eRum exercises

Mateusz Staniak

Exercises

Exercise 1

Run the following code to fit random forest, linear regression and SVM to the housing prices data.

```
library(tidyverse)
library(live)
library(DALEX)
library(randomForest)
library(e1071)
library(auditor)
load("./rda_files/houses.rda")

set.seed(33)
house_rf <- randomForest(sqm_price ~., data = houses)
house_svm <- svm(sqm_price ~., data = houses)
house_lm <- lm(sqm_price ~., data = houses)
```

Create DALEX explainer object for each of the models. Create and compare boxplots of residuals for all the models (`model_performance`).

- Which model is the best?
- Are there any outlying predictions?
- Find the observation with the largest absolute value of residual among houses cheaper than 7000 PLN.

TIP: object returned by `model_performance` function is a data frame with colnames *predicted*, *observed*, *diff* and *label*.

Exercise 2

Create single prediction explainers for the instance chosen in **Exercise 1**. Create Break Down plots for each of the observations. What are the keys factors that drive the prediction? Are they the same for every model?

Exercise 3

Run the following code to train model random forest model using `mlr` interface (this is necessary for `shapleyR` package).

```
library(mlr)
load("./rda_files/houses.rda")
n_obs <- 1189

house_task <- makeRegrTask(data = houses, target = "sqm_price")
house_rf_mlr <- train("regr.randomForest", house_task)
```

Use shapleyR package to calculate Shapley values for prediction chosen in **Exercise 1** (its index is in `n_obs` object). Are the results consistent with Break Down results from **Exercise 2**? Draw a plot of Shapley values.

TIP: remember to set class of the object returned by `shapley` function to `shapley.singleValue` before using `plot`.

Bonus 1:

Draw plots of fitted vs observed values for each of these models. Can you spot any problems with the predictions? Are the prices usually underestimated of overestimated?

Bonus 2:

Create variable importance explainer. Compare global variable importance to scores obtained in **Exercise 2** and **Exercise 3**.

Exercise 4

```
n_obs <- 1189
```

Simulate new data around the observation from **Exercise 1** (its index is in the `n_obs` object.) and the add random forest predictions. Then fit a linear model locally.

TIP: remember to load `mlr` package. TIP2: don't use too small `size` for the simulated dataset. I recommend at least 1000.

Exercise 5

Visualize approximation created in **Exercise 4**. Use `plot_explanation2` function to create a forest plot of the linear model and then the Break Down plot.

Exercise 6

Use `lime` package to approximate random forest model around prediction chosen in **Exercise 1** (its index is in `n_obs` object). Follow the `lime` work flow: 1. Create an explainer. 2. Approximate the model around the explained instane. 3. Use `plot_features` function to see, how features influence this prediction.

TIP: use `house_rf_mlr` object from **Exercise 5**, because `lime` works well with `mlr` objects.

Bonus 3

Use `add_predictions` function to add SVM and LM predictions to the simulated dataset. Compare plots for all three models.

Bonus 4

Run the following code to see largest residuals for *Psie Pole* district.

```
library(tidyverse)
houses %>%
  mutate(id = 1:n()) %>%
  mutate(rf_pred = predict(house_rf)) %>%
  mutate(abs_res = abs(sqm_price - rf_pred)) %>%
```

```r
  arrange(desc(abs_res)) %>%
  filter(sqm_price < 7000,
         district == "Psie Pole") %>%
  head(5)
```

```
## # A tibble: 5 x 10
##   rooms_num  area sqm_price  year floor max_floor district     id rf_pred
##       <int> <dbl>     <int> <int> <int>     <int> <fct>     <int>   <dbl>
## 1         2  46.0      2174  2005     2         5 Psie Pole  5830   6541.
## 2         2  46.0      1957  2001     4         5 Psie Pole   942   5829.
## 3         3  65.0      1877  2001     8        10 Psie Pole  4303   5663.
## 4         3  75.0      4267  2013     0         0 Psie Pole  4308   6388.
## 5         4  88.4      3394  2004     4         5 Psie Pole  3489   5495.
## # ... with 1 more variable: abs_res <dbl>
```

