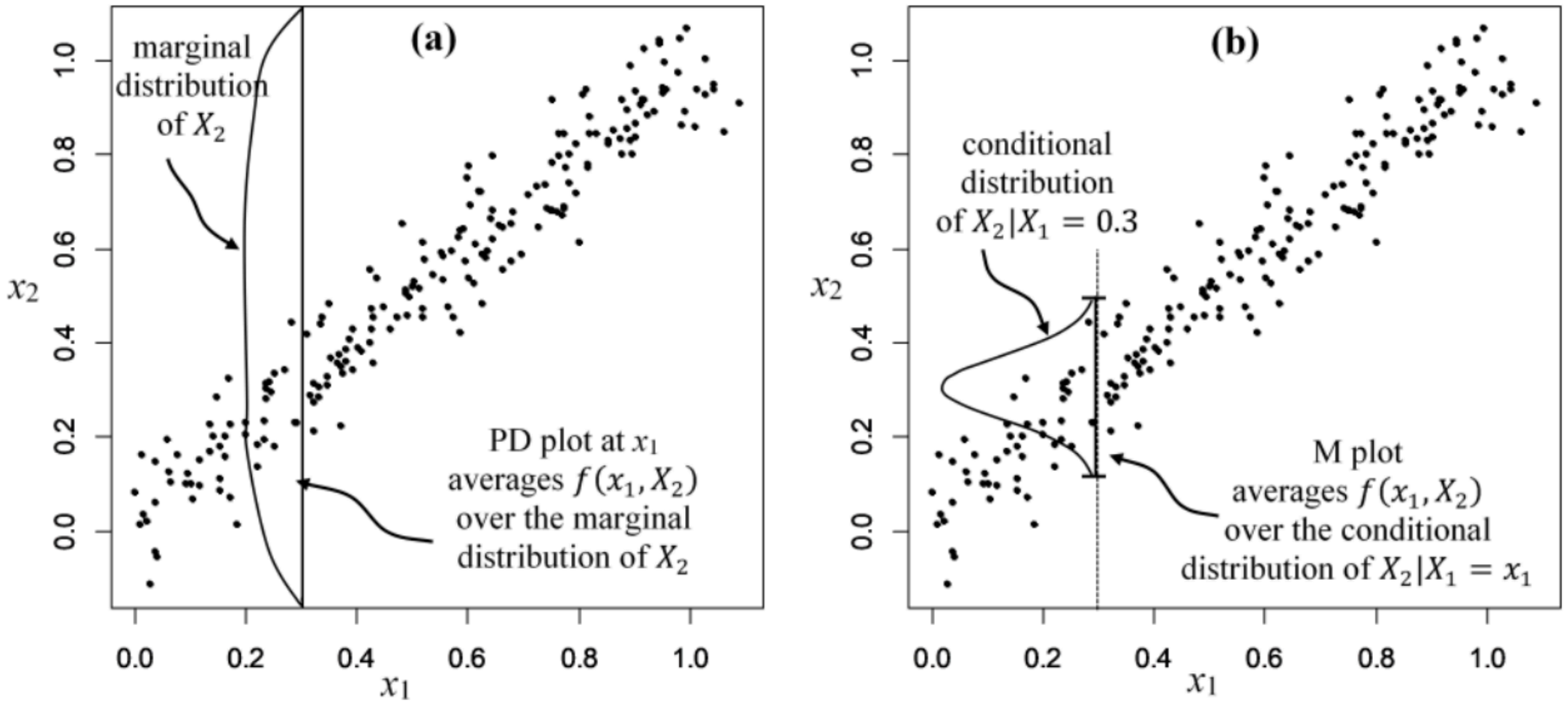
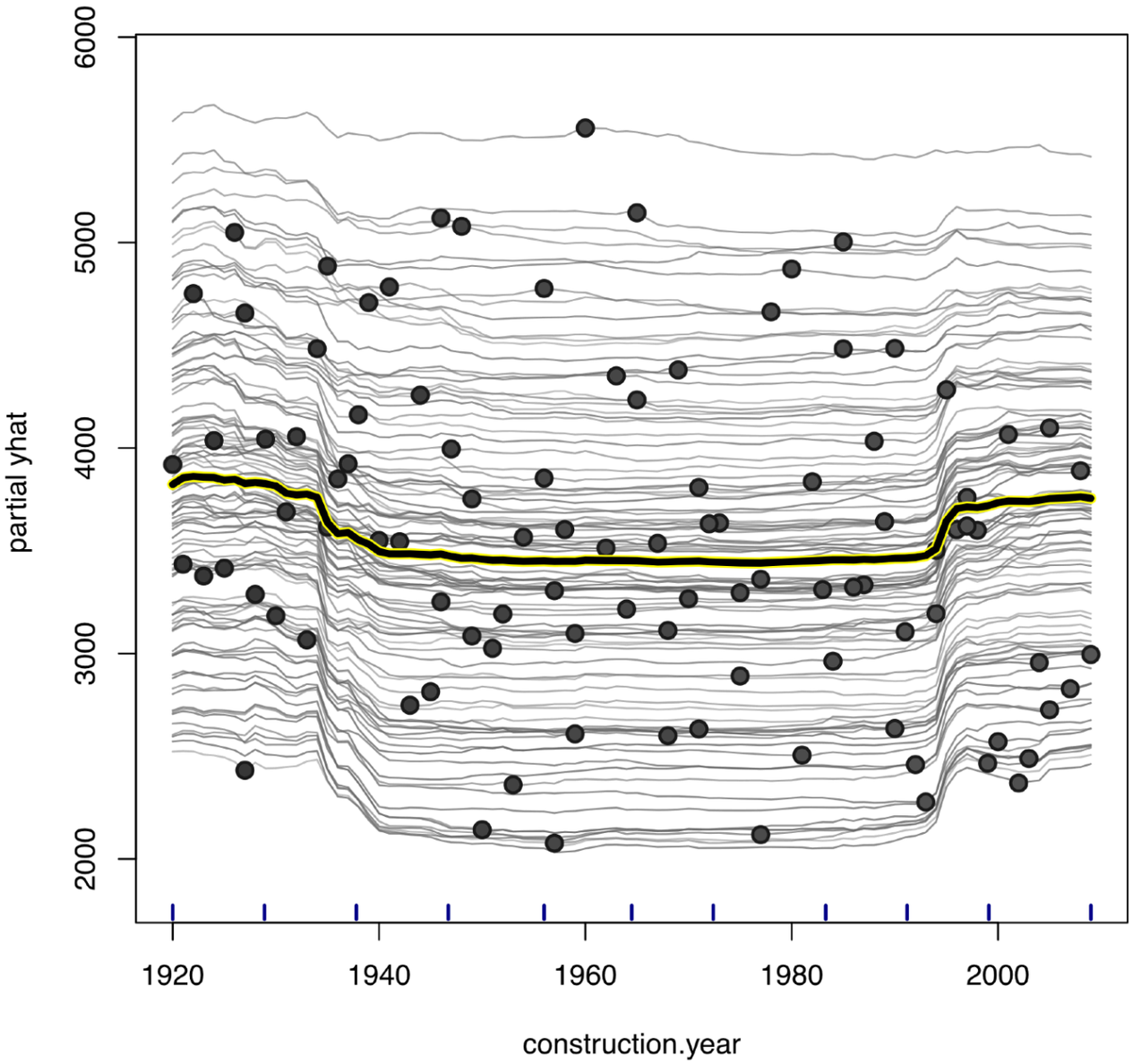
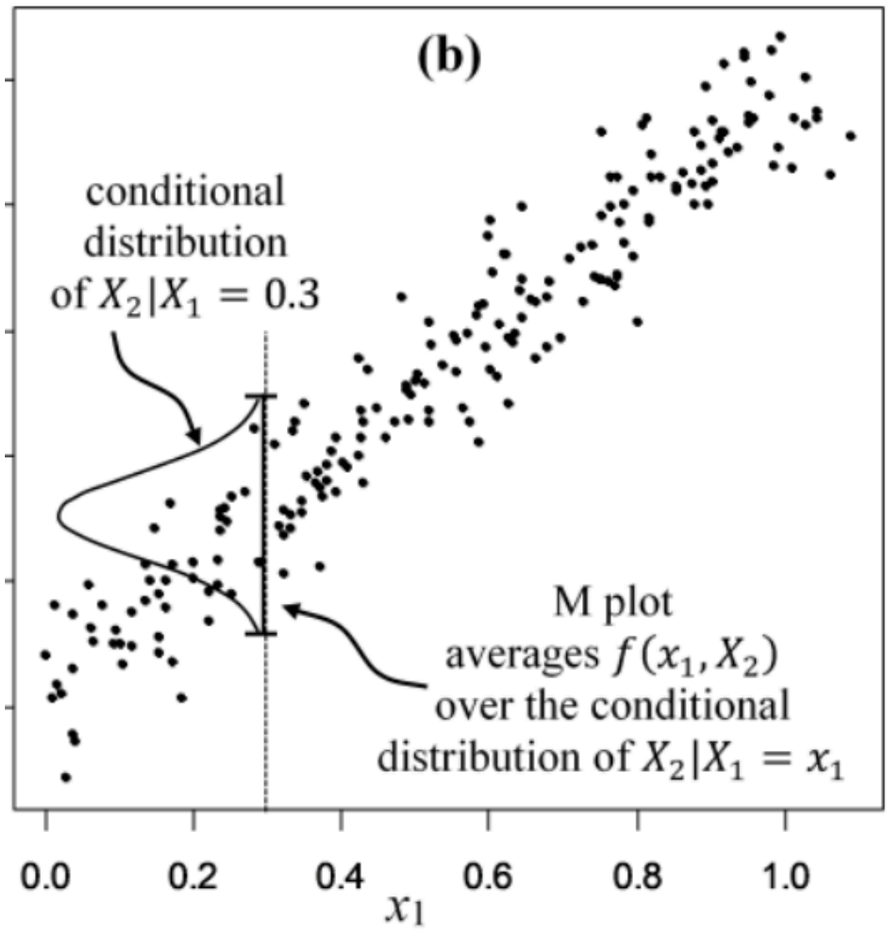


Figure 1: Illustration of the differences between the computation of (a) $f_{1,PD}(x_1)$ and (b) $f_{1,M}(x_1)$ at $x_1 = 0.3$.

From: https://cran.r-project.org/web/packages/ALEPlot/vignettes/AccumulatedLocalEffectPlot.pdf

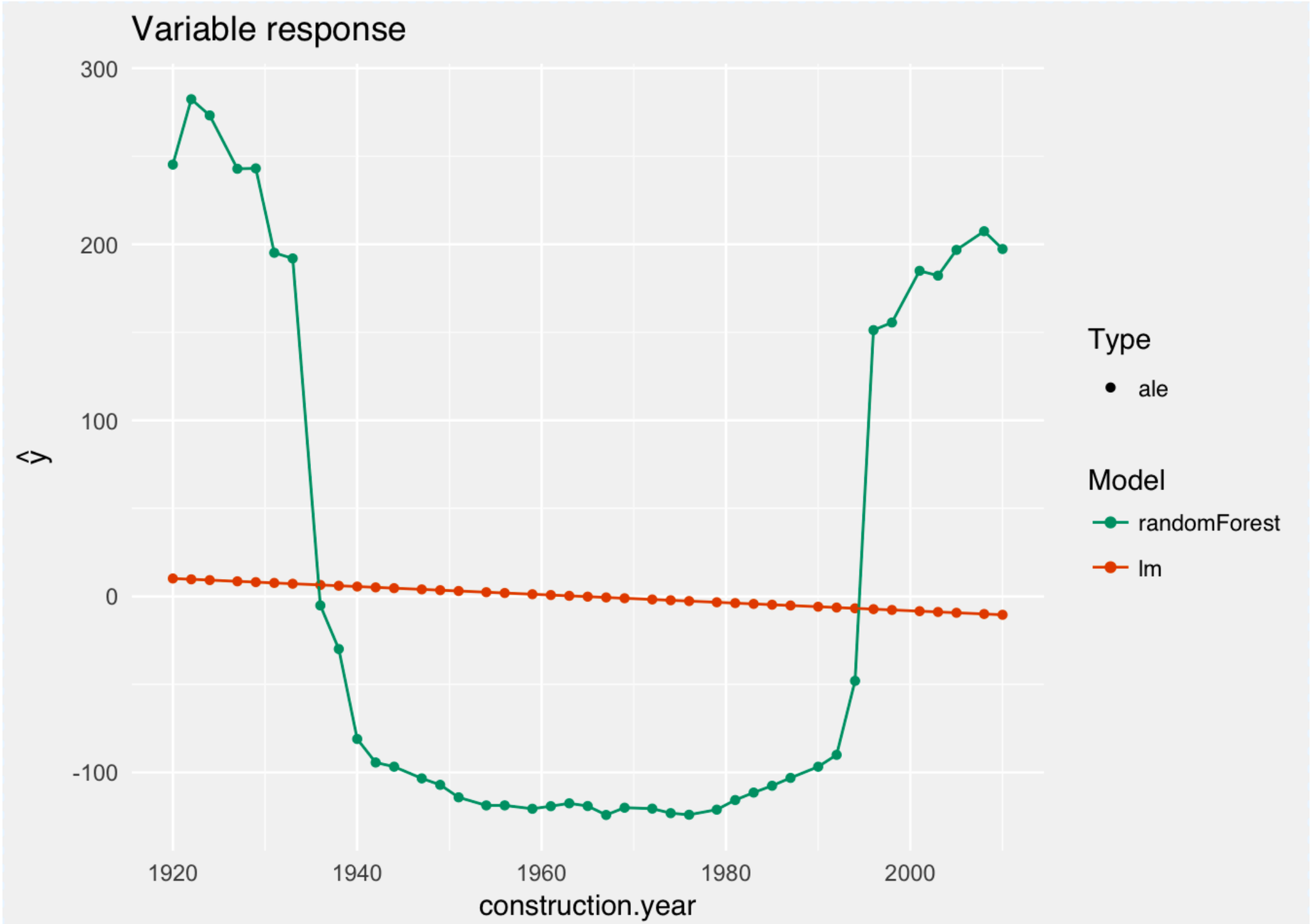
No Silver Bullet





From: https://cran.r-project.org/web/packages/ALEPlot/vignettes/AccumulatedLocalEffectPlot.pdf

```r
sva_rf  <- single_variable(explainer_rf, variable = "construction.year", type = "ale")
sva_lm  <- single_variable(explainer_lm, variable = "construction.year", type = "ale")

plot(sva_rf, sva_lm)
```



Find more in:

Package pdp
Partial Dependence Plots, Greenwell 2017

Package ALEPlot
Accumulated Local Effects Plots, Apley 2017

Package ICEbox
Individual Conditional Expectations, Goldstein , Kapelner, Bleich, Pitkin 2015

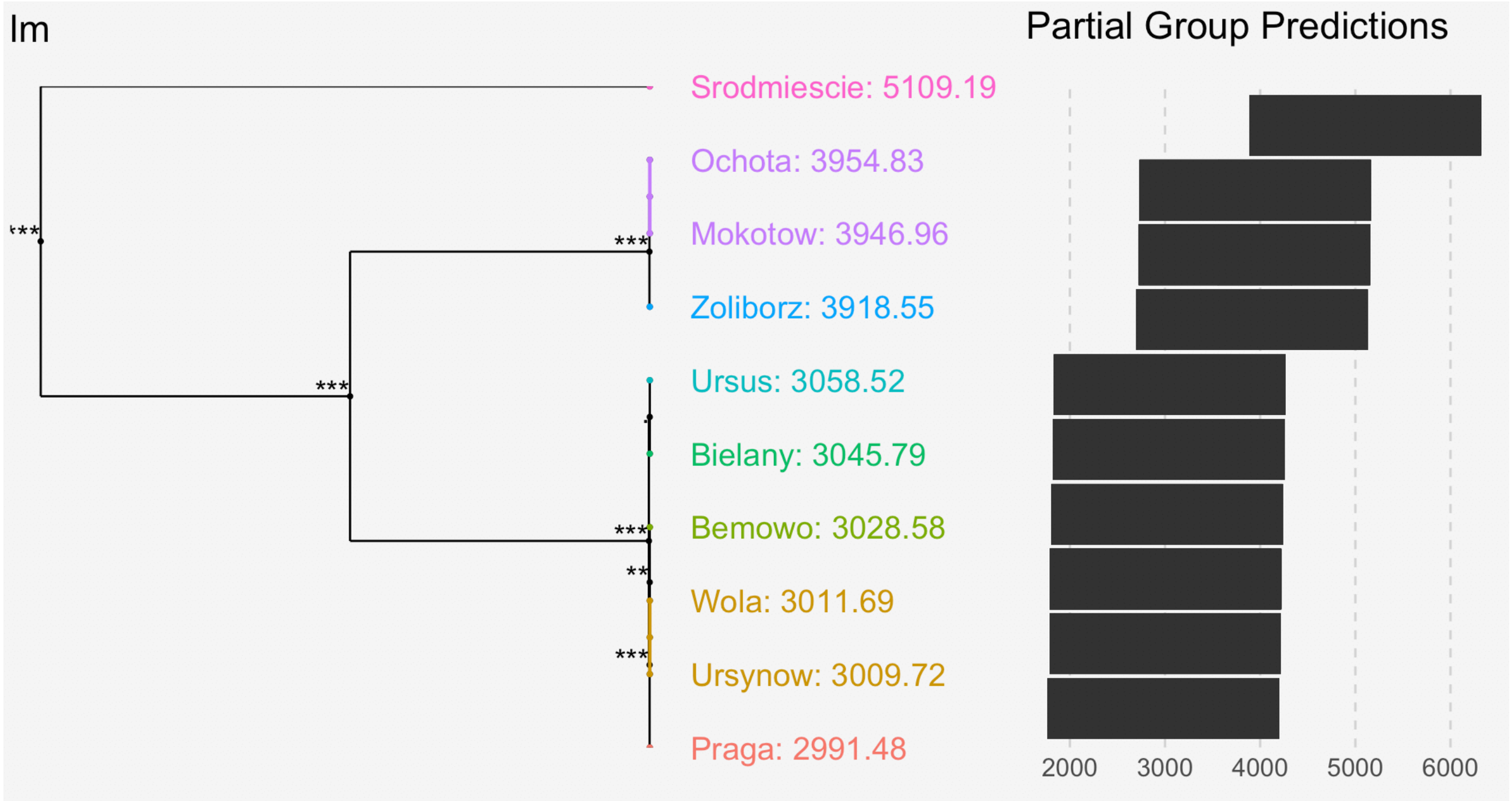
Model explainers
Categorical variable

How a single feature affects the model response?

What to do with factors?

- Partial Dependency Plots are designed for continues variables.
- For categorical variables we would like to see how the model response is affected by a particular level of categorical variable.
- To show this we are using The Merging Path Plots, as described in Sitko, Biecek (2017) https://arxiv.org/pdf/1709.04412.pdf
- Similarity of factors are shown with a dendrogram. Height of splits corresponds to log likelihood for a model with combined levels.

How a categorical variable affects the response?



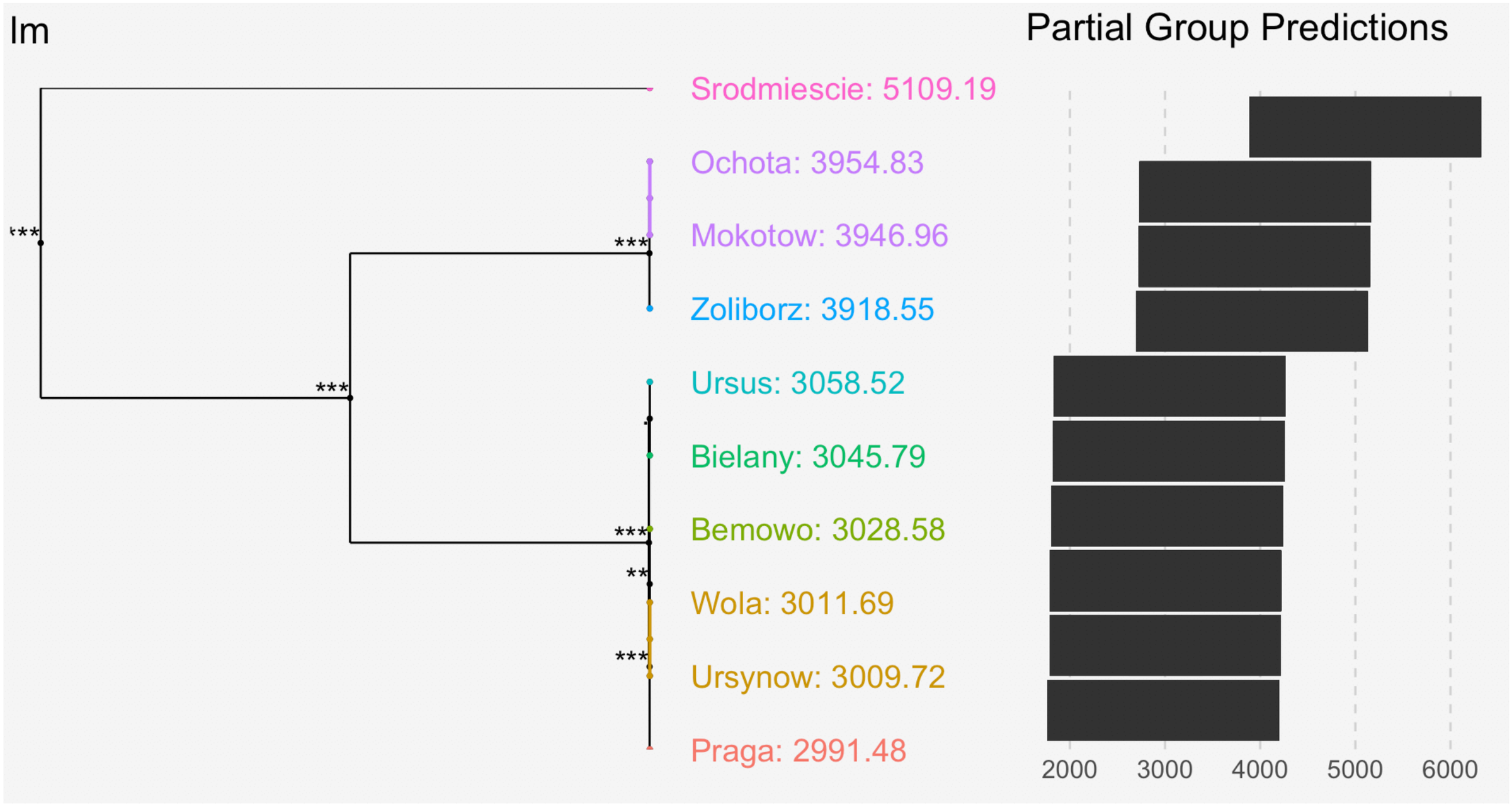
In details

Algorithm 1 The outline of the Merging Path Plot algorithm implemented in **factorMerger**

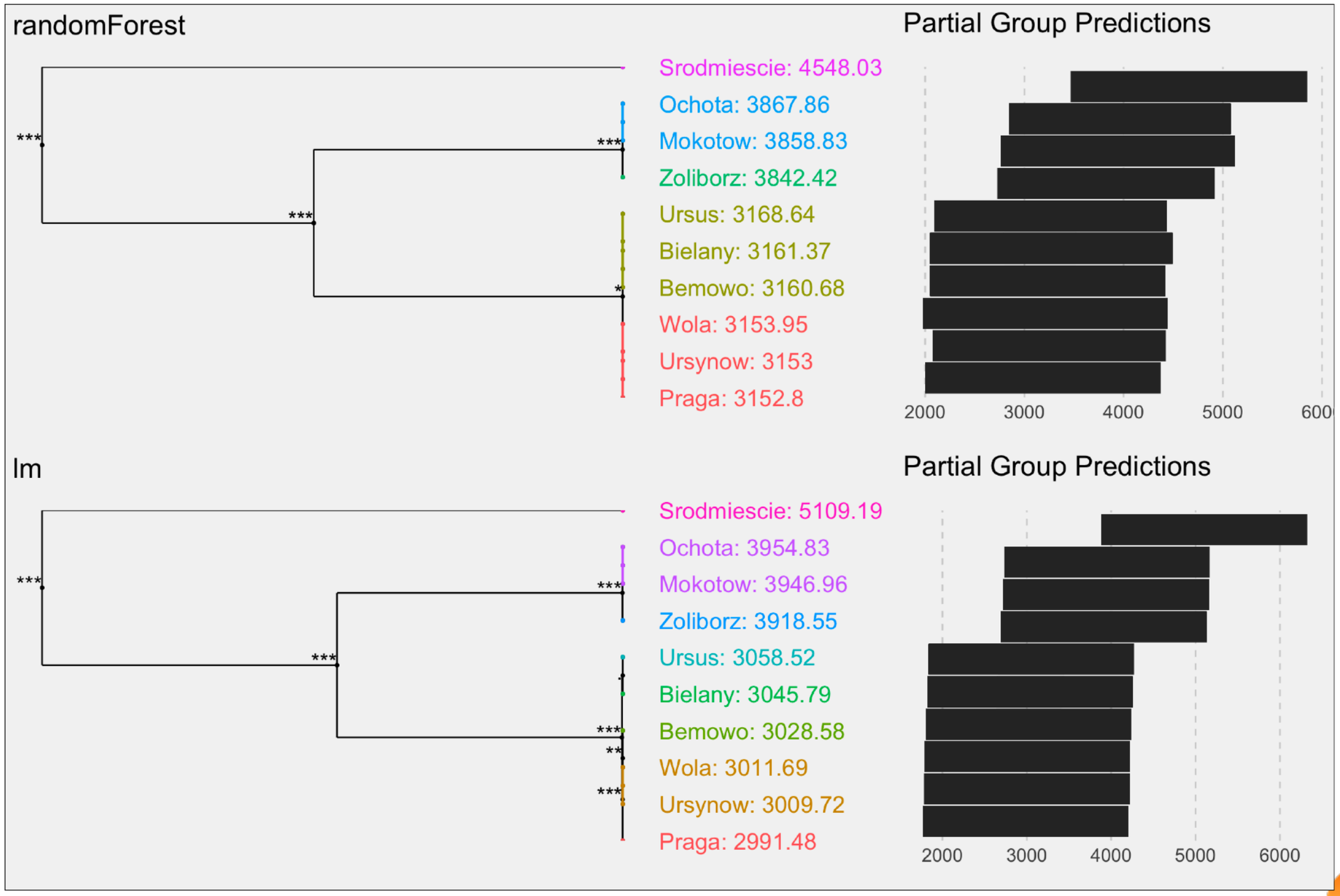
function MERGEFACTORS($responseVariable, groupingVariable, adjacent$)
  $currentModel := createModel(responseVariable, groupingVariable)$
  $mergingPath := list(currentModel)$
  while $|levels(groupingVariable)| \geq 1$ **do**
    $pairsSet := generatePairs(groupingVariable, responseVariable, adjacent)$
    $selectedPair := \text{argmax}_{\text{pair} \in \text{pairsSet}} objectiveFunction(pair, responseVariable, groupingVariable)$
    $groupingVariable := mergeLevels(groupingVariable, selectedPair)$
    $currentModel := createModel(responseVariable, groupingVariable)$
    $mergingPath := add(mergingPath, currentModel)$
  end while
  return(mergingPath)
end function

The Merging Path Plot: adaptive fusing of k-groups with likelihood-based model selection
Agnieszka Sitko, Przemysław Biecek (2017)
https://arxiv.org/abs/1709.04412
https://github.com/geneticsMiNIng/FactorMerger

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

plot(svd_lm)
```

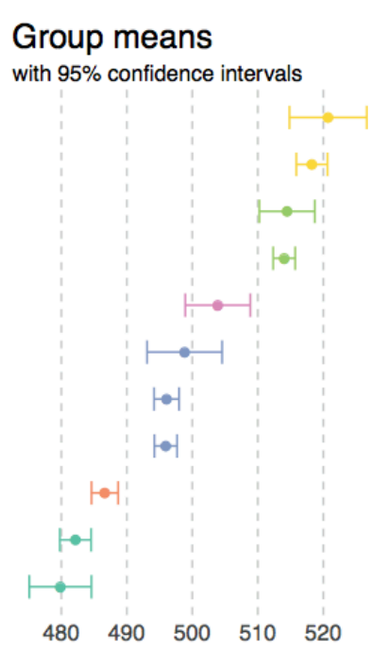


```r
svd_rf  <- single_variable(explainer_rf, variable = "district", type = "factor")
svd_lm  <- single_variable(explainer_lm, variable = "district", type = "factor")

plot(svd_rf, svd_lm)
```

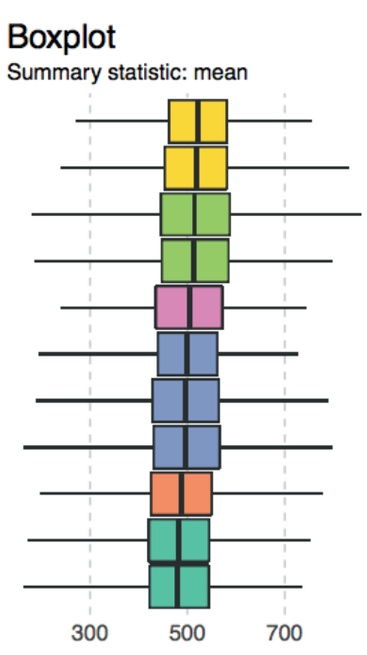


Other panels implemented in factorMerger

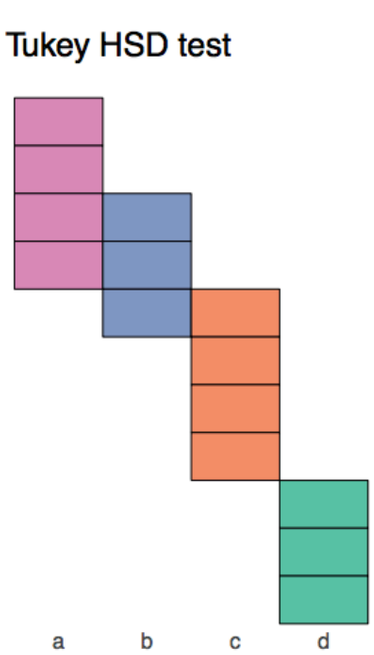
responsePanel = "means"



responsePanel = "boxplot"

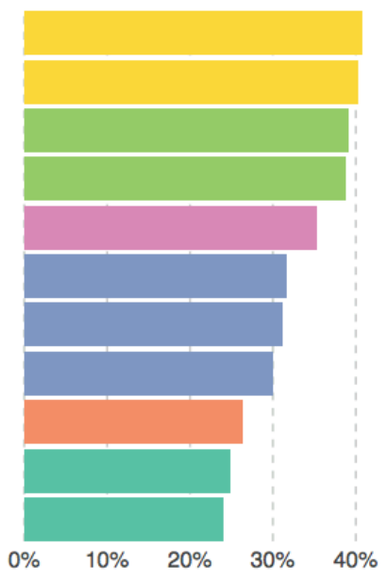


responsePanel = "tukey"

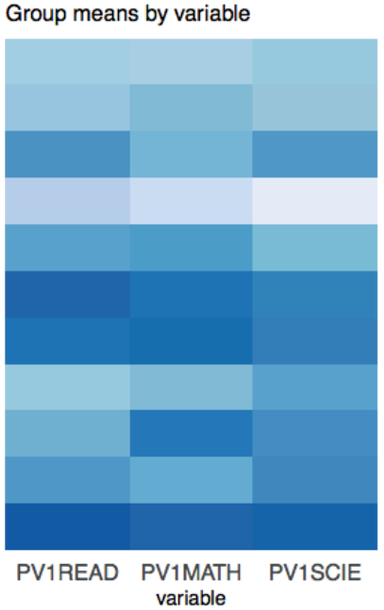


responsePanel = "frequency"

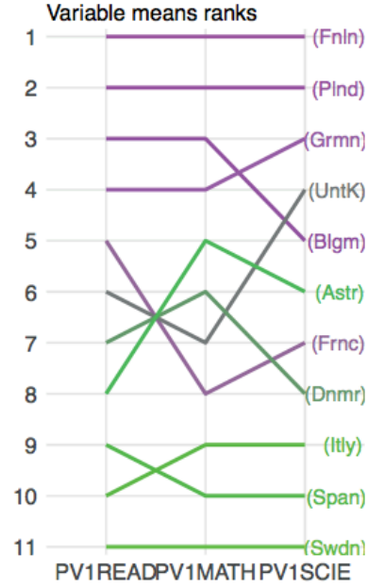
responsePanel = "proportion"



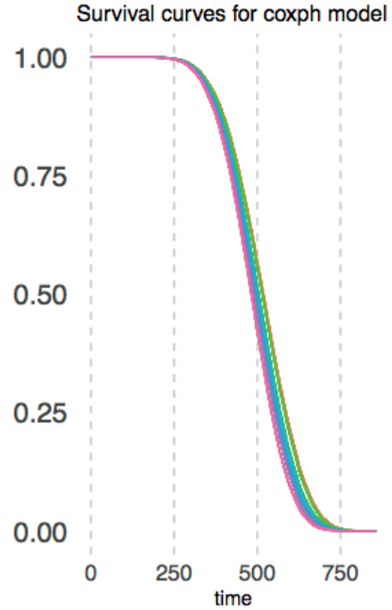
responsePanel = "heatmap"



responsePanel = "profile"



responsePanel = "survival"



Developer:
Agnieszka Sitko

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

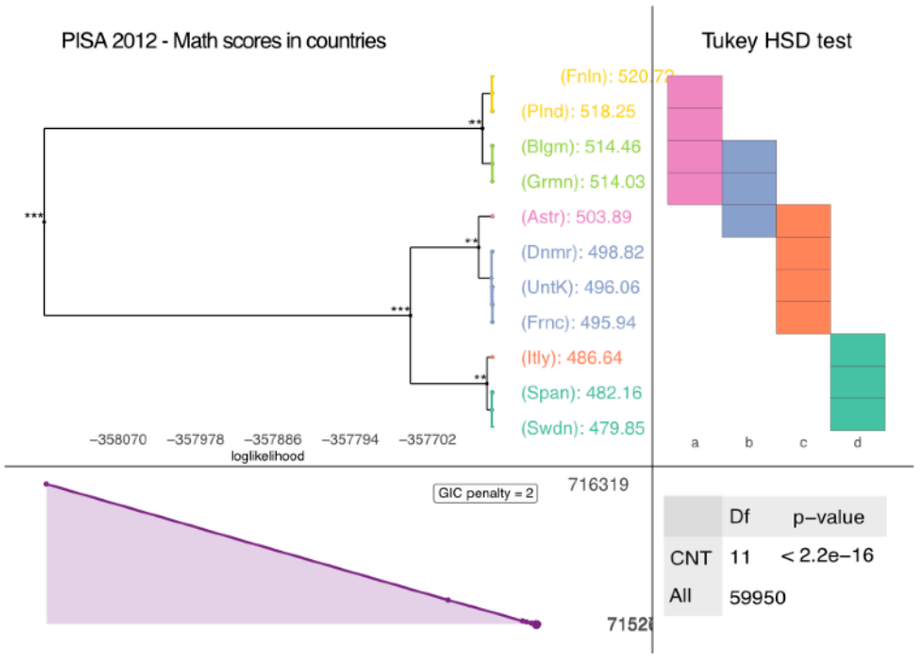
```
library(factorMerger)

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

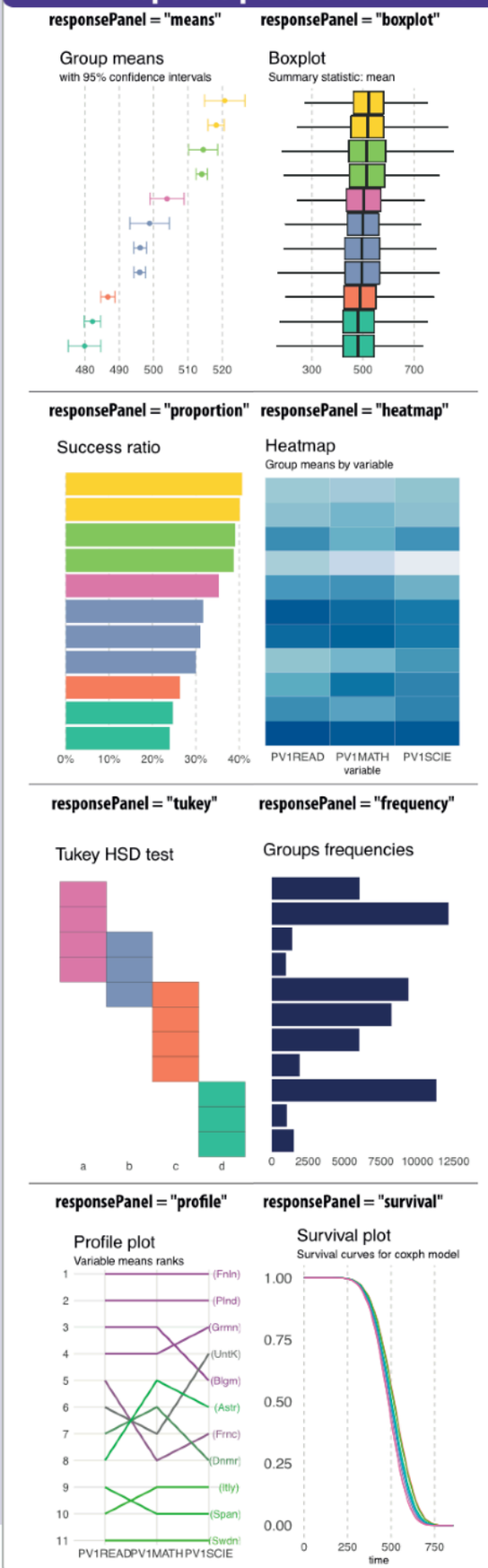
The **method** argument determines how pairs of groups are considered for grouping. The most computational demanding is *adaptive*. For larger number of groups one may consider only pairs of groups with closest averages or use the *fast-fixed* heuristic (fastest solution).