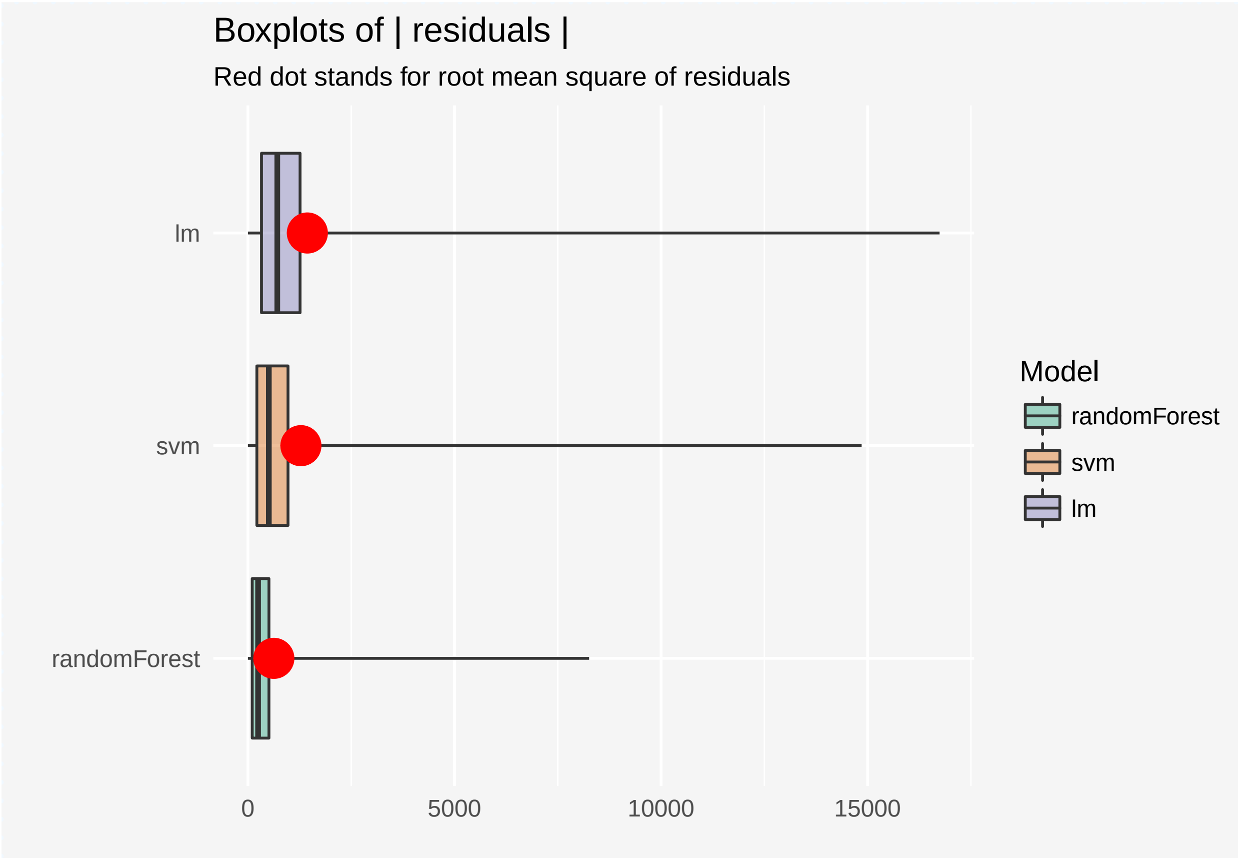
```r
n_obs2 <- 5830
```

Using `live` package, fit a linear model around the top observation. Compare waterfall plots for this prediction and the prediction from **Exercise 5**. How are they different?

Solutions

Exercise 1

```r
rf_expl <- DALEX::explain(house_rf, data = houses, y = houses$sqm_price)
svm_expl <- DALEX::explain(house_svm, data = houses, y = houses$sqm_price)
lm_expl <- DALEX::explain(house_lm, data = houses, y = houses$sqm_price)

rf_perf <- model_performance(rf_expl)
svm_perf <- model_performance(svm_expl)
lm_perf <- model_performance(lm_expl)

plot(rf_perf,
     svm_perf,
     lm_perf,
     geom = "boxplot")
```




```r
rf_perf %>%
  mutate(id = 1:n()) %>%
  arrange(desc(abs(diff))) %>%
  filter(observed < 7000) %>%
  head(5)
```

```
##   predicted observed     diff        label   id
## 1  4338.774     1585 2753.774 randomForest 1189
## 2  4842.923     2174 2668.923 randomForest 5830
## 3  7399.405     4750 2649.405 randomForest 4824
## 4  6326.248     3742 2584.248 randomForest 4419
## 5  8345.797     6000 2345.797 randomForest 4005
```

```r
svm_perf %>%
  mutate(id = 1:n()) %>%
  arrange(desc(abs(diff))) %>%
  filter(observed < 7000) %>%
  head(5)
```

```
##   predicted observed     diff label   id
## 1  5771.421     1585 4186.421   svm 1189
## 2  7983.699     4063 3920.699   svm  356
## 3  5722.197     1957 3765.197   svm  942
## 4  5907.319     2174 3733.319   svm 5830
## 5  5544.827     1877 3667.827   svm 4303
```

```r
lm_perf %>%
  mutate(id = 1:n()) %>%
```

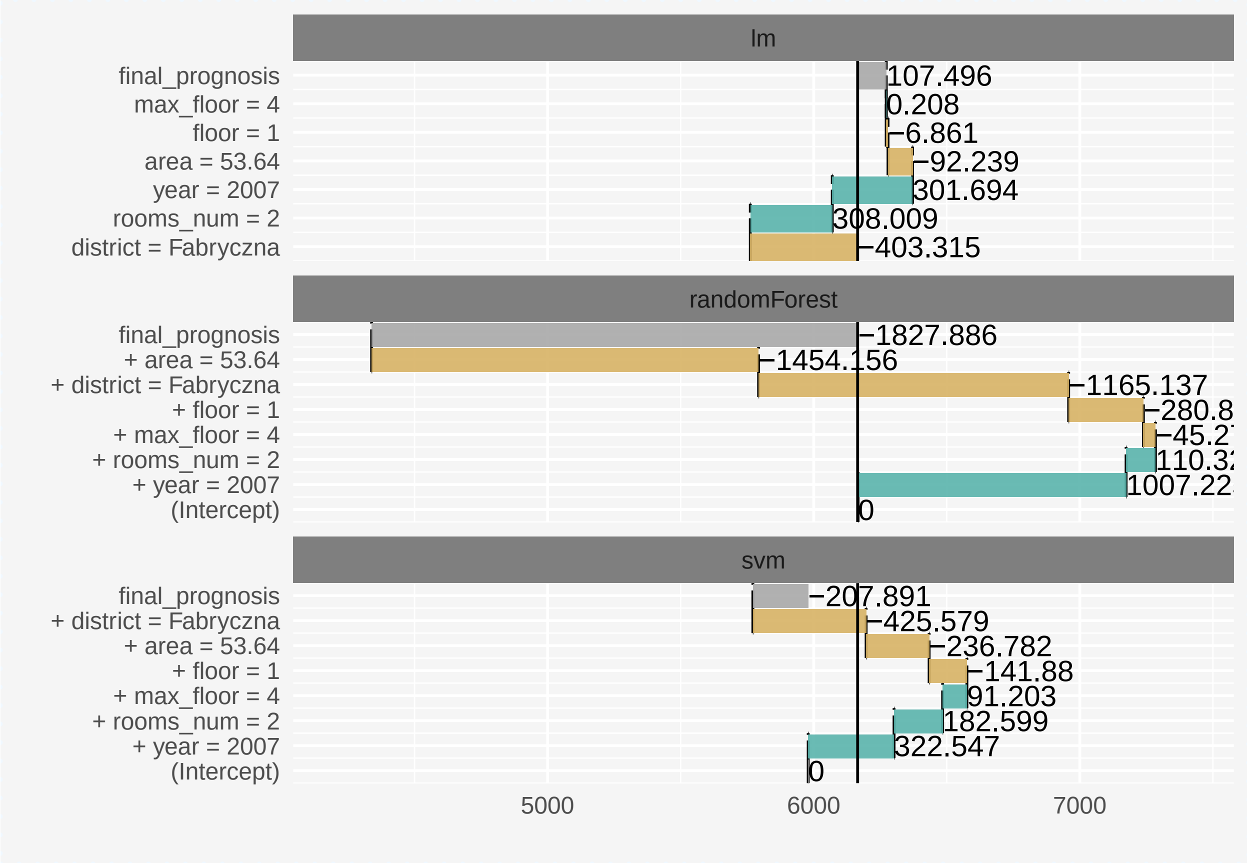
```r
  arrange(desc(abs(diff))) %>%
  filter(observed < 7000) %>%
  head(5)
```

```
##   predicted observed     diff label   id
## 1  6272.608     1585 4687.608    lm 1189
## 2  8325.985     3702 4623.985    lm 1208
## 3  8250.050     4267 3983.050    lm 5726
## 4  6466.103     2638 3828.103    lm 5751
## 5  5744.796     1957 3787.796    lm  942
```

```r
n_obs <- which(houses$sqm_price == 1585)
```