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



Your turn! (approx 15 minutes)

1. Load the DALEX package. Summarise the apartments data. What variables do we have there?
2. Create some predictive models for the m2.price variable. Try
   - linear model (lm),
   - Random Forest model (randomForest),
   - Support Vector Machines model (e1071::svm)
3. Create wrappers for these models explain() function
4. Plot PDP curves for variables surface and construction.year . How different models handle these variables?
5. Plot FMP dendrograms for the district variable. Are there any differences between these models?

Model explainers
Performance

How accurate is the model?

Accuracy as a single number is not enough!

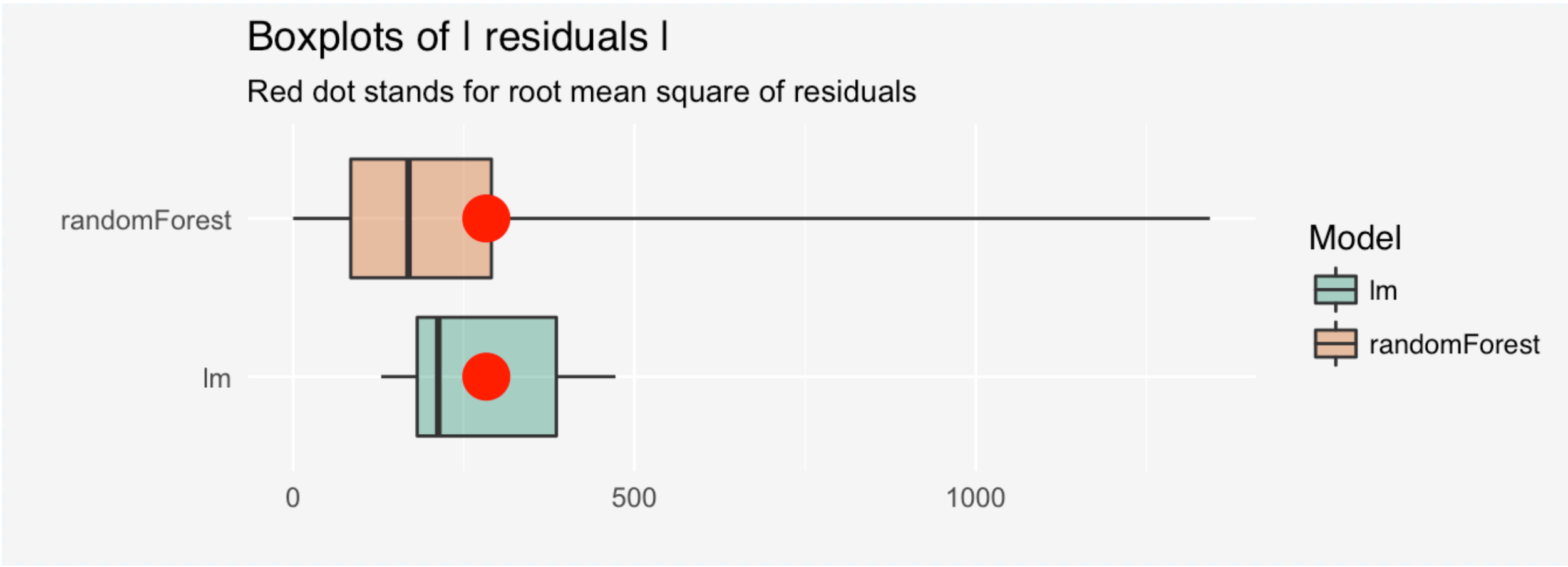
It is common to select a model based on a single criteria, like an accuracy calculated on a test set or with cross-validation.

But what to do in such situation:

```
# 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
```

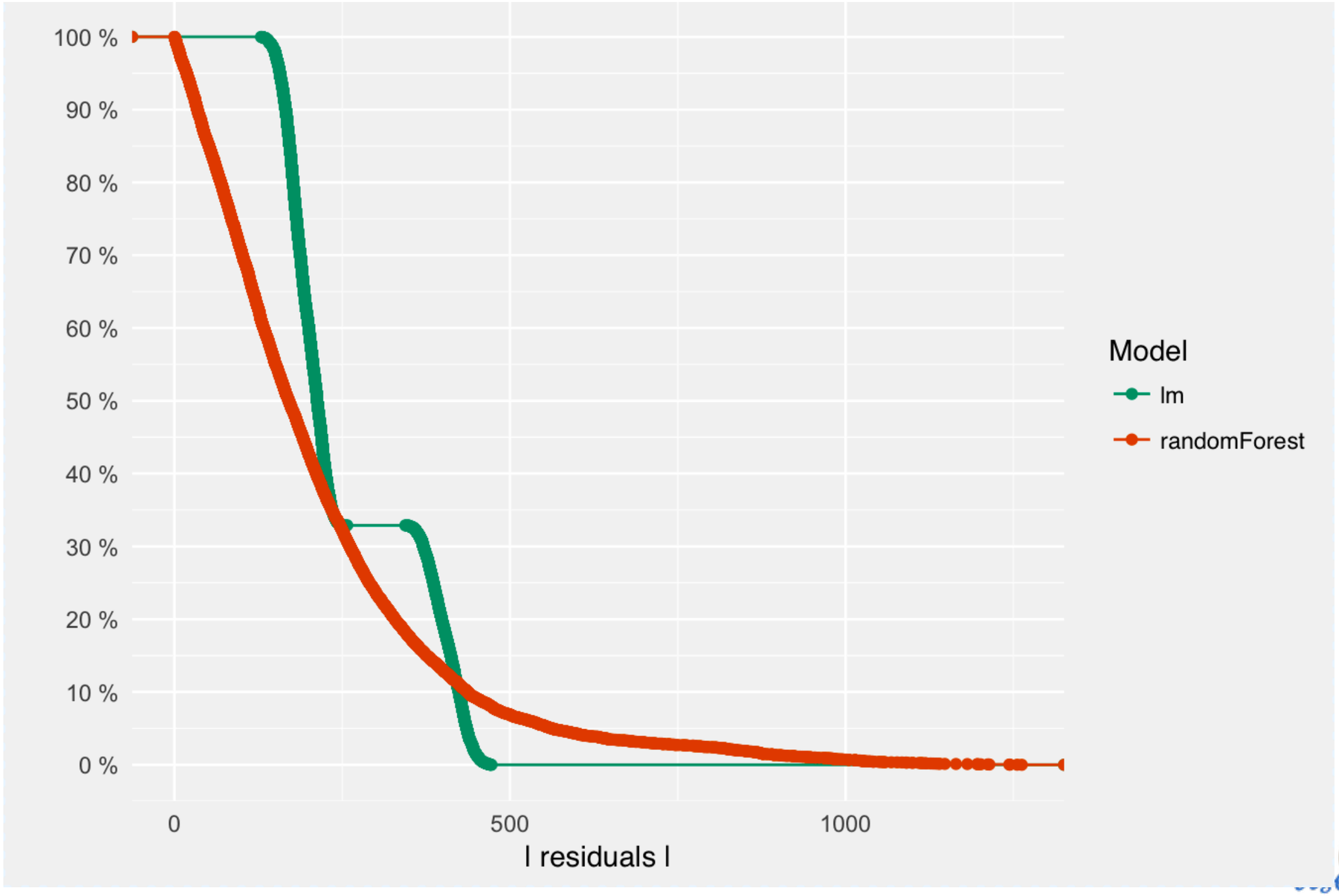
Accuracy as a single number is not enough!

```
mp_svm <- model_performance(explainer_svm)
mp_rf <- model_performance(explainer_rf)
plot(mp_svm, mp_rf, geom = "boxplot")
```



Accuracy as a single number is not enough!

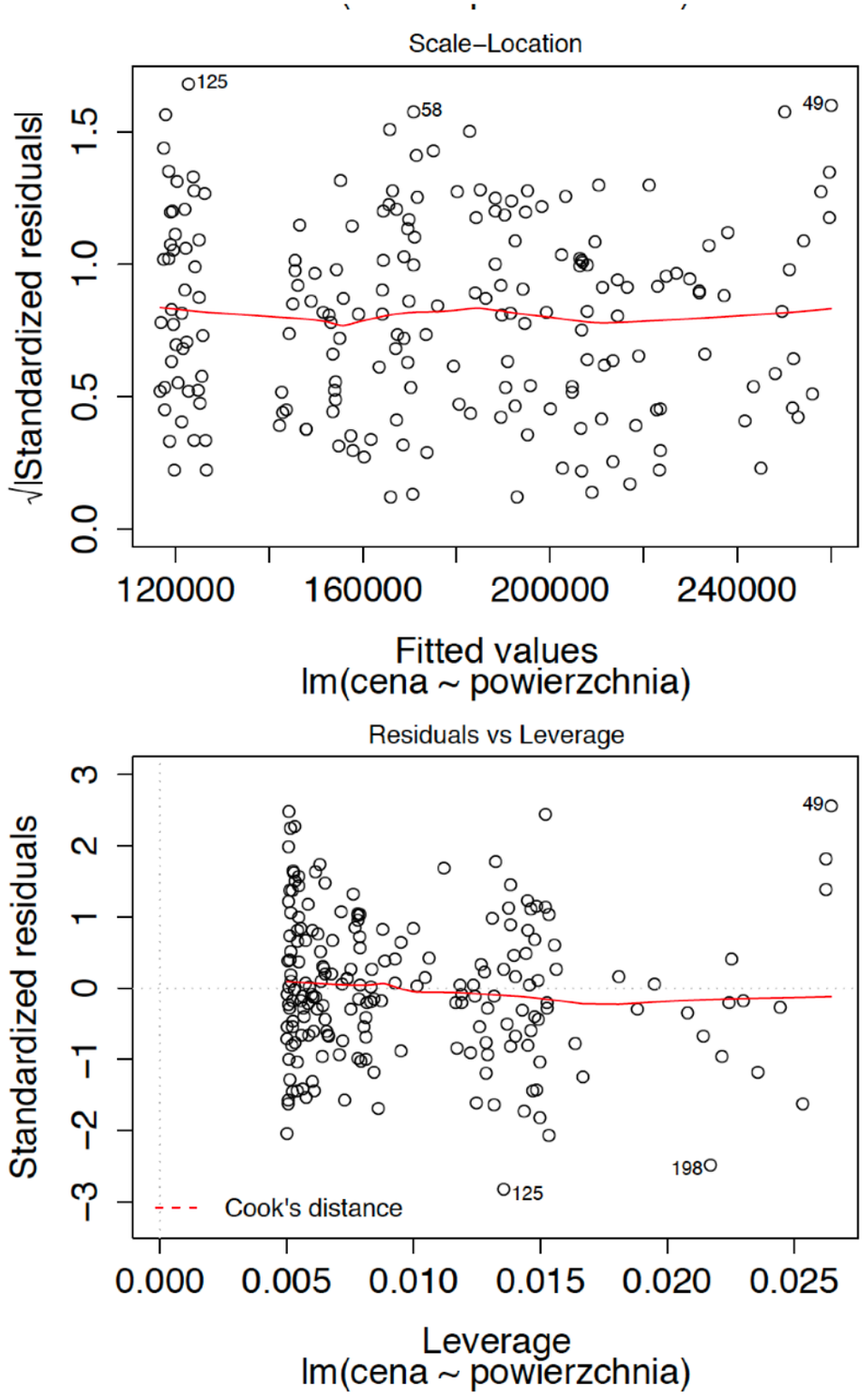
1 - Empirical cumulative distribution function for |residuals|

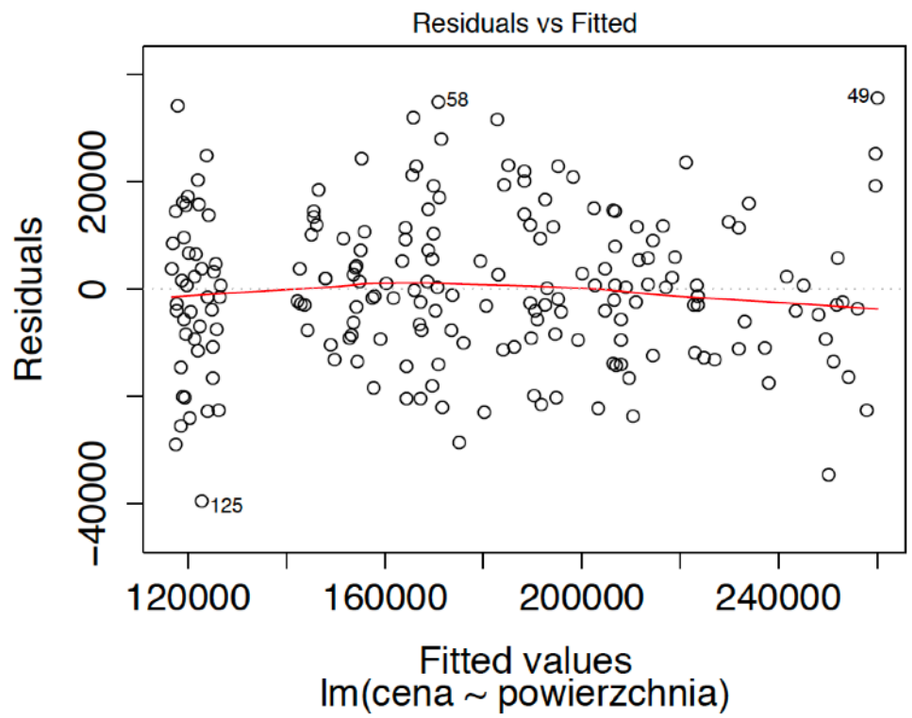


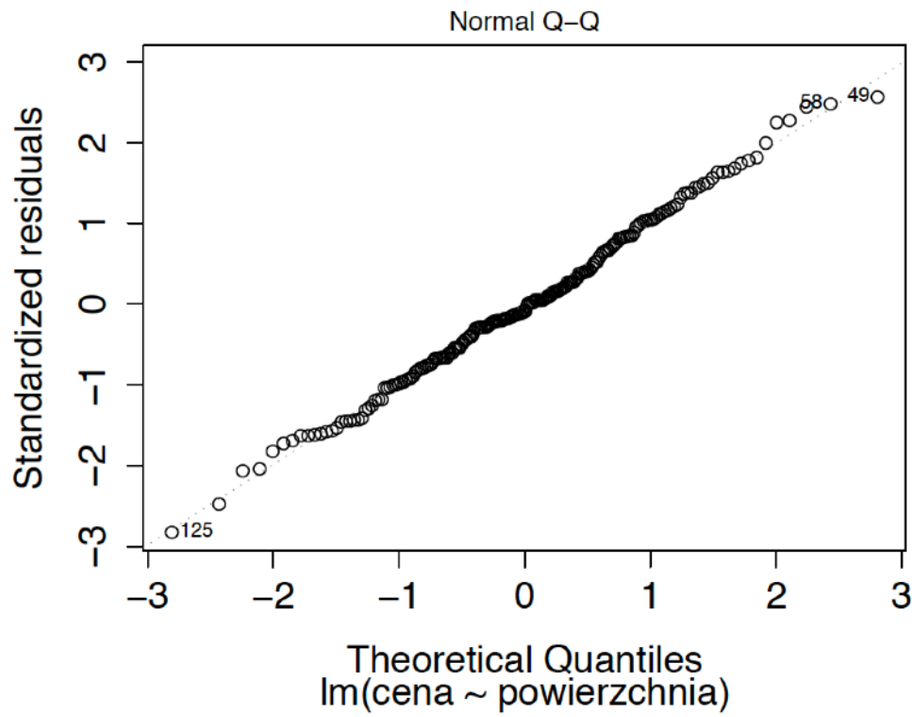
Validation for model residuals

auditor

Do you know these plots?