Exercise 2

```r
rf_expl_sp <- single_prediction(rf_expl, houses[n_obs, -3])
svm_expl_sp <- single_prediction(svm_expl, houses[n_obs, -3])
lm_expl_sp  <-single_prediction(lm_expl, houses[n_obs, -3])

plot(rf_expl_sp,
     svm_expl_sp,
     lm_expl_sp)
```



Exercise 3

```r
library(shapleyr)
```

```
## Loading required package: checkmate

## Loading required package: combinat

##
## Attaching package: 'combinat'

## The following object is masked from 'package:utils':
##
##     combn

## Loading required package: shiny

## Loading required package: shinydashboard

##
## Attaching package: 'shinydashboard'

## The following object is masked from 'package:graphics':
##
##     box

## Loading required package: reshape2

##
## Attaching package: 'reshape2'

## The following object is masked from 'package:tidyr':
##
##     smiths

## Welcome to the ShapleyR package!
```

```r
shapley_vals <- shapley(n_obs, house_task, house_rf_mlr)

gather(shapley_vals$values, "colname", "contribution") %>%
  filter(colname %in% colnames(houses)) %>%
  mutate(contribution = as.numeric(contribution)) %>%
  arrange(desc(abs(contribution)))
```

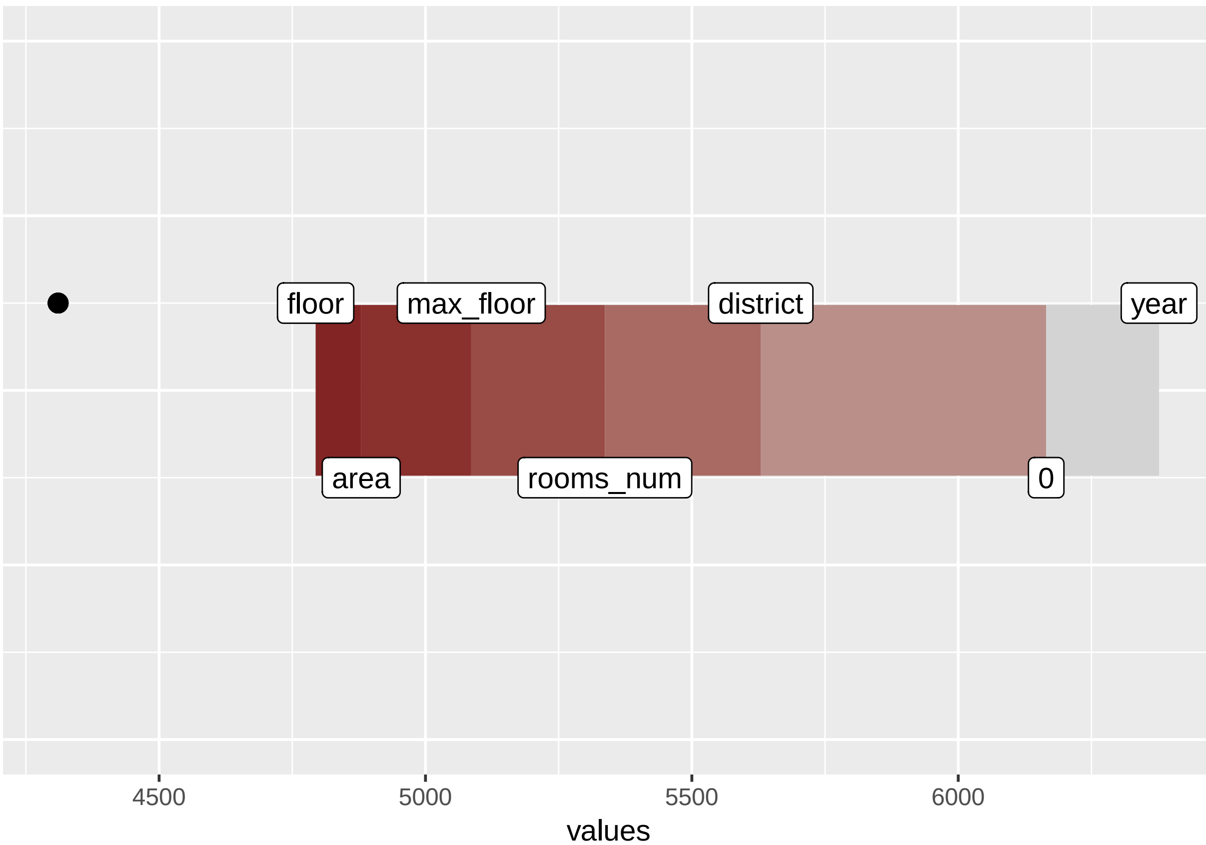
```
##     colname contribution
## 1  district     -535.795
## 2 rooms_num     -292.509
## 3 max_floor     -250.853
## 4      year      211.827
## 5      area     -206.529
## 6     floor      -85.600
```

```r
# alternatively use just
shapley_vals
```

```
## $task.type
## [1] "regr"
##
## $feature.names
## [1] "rooms_num" "area"      "year"      "floor"     "max_floor" "district"
##
```

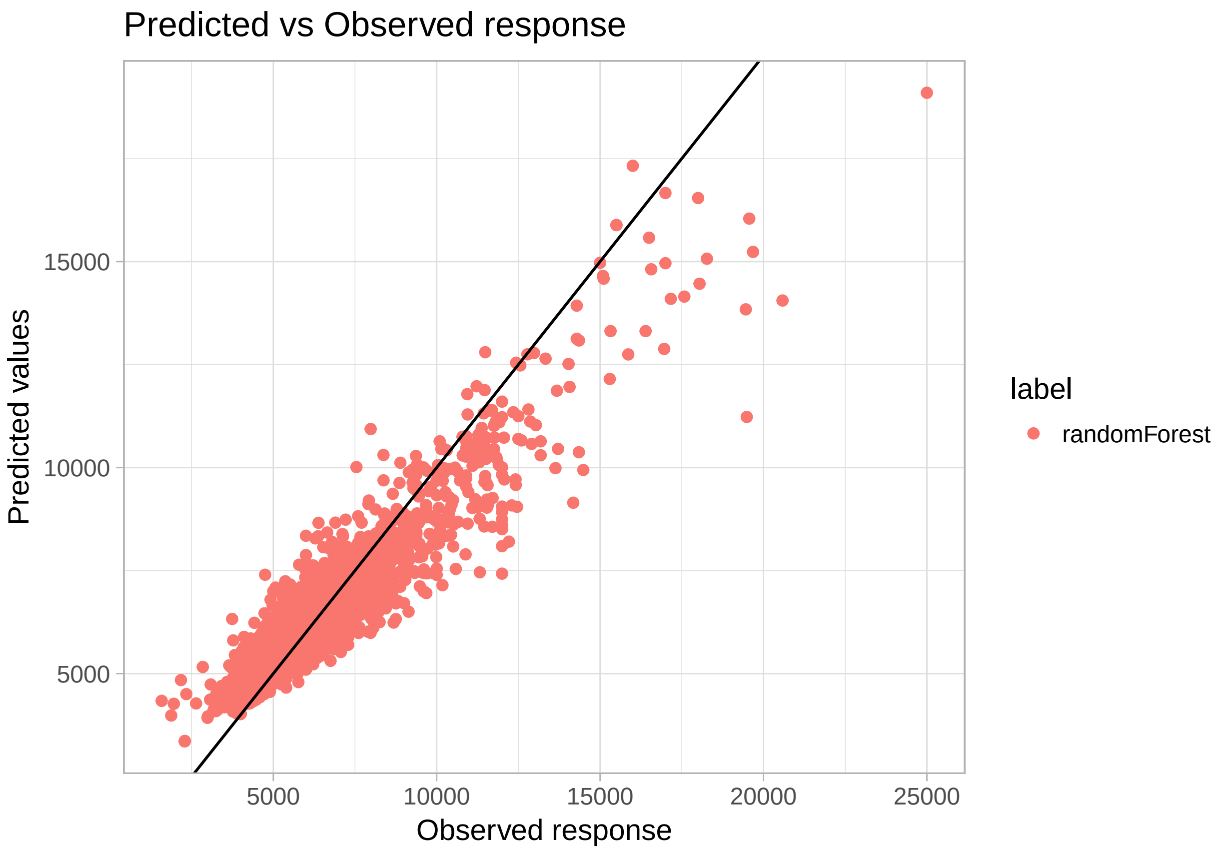
```
## $predict.type
## [1] "response"
##
## $prediction.response
## [1] 4310.482
##
## $data.mean
## [1] 6165.112
##
## $values
##     _Id _Class rooms_num     area     year floor max_floor district
## 1 1189     NA  -292.509 -206.529 211.827 -85.6  -250.853 -535.795
```

```r
class(shapley_vals) <- c("shapley.singleValue", "list")
plot(shapley_vals)
```