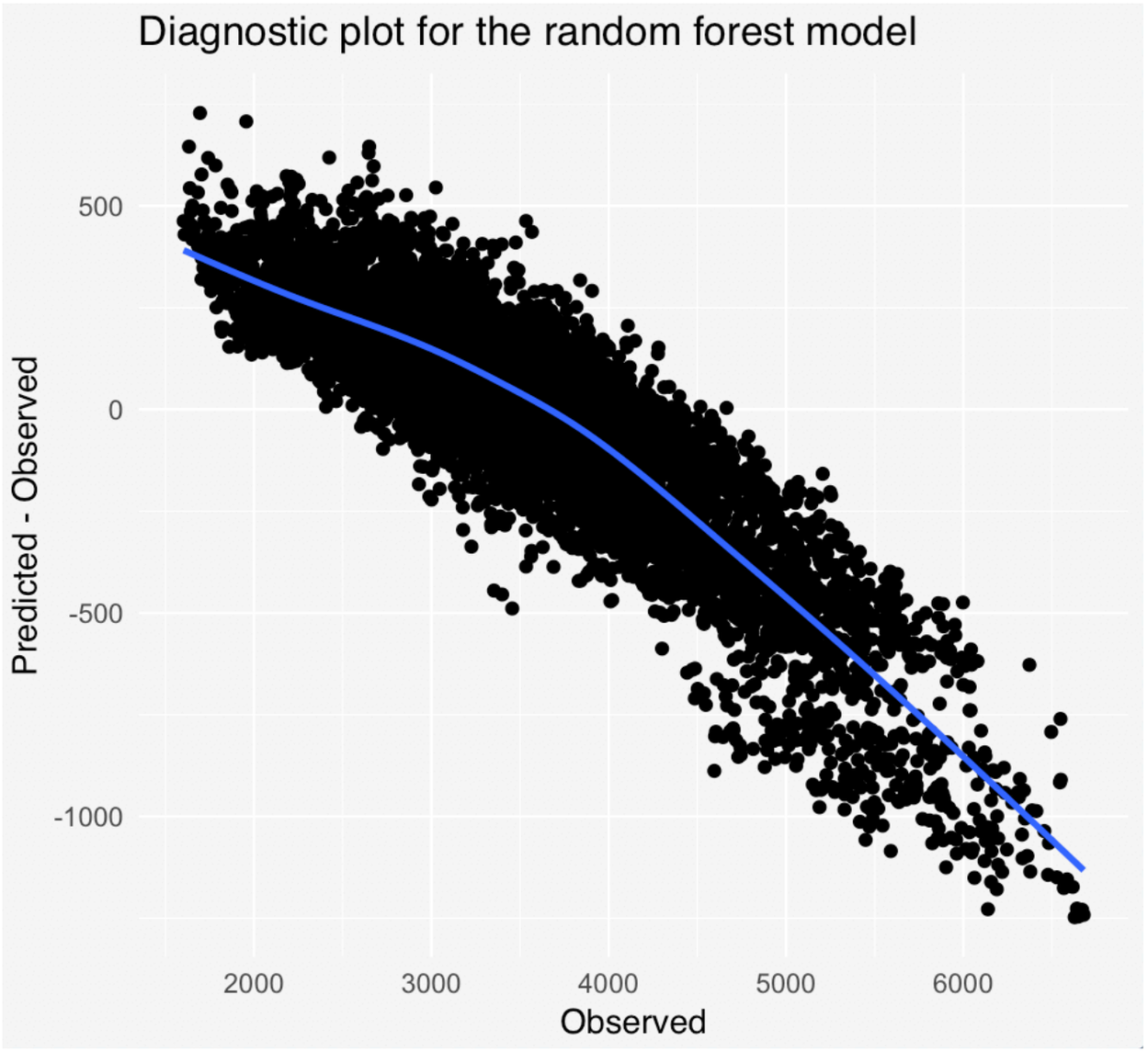


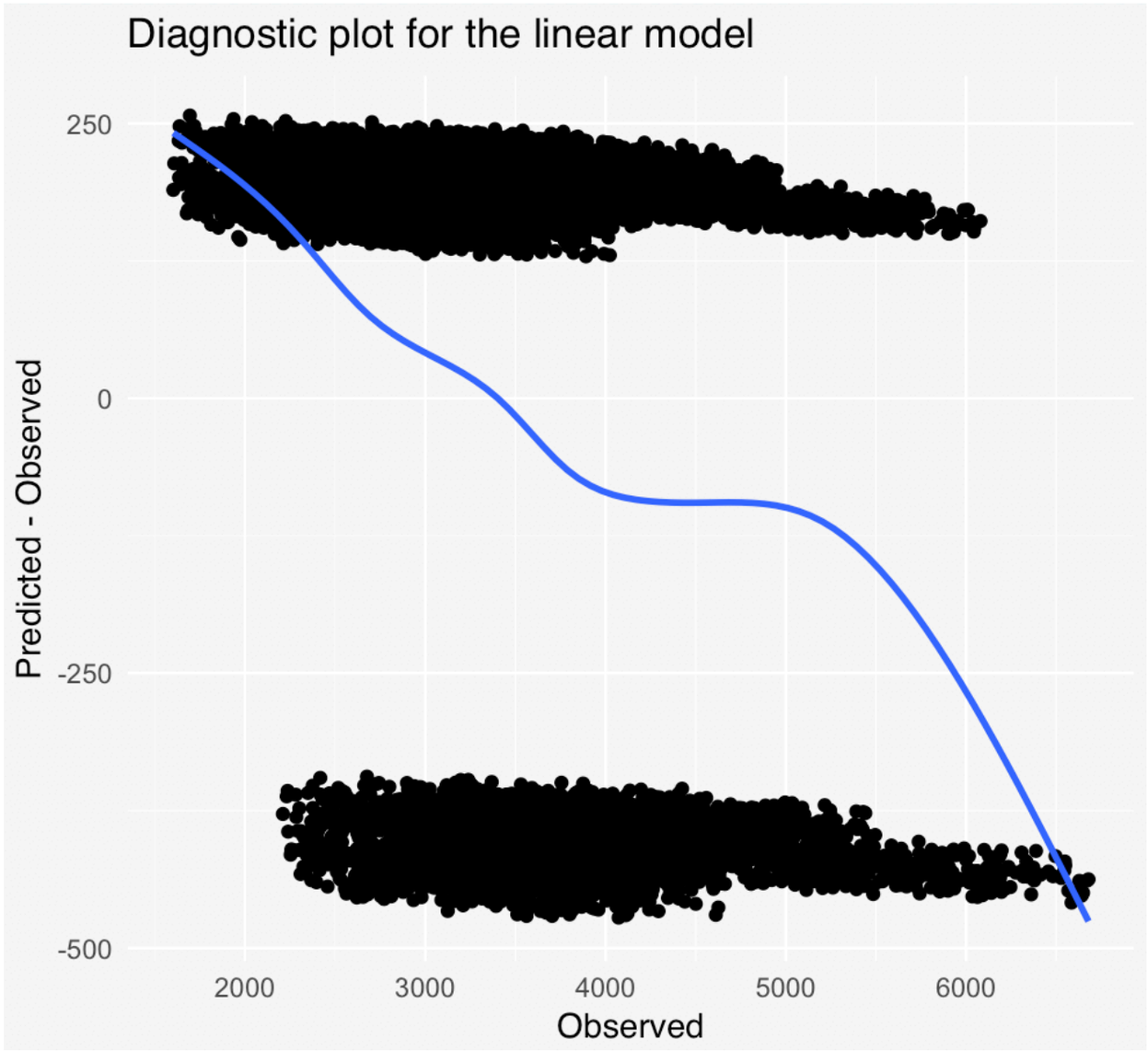




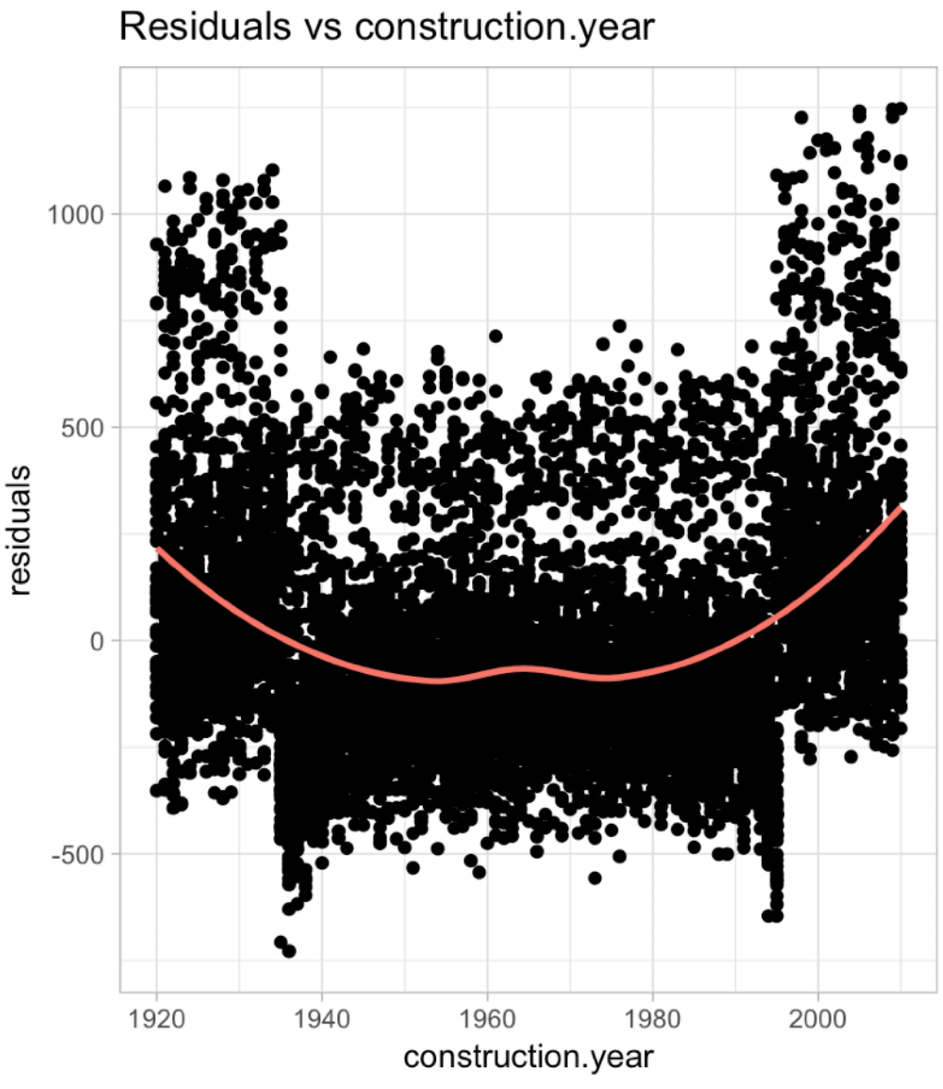
Model agnostic diagnostic tools for residuals

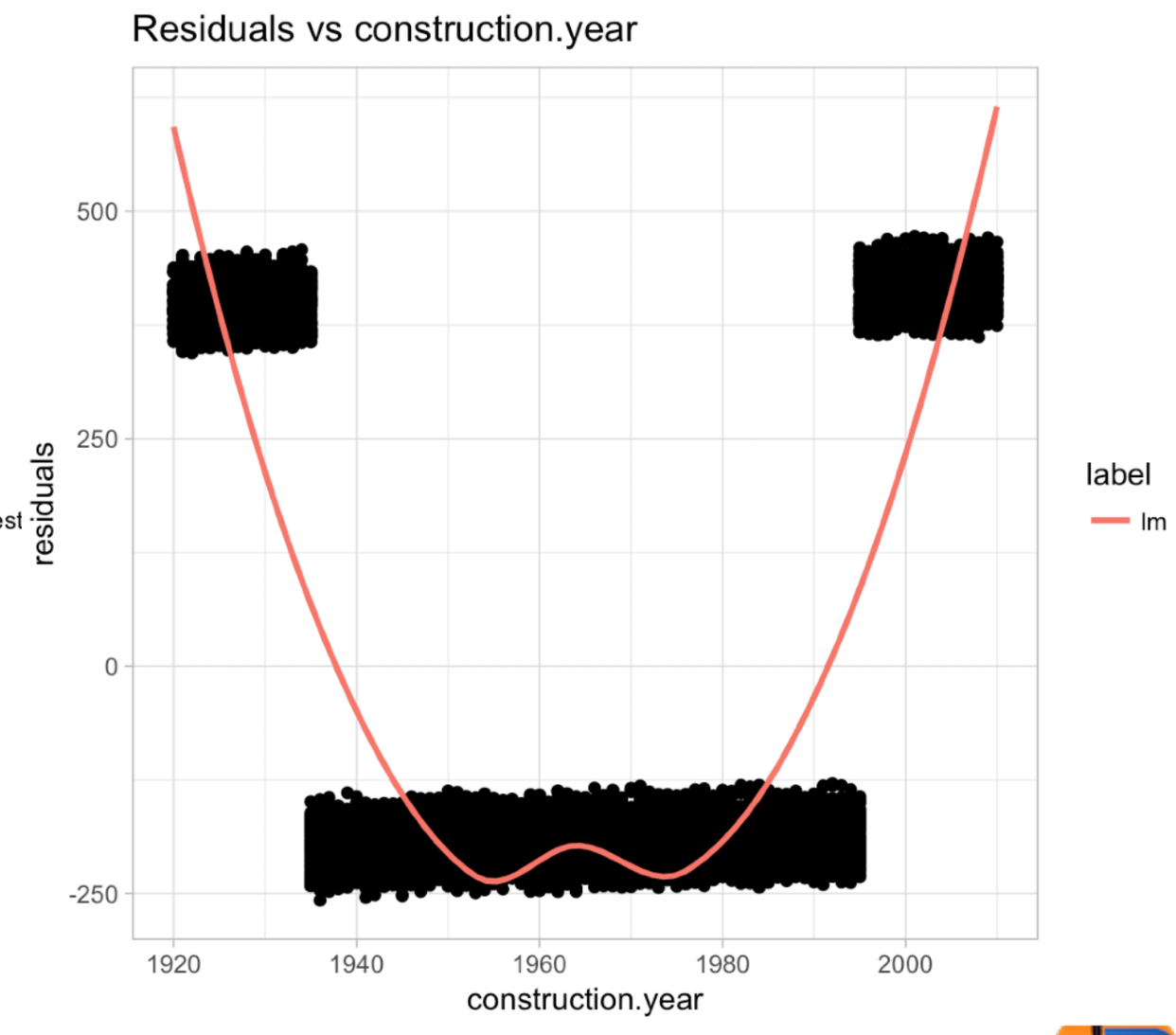
```
mp_rf <- model_performance(explainer_rf)
library("ggplot2")
ggplot(mp_rf, aes(observed, diff)) +
  geom_point()
```





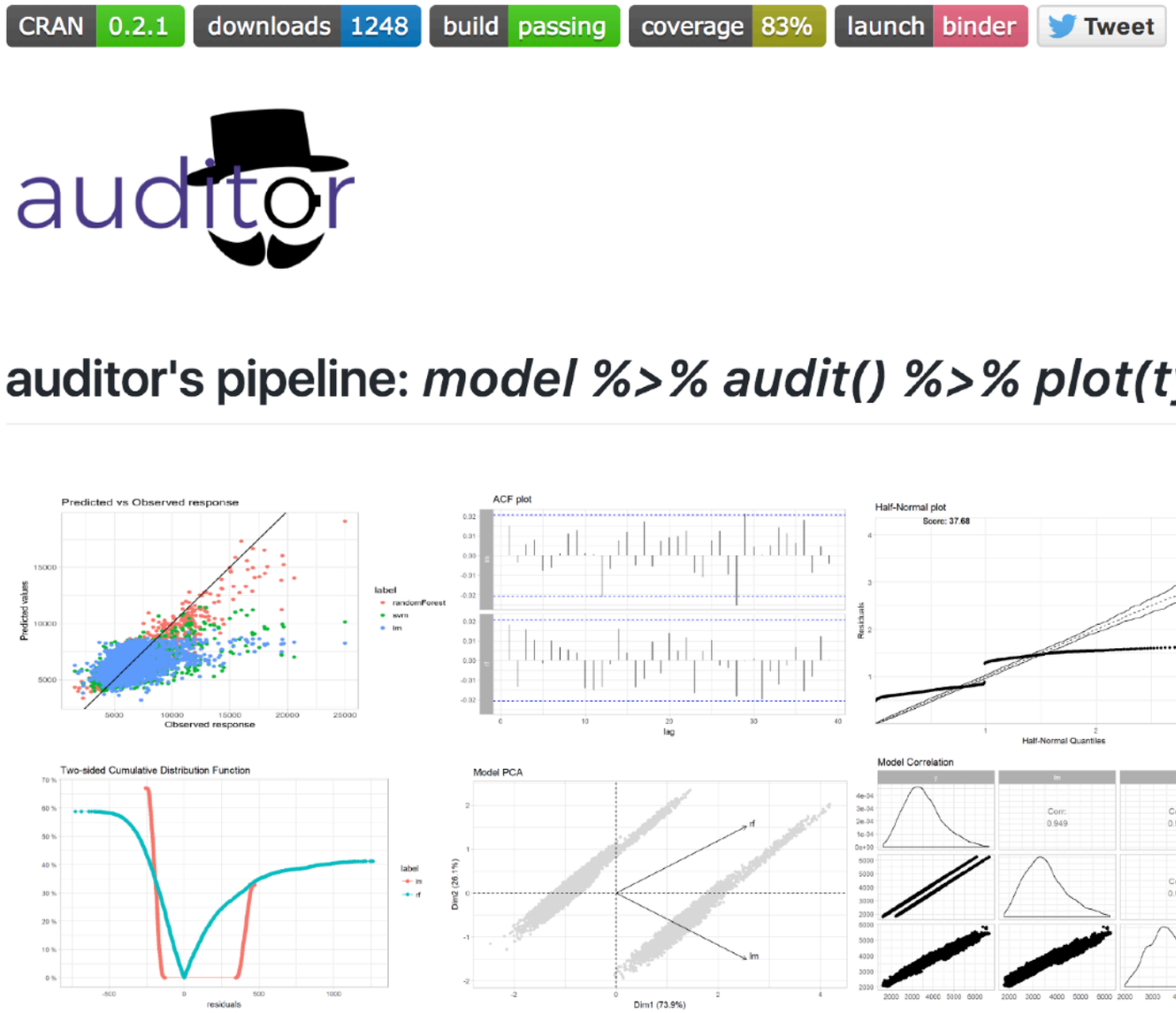
```
library(auditor)
audit_rf <- audit(explainer_rf)
plotResidual(audit_rf, variable = "construction.year")

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





https://github.com/mi2-warsaw/auditor/

The auditor package - model verification, validation, and error analysis



auditor's pipeline: *model %>% audit() %>% plot(type=...)*

Installation

from GitHub

```
devtools::install_github("mi2-warsaw/auditor")
```

Developer:
Alicja Gosiewska

Model explainers
Variable importance

Which variables are influential?

Which variables are important?

- Some models (additive models, random forest, etc) have build-in techniques to evaluate variable importance.

- Some of these concepts bay be applied in a model agnostic fashion (for example see Fisher, Rudin, Dominici (2018)).

- It's a global view on variable importance calculated on test dataset.

Intuition

Calculate loss function for original data vs loss function for data with a single variable permuted/blinded.

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

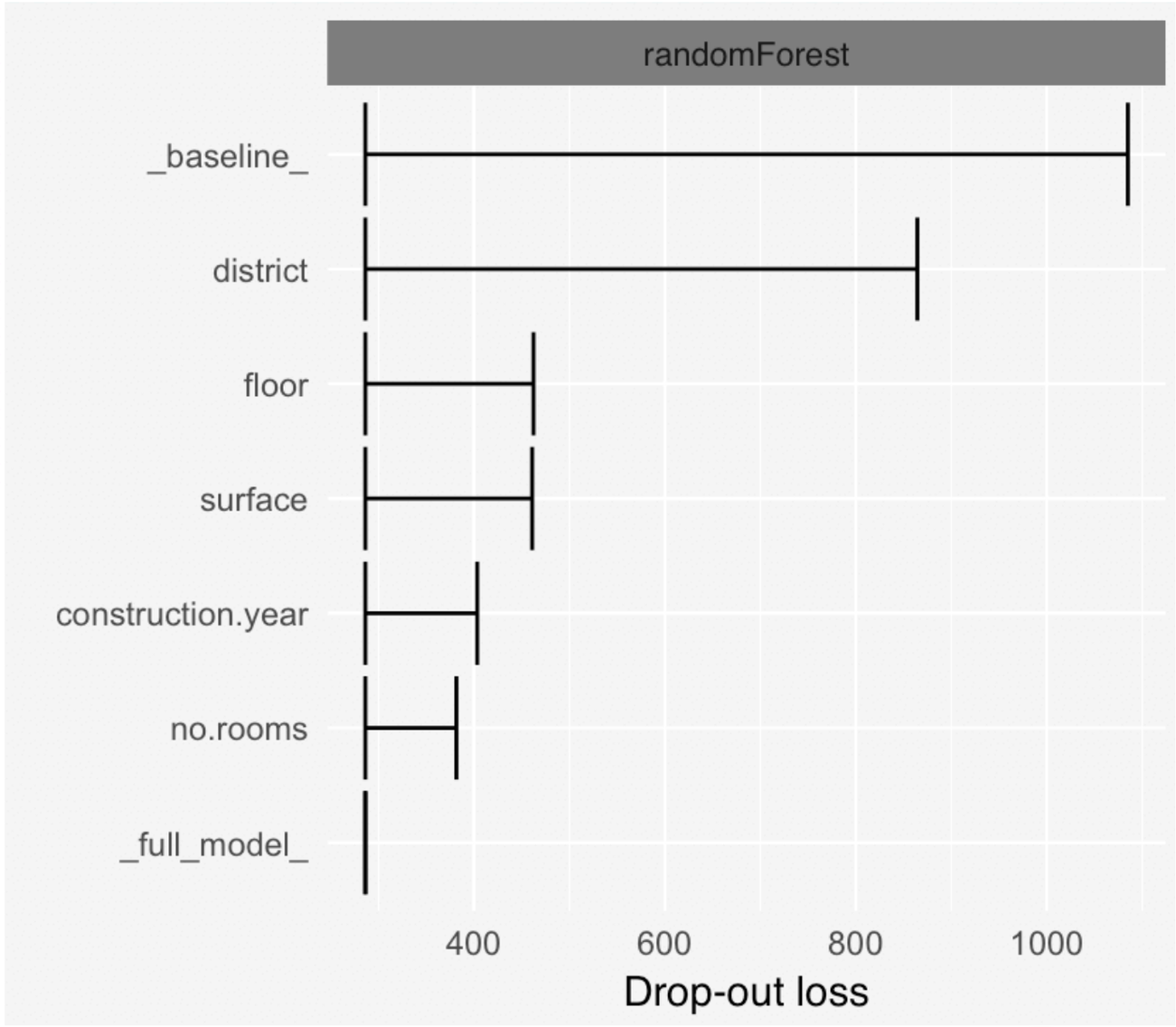
$$\begin{aligned}\hat{e}_{\text{switch}}(f) &:= \frac{1}{n(n-1)}\sum_{i=1}^{n}\sum_{j\neq i} h_f(\mathbf{Z}_{[i,\cdot]}, \mathbf{Z}_{[j,\cdot]}) \\ &= \frac{1}{n(n-1)}\sum_{i=1}^{n}\sum_{j\neq i} L\{f, (\mathbf{y}_{[j]}, \mathbf{X}_{1[i,\cdot]}, \mathbf{X}_{2[j,\cdot]})\}.\end{aligned}$$

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

Fisher, Aaron, Cynthia Rudin, and Francesca Dominici. 2018. "Model Class Reliance: Variable Importance Measures for Any Machine Learning Model Class, from the 'Rashomon' Perspective." Journal of Computational and Graphical Statistics http://arxiv.org/abs/1801.01489

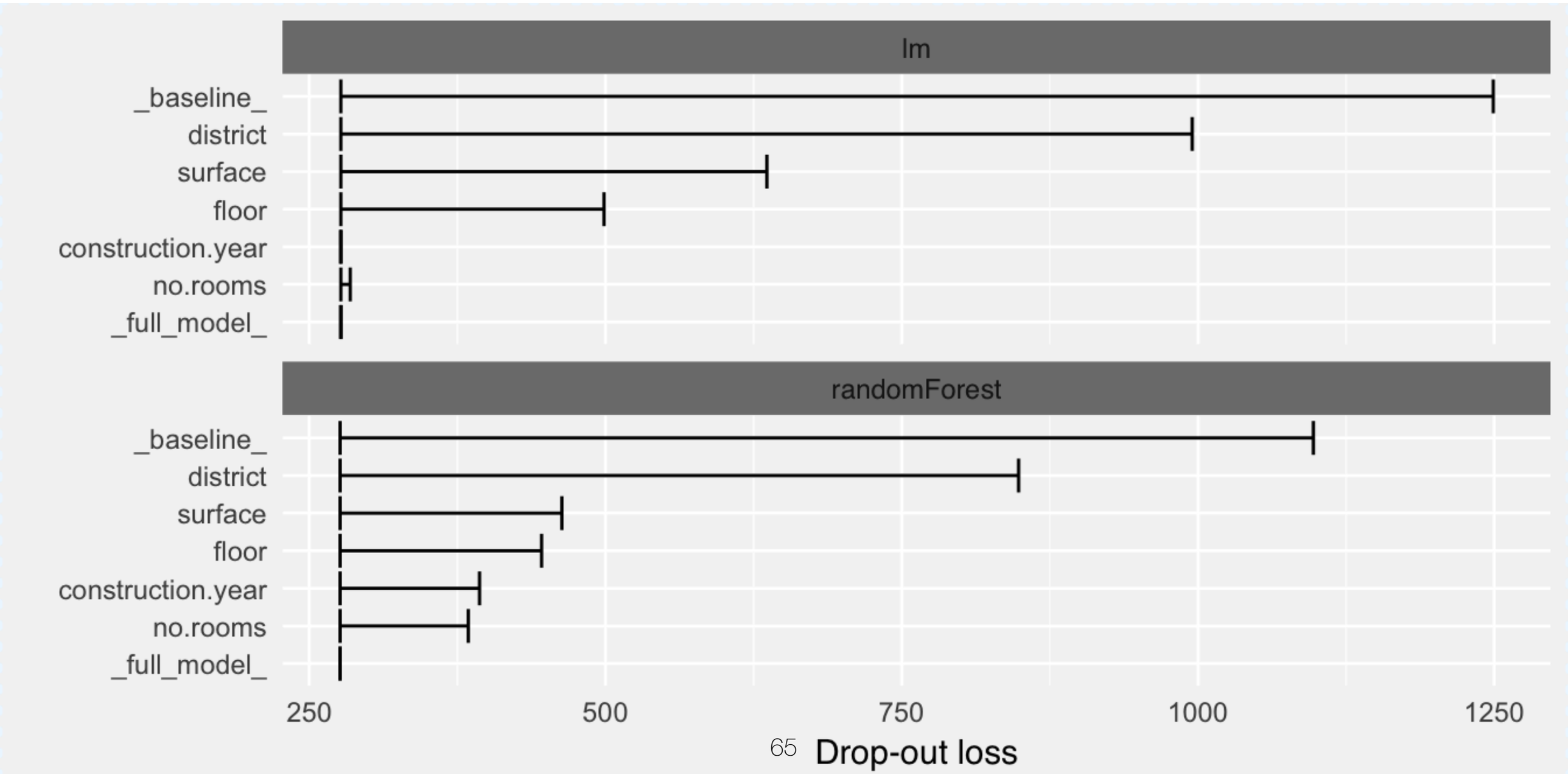
Which variables are important?

```
> 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!



Your turn! (approx 15 minutes)

1. Make sure that you still have wrappers for three models created in last exercises.
2. Prepare model performance explainers with the model_performance() function. Visualize them as boxplots and ecdf curves.
3. Perform simple audit for residuals in these three models. Use the auditor package. Are there any issues with these residuals?
4. Prepare variable importance explainers for all three models. Use the variable_importance() function. Are there variables that have different importance in different models? Which variables are the most important?

Coffee break

```
repeat ({
  slides()
  exercises()
  if (is.coffe()) break
})
```

Single prediction explainers

Why explain a single prediction?

- Someone asks about it
  - we are forced to explain it

- Someone wants to change the prediction
  - we want to recommend something

- Unusually high residual for an observation
  - we want to improve the model

- Very different predictions from two models
  - we want to compare models

Think about other reasons…

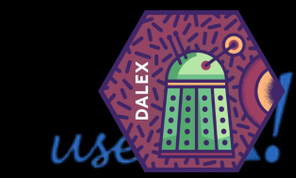
How to explain a single prediction?

- Explore local behaviour of a model
    - Ceteris Paribus Plots
    - local approximations with a white box model
        - LIME
        - live

- Explore local goodness-of-fit
    - Wangkardu Plots (work in progress)

- Explore local variable contributions
    - Break Down
    - Shapley values

Ceteris Paribus Plots

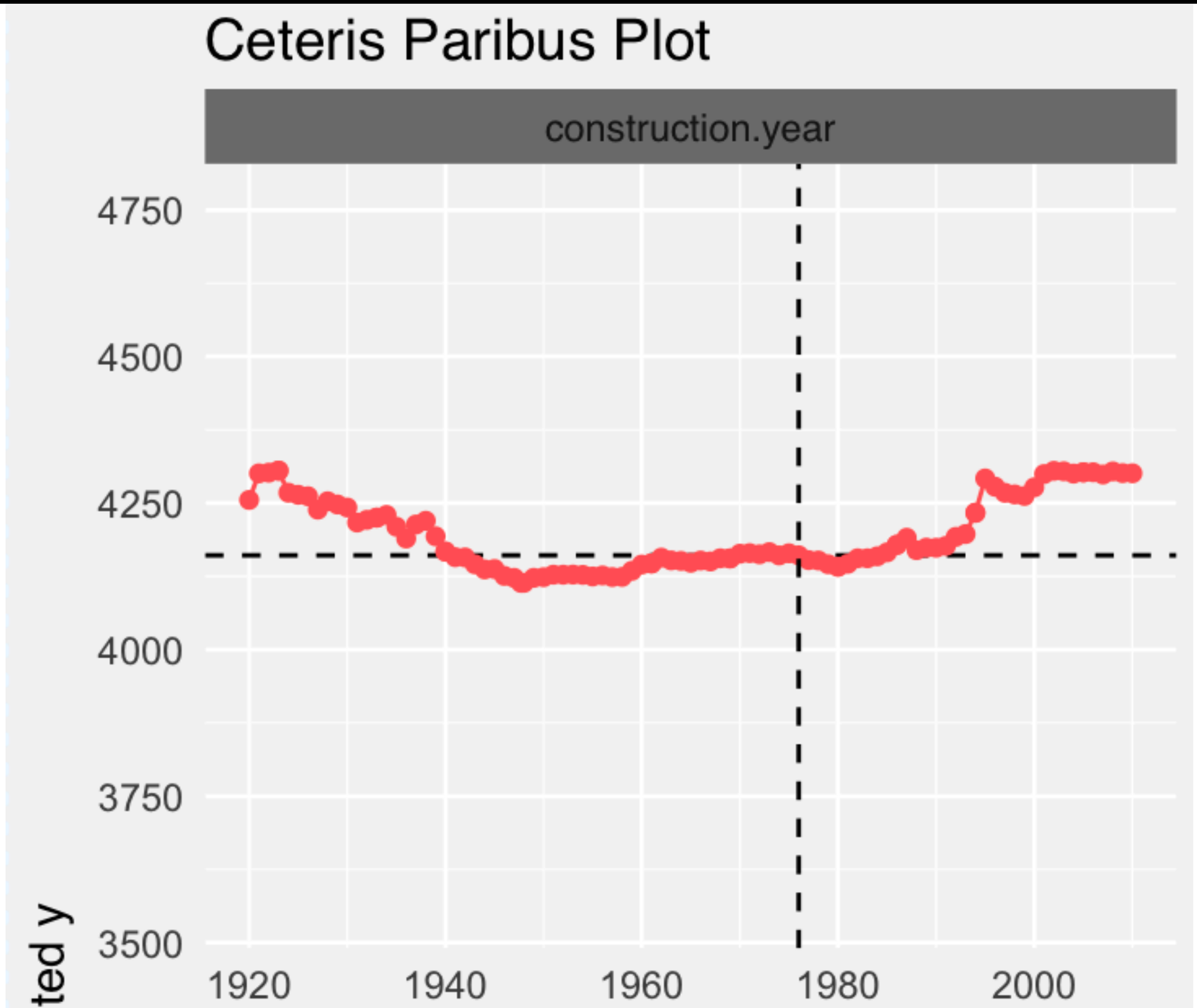


How to explain this?

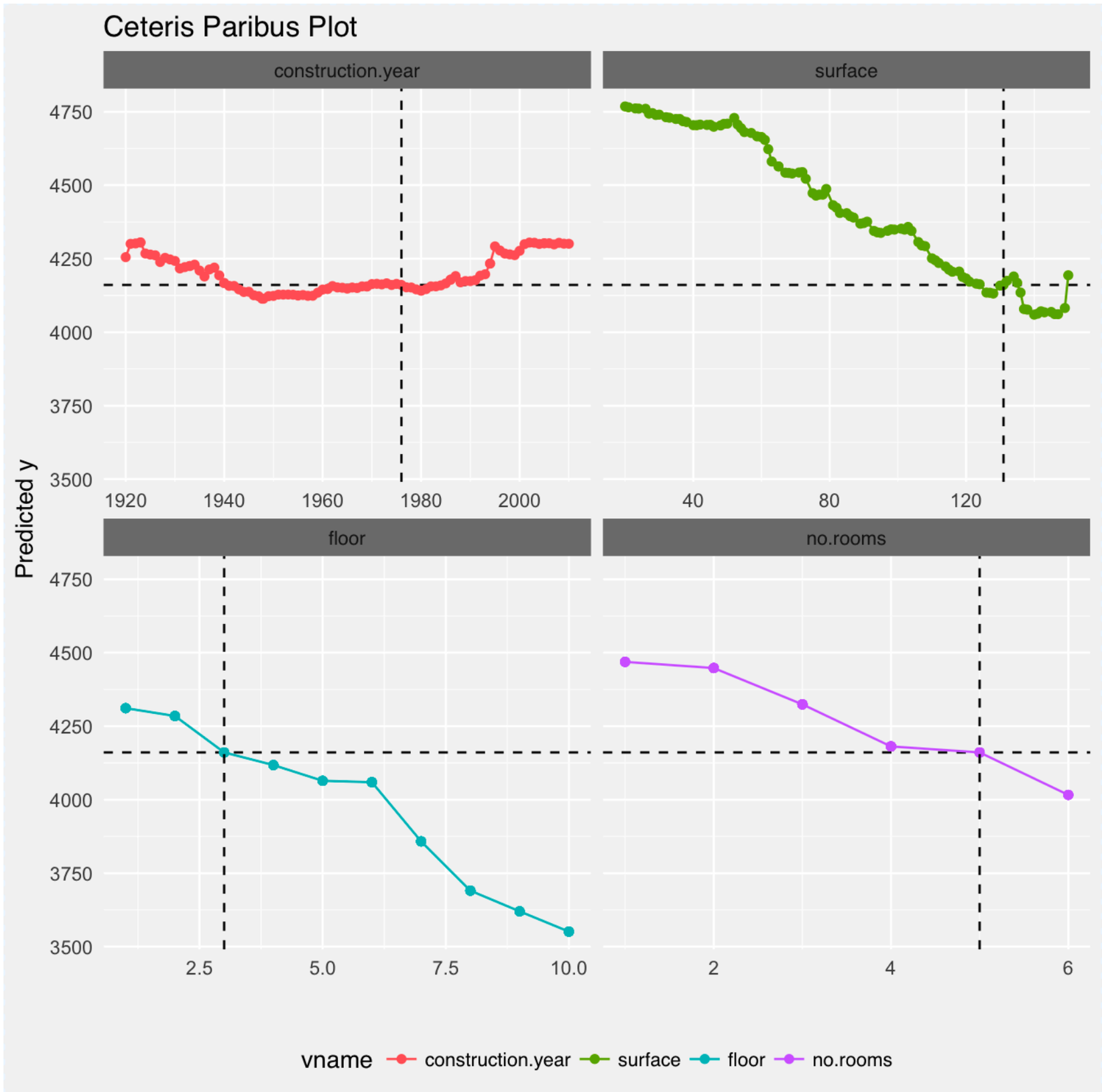
Case:
large (130 m2) flat, 5 rooms,
3 rd floor, built in 1978

Prediction from random forest:
4200 EUR/m2

Ceteris Paribus is a Latin phrase for "all else unchanged"

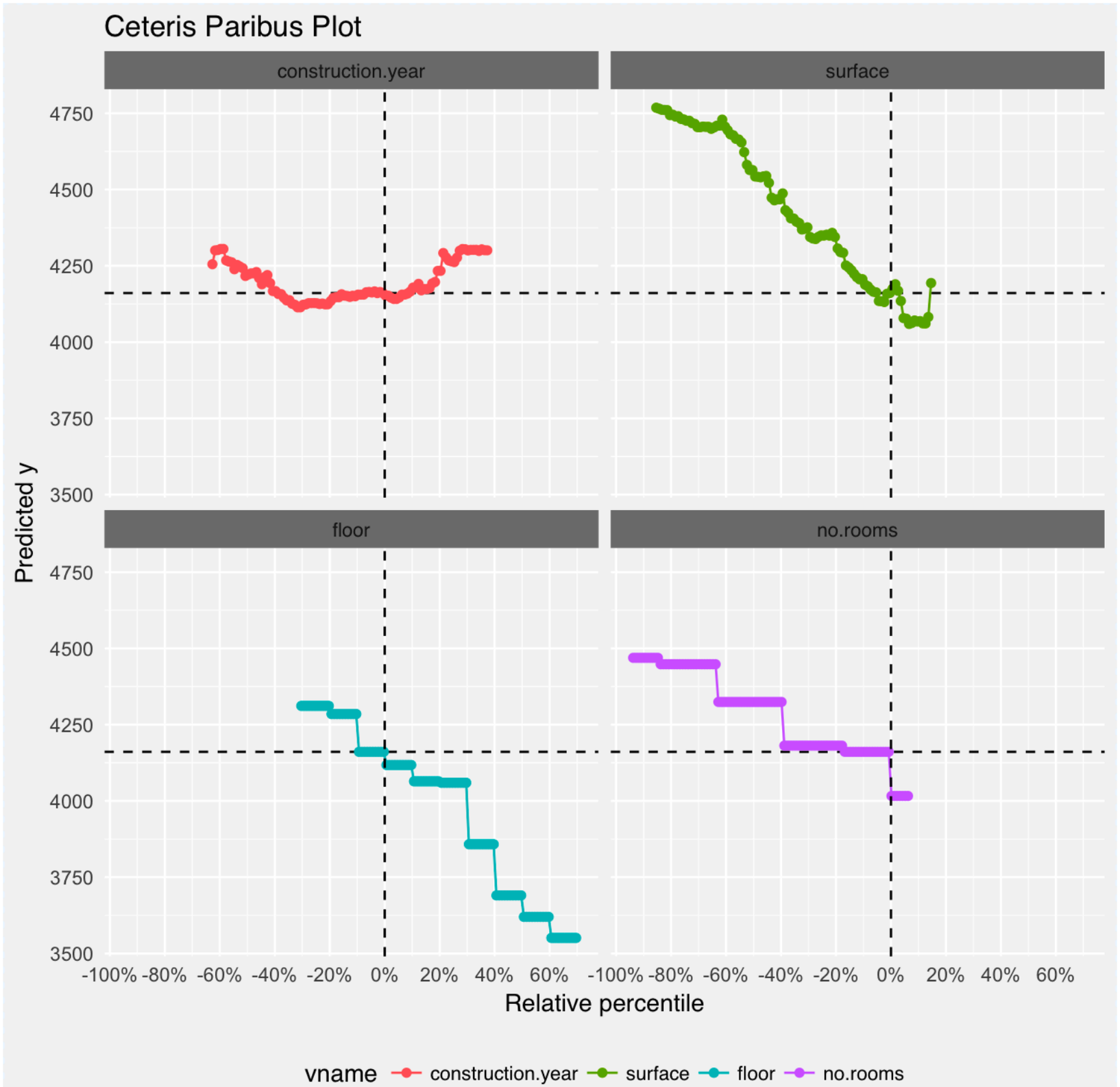


Use facets to show profiles for different variables

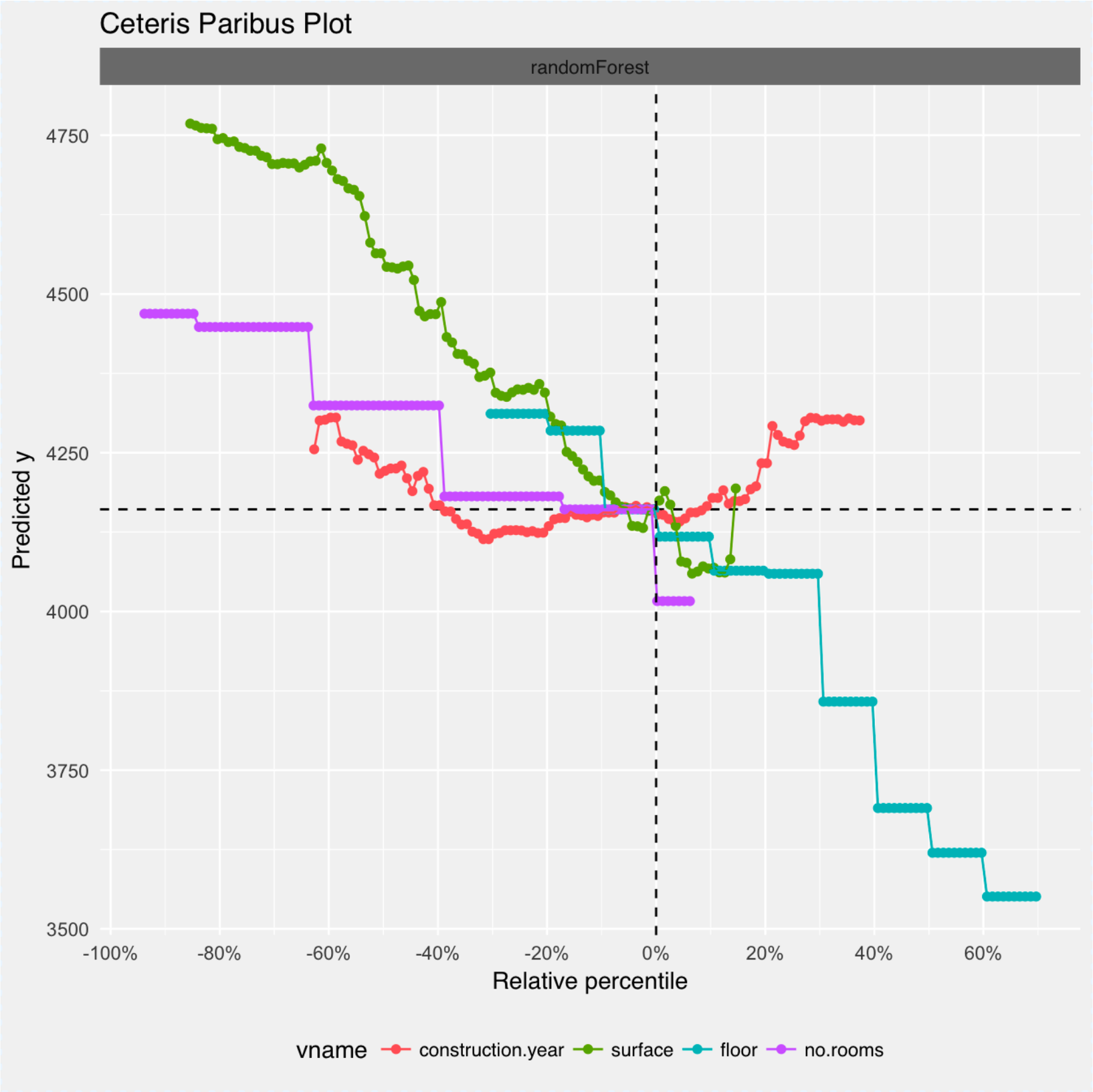


```r
# we need a DALEX object
explainer_rf <- explain(apartments_rf_model,
                        data = apartmentsTest[,2:6],
                        y    = apartmentsTest$m2.price)

# explanations for this data point
new_apartment <- apartmentsTest[1, ]

# as usual, create an explainer and plot it
library("ceterisParibus")
wi_rf <- ceteris_paribus(explainer_rf,
                         observation = new_apartment)

plot(wi_rf,
     split     = "variables",
     color     = "variables",
     quantiles = FALSE)
```

Plot relative quantiles on the OX axis



With relative quantiles you can fit a lot in a single graph



Compare few models in a single plot

] | https://github.com/MI2DataLab/live

live: Local Interpretable (Model-agnostic) Visual Explanations

Installation

Sparse explanations!

To get started, install stable CRAN version:

```
install.packages("live")
```

or the development version:

```
devtools::install_github("MI2DataLab/live")
```

See the latest changes.

Features coming up next:

- better support for comparing explanations for different models / different instances,
- improved Shiny application (see `live_shiny` function in development version).

If you have any bug reports, feature requests or ideas to improve the methodology, feel free to leave an issue.

Materials

Developer: Mateusz Staniak

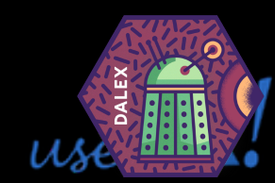
Find the paper about `live` and breakDown on arXiv.

Website: https://mi2datalab.github.io/live/

Conference talk on `live` : https://github.com/mstaniak/Berlin_2017

Wangkardu Plots

Local goodness-of-fit Plots

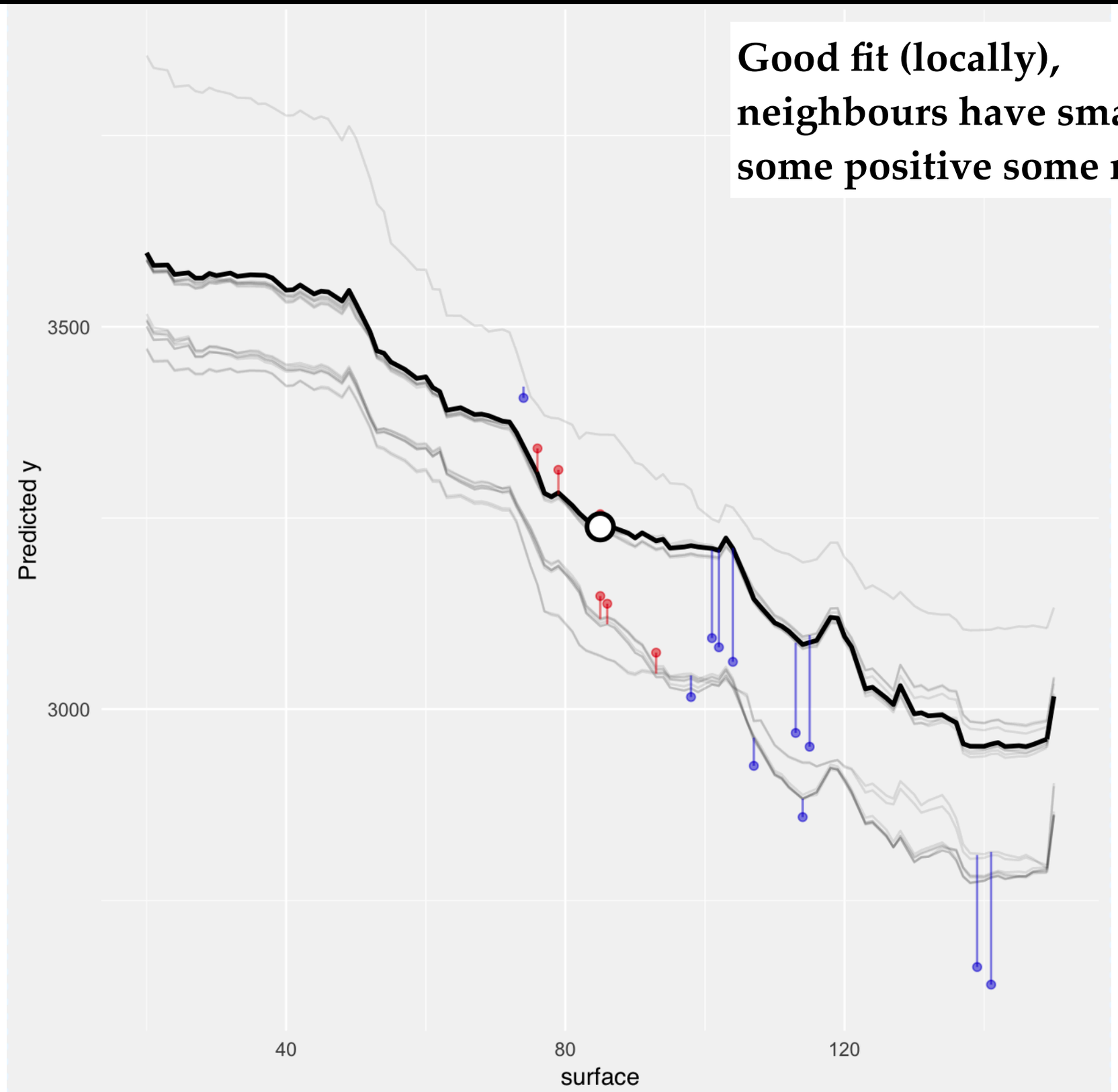


Is it a good prediction?

Case:
large (130 m2) flat, 5 rooms,
3 rd floor, built in 1978

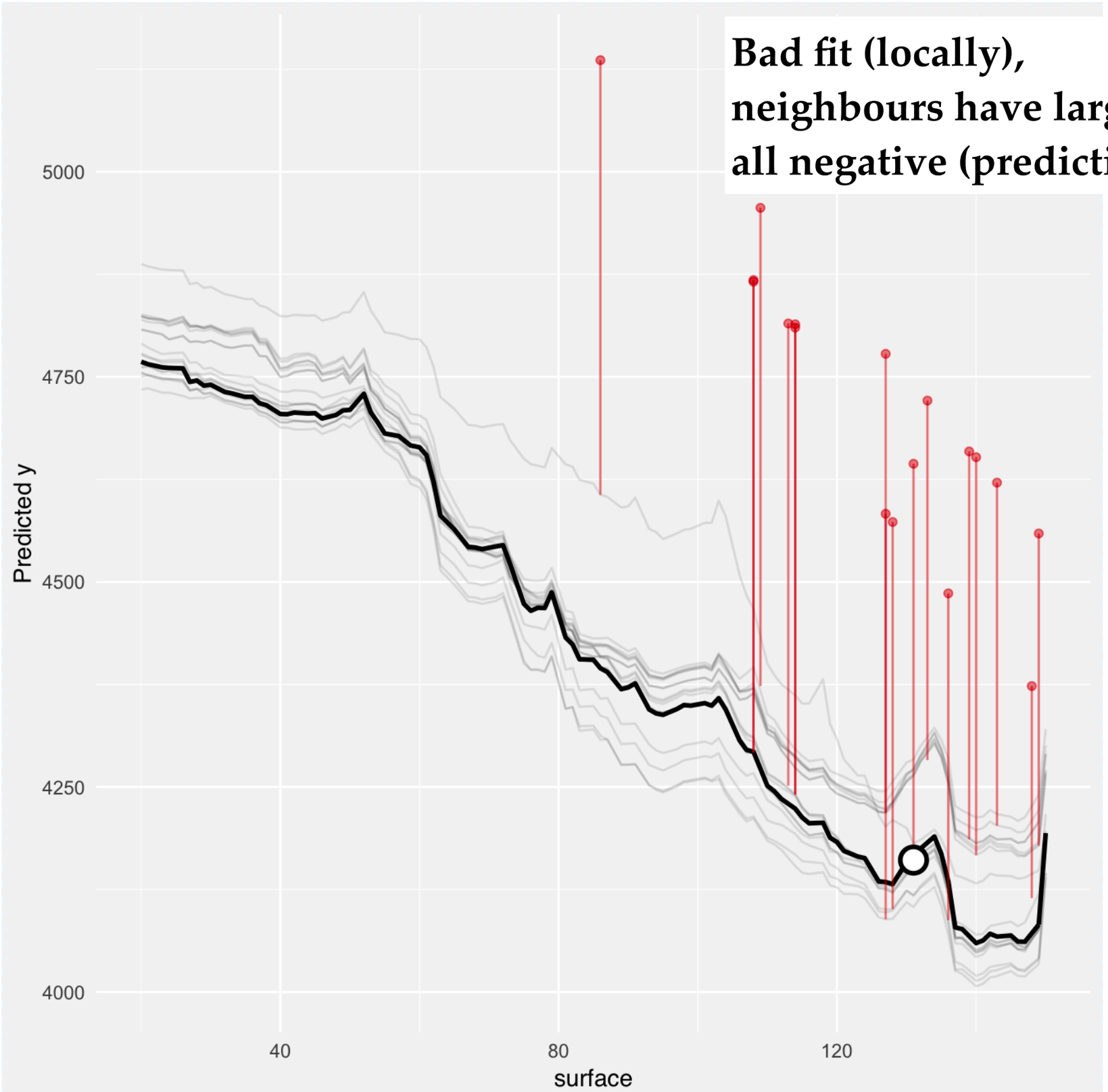
Prediction from random forest:
4200 EUR/m2

How large are residuals around?



Good fit (locally),
neighbours have small residuals
some positive some negative

How large are residuals around?



Bad fit (locally),
neighbours have large residuals
all negative (predictions below truth)

```r
# we need a DALEX object
explainer_rf <- explain(apartments_rf_model,
                        data = apartmentsTest[,2:6],
                        y    = apartmentsTest$m2.price)

# explanations for this data point
new_apartment <- apartmentsTest[1, ]
```

```r
# as usual, create an explainer and plot it
library("ceterisParibus")
cr_rf <- local_fit(explainer_rf,
                   observation   = new_apartment,
                   select_points = 0.002)
plot(cr_rf)
plot(cr_rf,        palette       = "wangkardu")
```

Side story

Title

Wangkardu

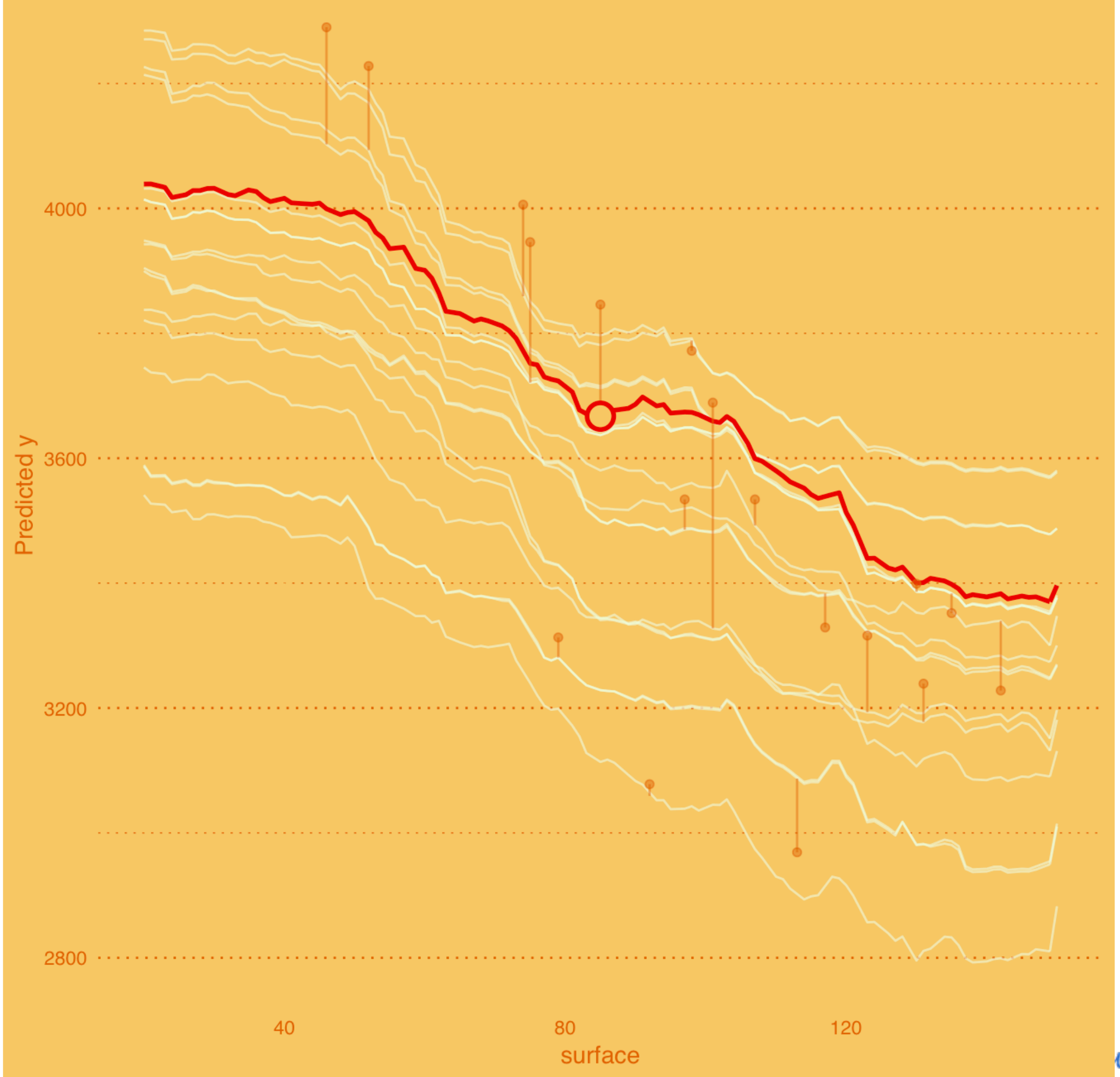
2001

Artist

Helicopter Tjungurrayi

Australia
01 Jul 1947 -

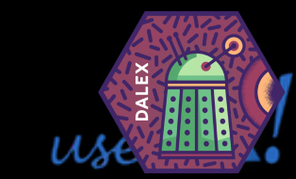
Language group
Kukatja, Western Desert region


4000
3600
3200
2800
Predicted y
40
80
120
surface

Your turn! (approx 15 minutes)

1. Make sure that you still have wrappers for three models created in last exercises.
2. Select a single observation (e.g. apartmentsTest[1,]). Prepare Ceteris Paribus Plots for this observation for the Random Forest model. Which single variable should be changes to increase the prediction the most?
3. Compare Ceteris Paribus Plots for all three models in a single chart.
4. Repeat steps 2 and 3 for observation apartmentsTest[10,].
5. Prepare Wnagkardu explainers for observation apartmentsTest[1,]. Which models are well fitted for this observation and which are not?
6. Repeat step 5 for observation apartmentsTest[10,].

Break Down Plots

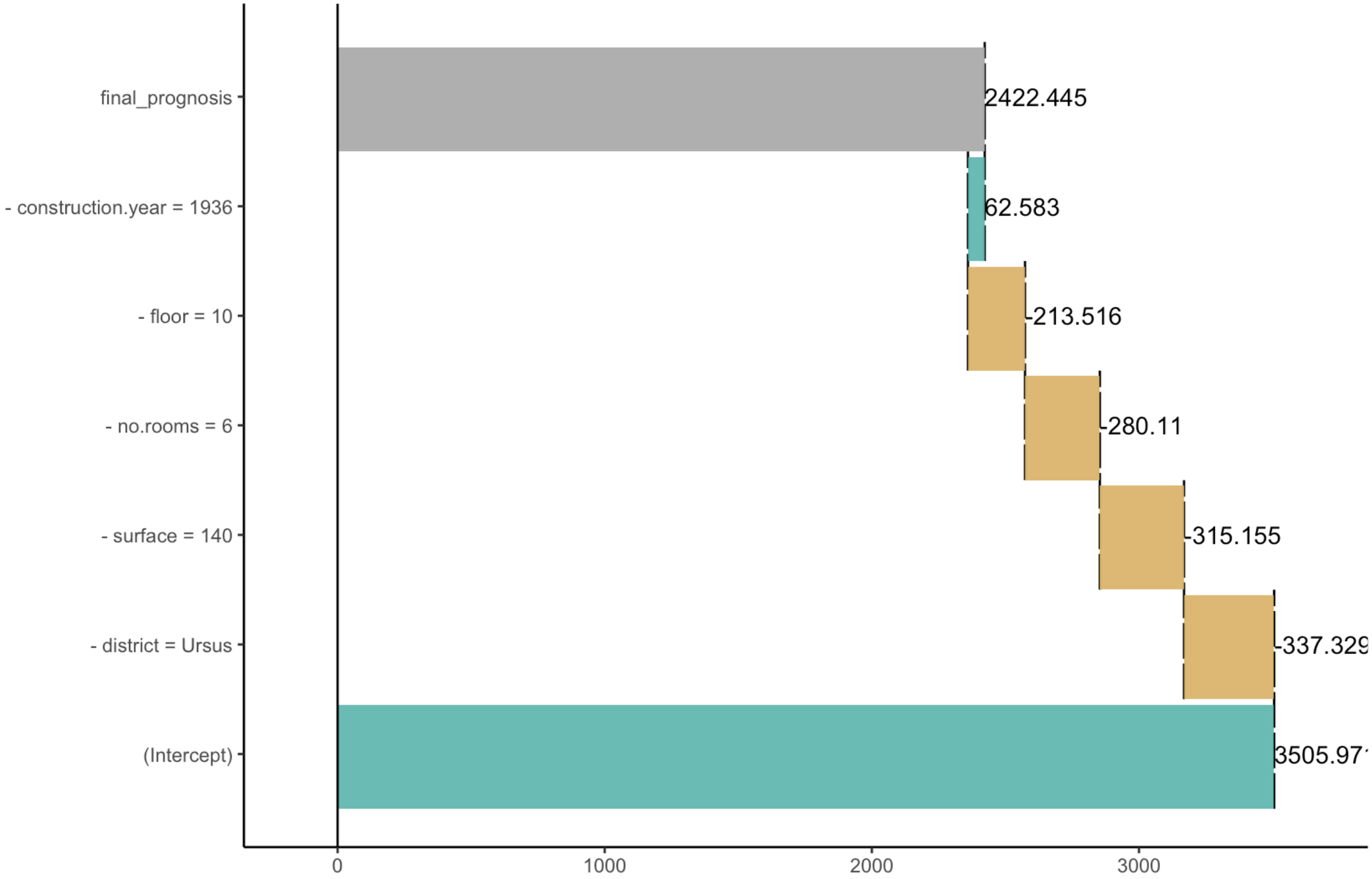


How variables affect this prediction?

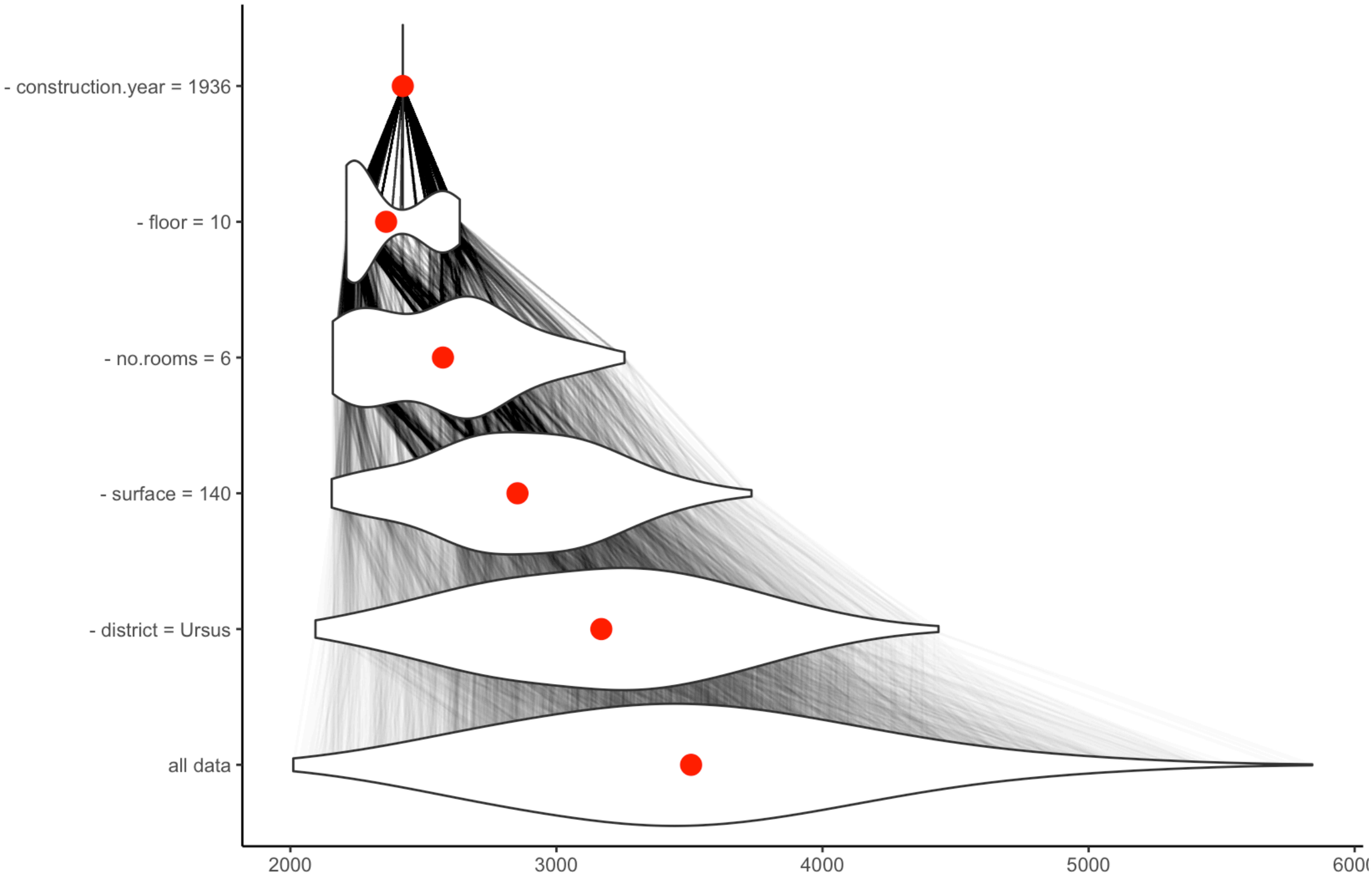
Case:
large (140 m2) flat, 6 rooms,
10 th floor, built in 1936

Prediction from random forest:
3505 EUR/m2

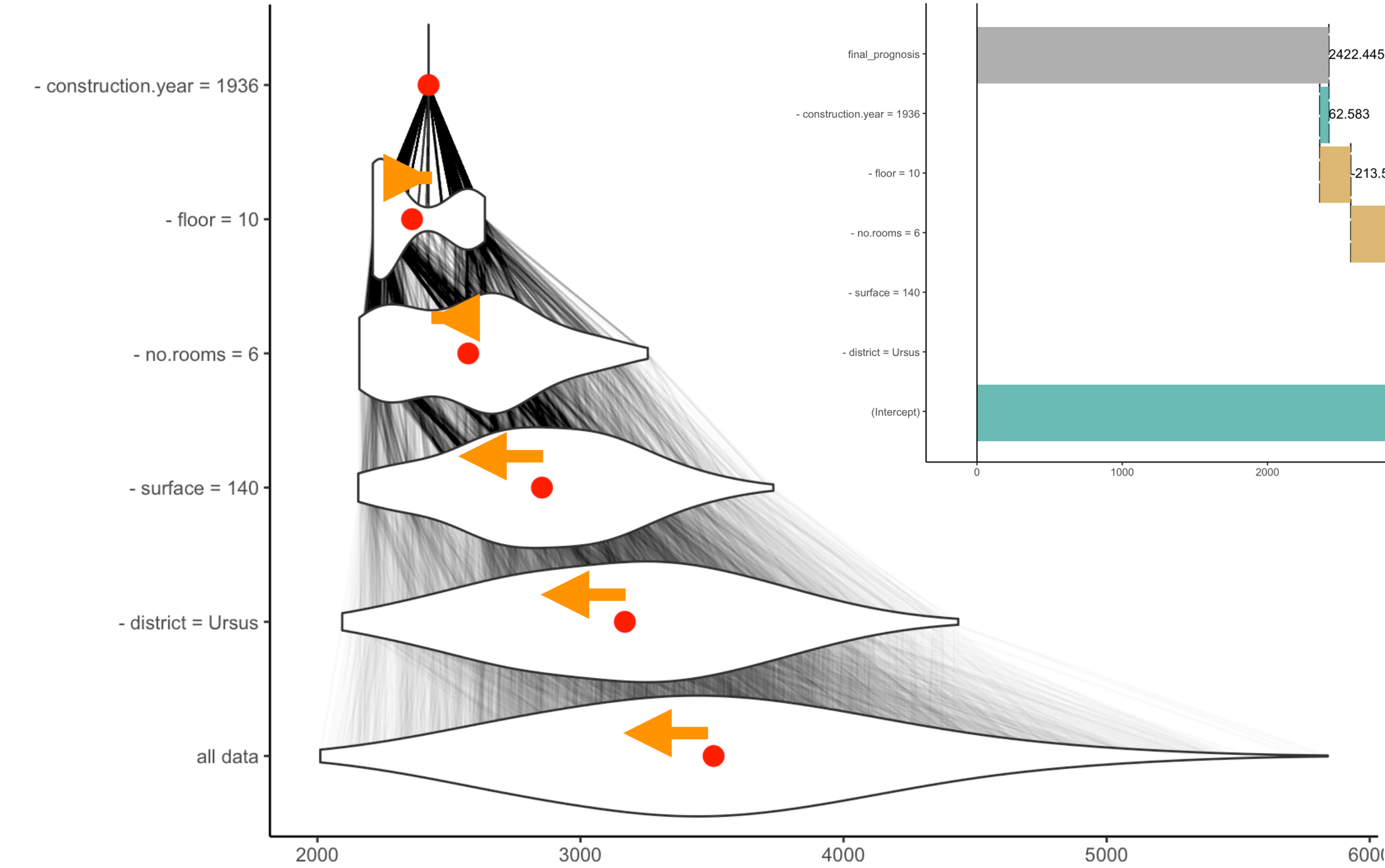
How variables affect this prediction?



How variables affect this prediction?



How variables affect this prediction?



In details

Algorithm 2 Model agnostic break down of model predictions. The *step-down* approach.

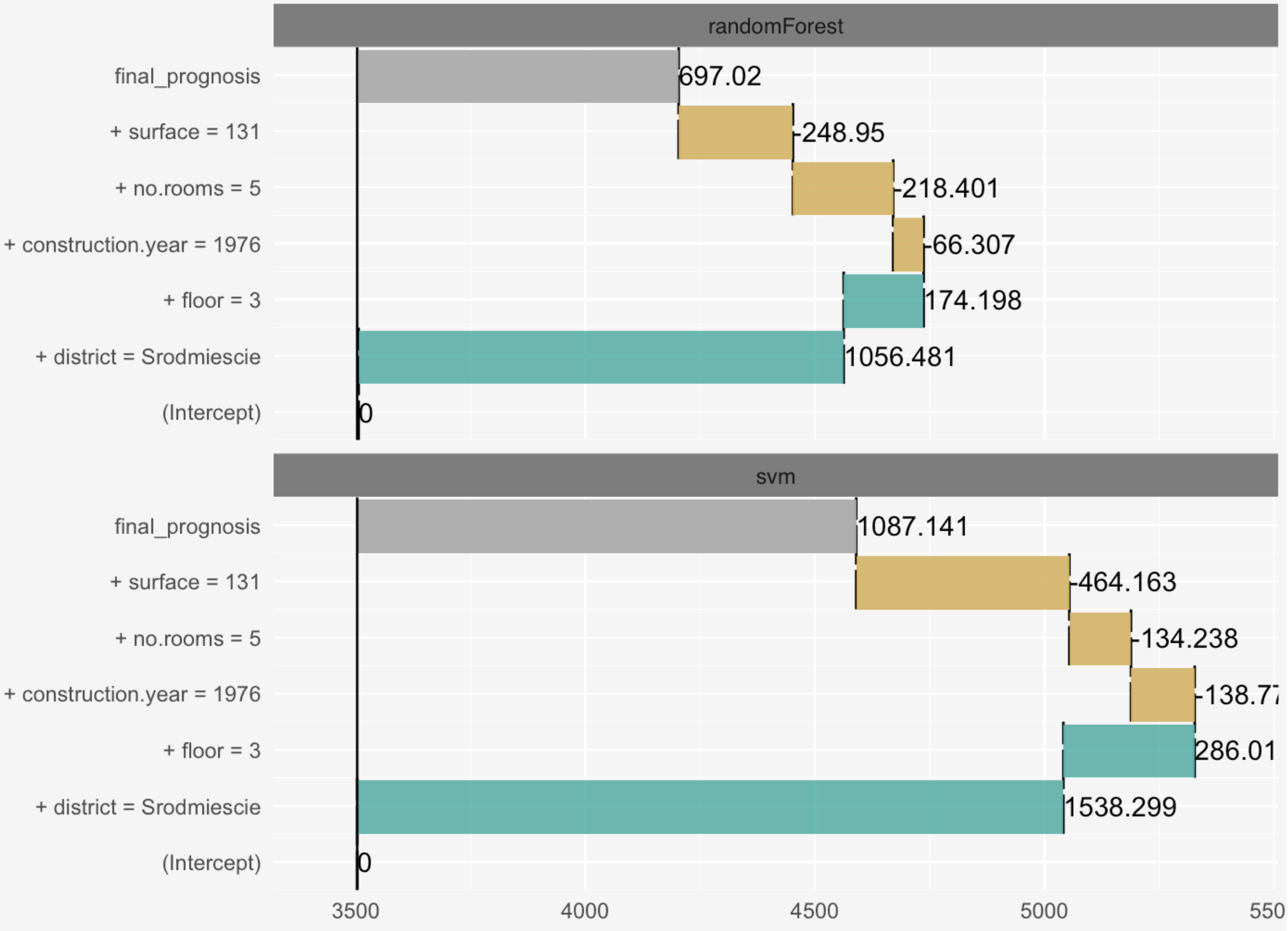
$p \leftarrow$ number of variables
$IndSet \leftarrow \{1, \ldots, p\}$ set of indexes of all variables
for i in $\{1, \ldots, p\}$ **do**
    Find new variable that can be relaxed with small loss in relaxed distance to $f(x^{new})$
    for j in $IndSet$ **do**
        Calculate relaxed distance with j removed
        $dist(j) \leftarrow d(x^{new}, IndSet \setminus \{j\})$
    end for
    Find and remove j that minimizes loss
    $j_{min} \leftarrow \arg\min_j dist(j)$
    $Contribution^{IndSet}(i) \leftarrow f^{IndSet}(x^{new}) - f^{IndSet \setminus \{j_{min}\}}(x^{new})$
    $Variables(i) \leftarrow j_{min}$
    $IndSet \leftarrow IndSet \setminus \{j_{min}\}$
end for

Explanations of model predictions with live and breakDown packages
Mateusz Staniak, Przemyslaw Biecek
https://arxiv.org/pdf/1804.01955.pdf

```r
# for Random Forest Model
br_rf <- prediction_breakdown(explainer_rf,
                              observation = new_apartment)
plot(br_rf)
```

```r
# + SVM models
br_svm <- prediction_breakdown(explainer_svm,
                               observation = new_apartment)
plot(br_rf, br_svm)
```

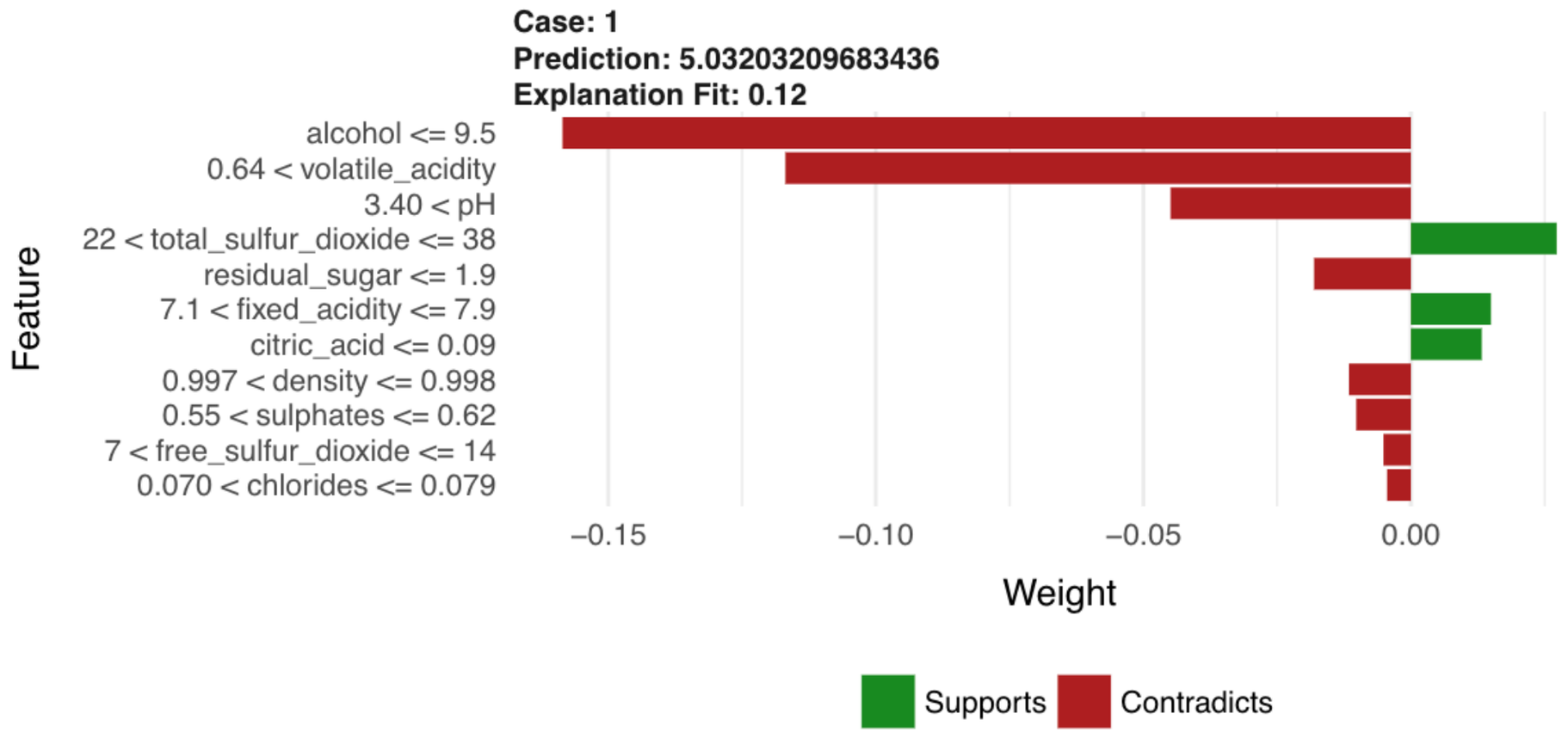
Compare models



LIME / live

VS

Break Down



(a) Husky classified as wolf

(b) Explanation

Other tools

modelDown: pkgdown for models

https://github.com/MI2DataLab/modelDown

modelDown

build passing

`modelDown` generates a website with HTML summaries for predictive models. Is uses DALEX explainers to compute and plot summaries of how given models behave. We can see how exactly scores for predictions were calculated (Prediction BreakDown), how much each variable contributes to predictions (Variable Response), which variables are the most important for a given model (Variable Importance) and how well out models behave (Model Performance).

`pkgdown` documentation: https://mi2datalab.github.io/modelDown/

An example website for regression models: https://mi2datalab.github.io/modelDown_example/

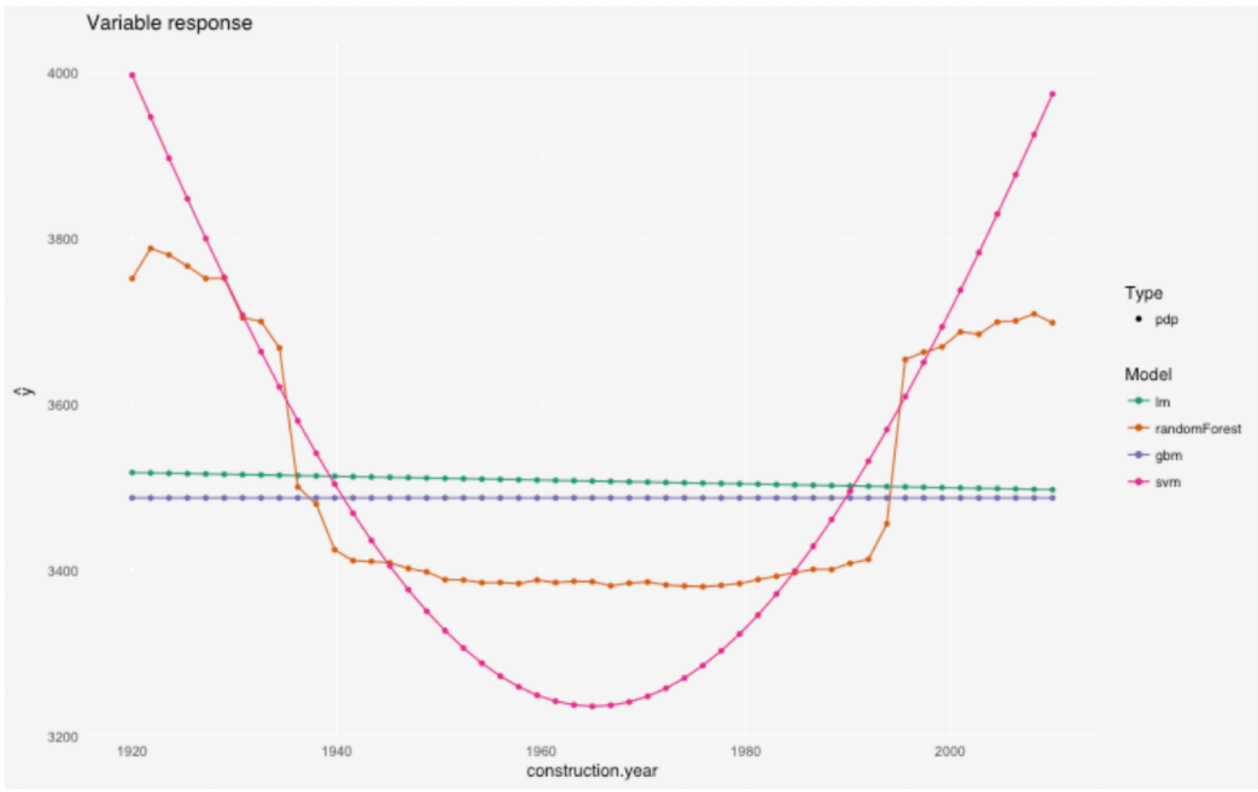
Getting started

Do you want to start right now ? Check out our getting started guide.

modelDown | Model Performance | Variable Importance | Variable Response | Prediction BreakDown

- construction.year
- district
- floor
- no.rooms
- surface

construction.year



https://mi2datalab.github.io/modelDown_example/

Interesting software (not mentioned above)

- LME: Local Interpretable Model-Agnostic Explanations (2016)

  https://github.com/thomasp85/lime - R

  https://github.com/marcotcr/lime - python

- SHAP (SHapley Additive exPlanations) (2017)

  https://github.com/slundberg/shap - python

- condvis: Conditional Visualization for Statistical Models (2017)

  https://github.com/markajoc/condvis/ - R

- H2O Driverless AI (2017)

  https://www.h2o.ai/driverless-ai/ - H2O

- ICEbox: Individual Conditional Expectation (2017)

  https://github.com/kapelner/ICEbox - R

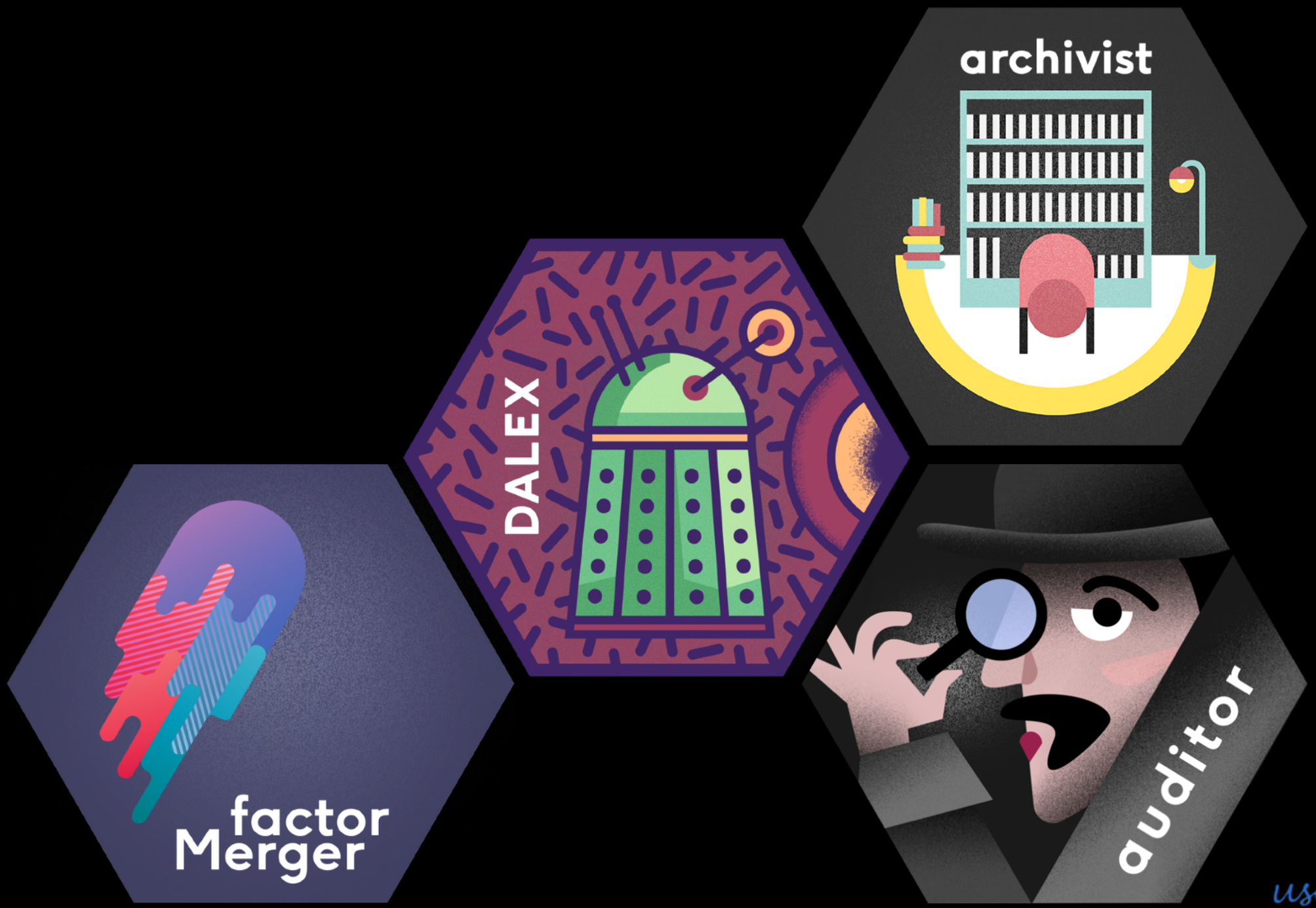
- iml: interpretable machine learning (2018)

  https://github.com/christophM/iml - R

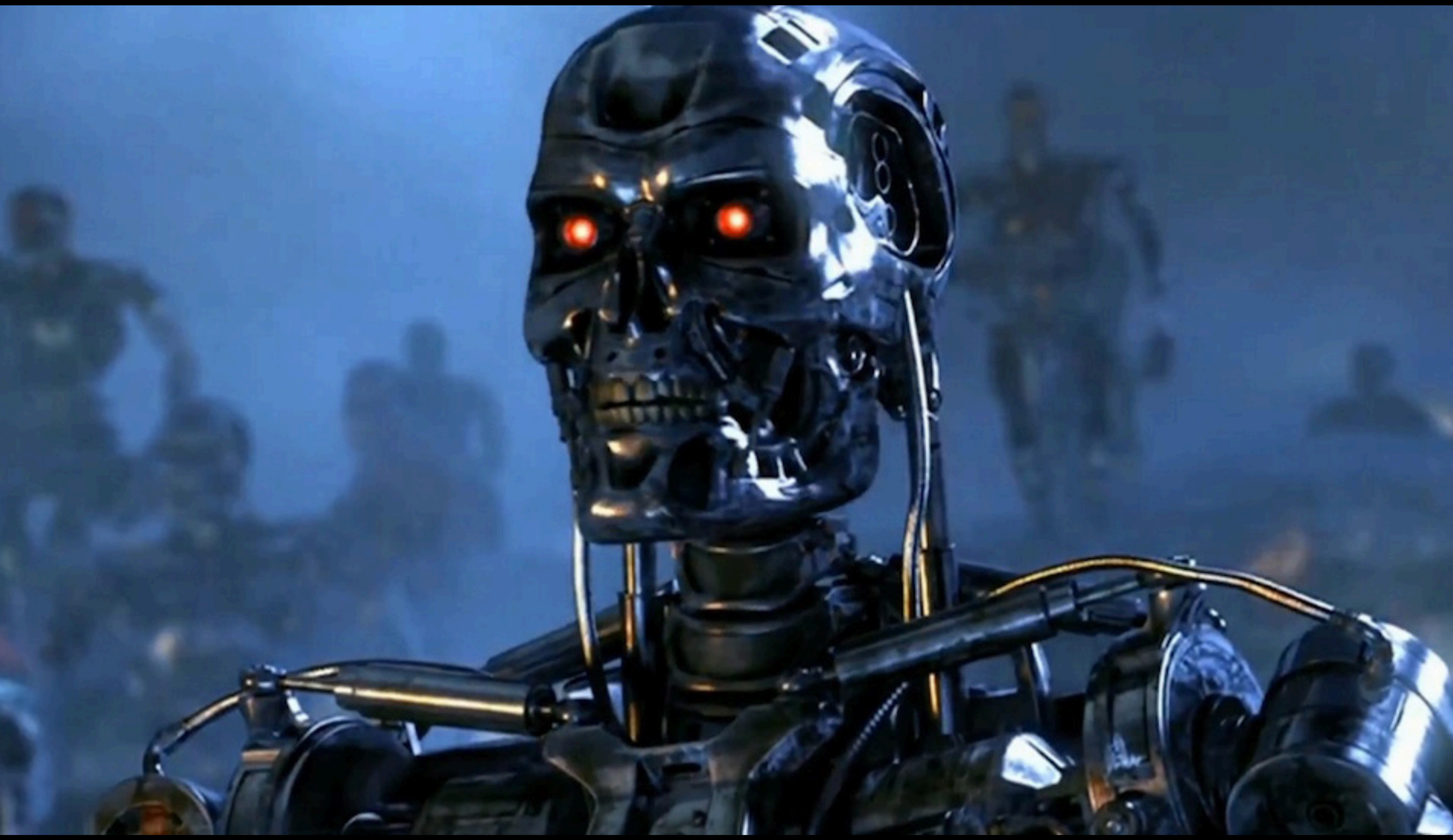
Your turn! (approx 15 minutes)

1. Make sure that you still have wrappers for three models created in last exercises.
2. Select a single observation (e.g. apartmentsTest[1,]). Prepare Break Down Plots for this observation.
3. Compare Break Down Plots for all three models. How they are different?
4. Use the modelDown package for all three models.
5. Try the live package for this observation.
6. Try the lime package for this observation.

DALEX as
Model - Human interface



Machine Learning Models will replace humans



Machine Learning Models will empower humans

Find more at https://github.com/pbiecek/DALEX

] | https://github.com/pbiecek/DALEX

DALEX

CRAN 0.2.3 | downloads 1841/month | downloads 4050 | build passing | coverage 92%

DALEX: Descriptive mAchine Learning EXplanations

Machine Learning models are widely used and have various applications in classification or regression tasks. Due to increasing computational power, availability of new data sources and new methods, ML models are more and more complex. Models created with techniques like boosting, bagging of neural networks are true black boxes. It is hard to trace the link between input variables and model outcomes. They are use because of high performance, but lack of interpretability is one of their weakest sides.

In many applications we need to know, understand or prove how input variables are used in the model and what impact do they have on final model prediction. `DALEX` is a set of tools that help to understand how complex models are working.

Find more about DALEX in this Gentle introduction to DALEX with examples.

DALEX Stories

- An interactive notebook with examples: launch binder

How to use DALEX

- How to use DALEX with caret
- How to use DALEX with mlr
- How to use DALEX with H2O
- How to use DALEX with xgboost package
- How to use DALEX for teaching. Part 1
- How to use DALEX for teaching. Part 2
- breakDown vs lime vs shapleyR