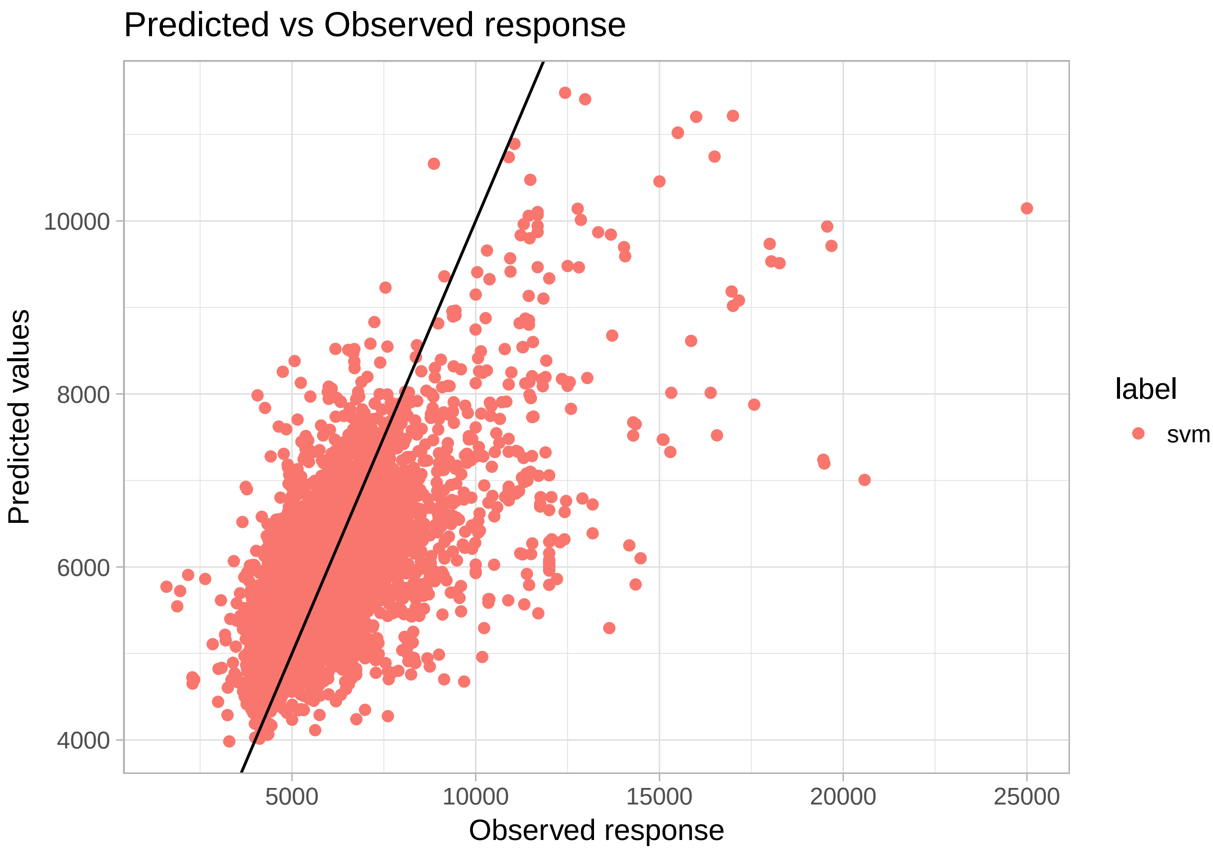


Bonus 1

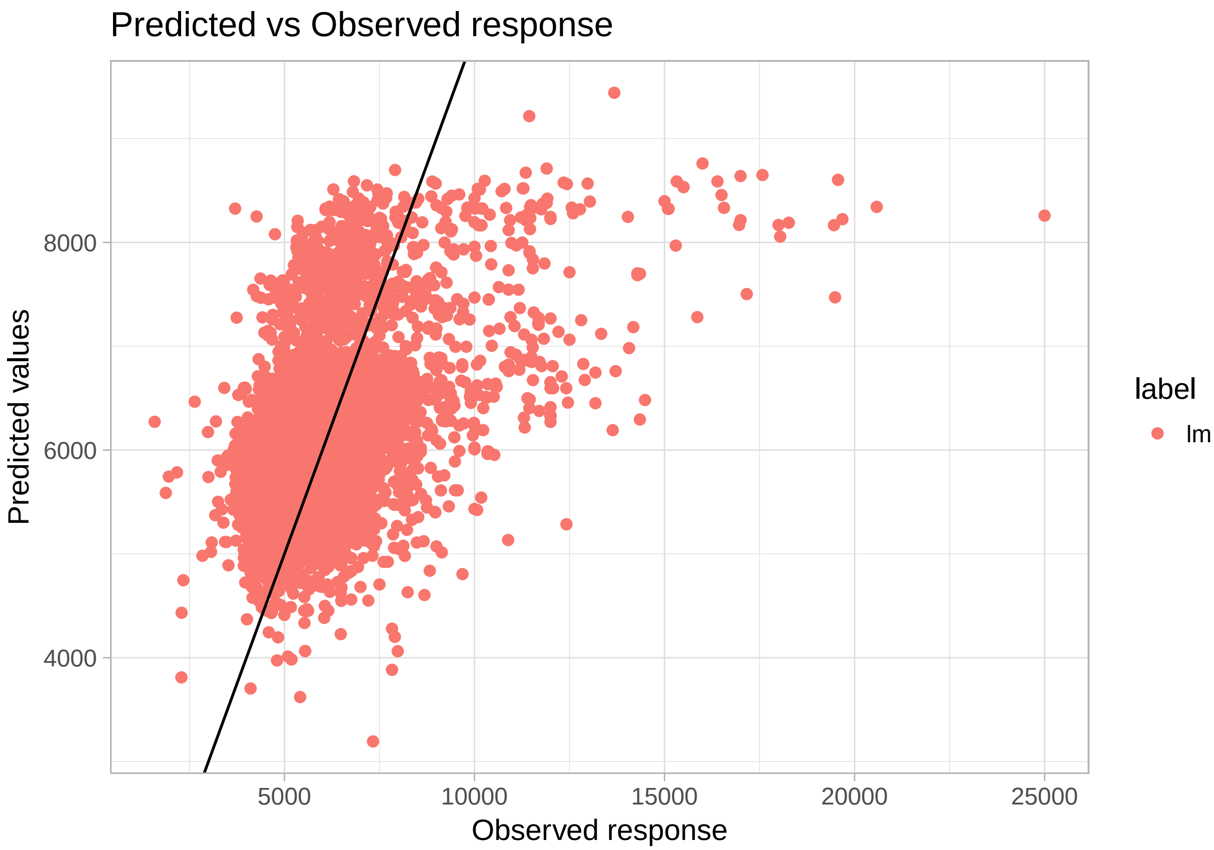
```r
rf_audit <- audit(rf_expl)
svm_audit <- audit(svm_expl)
lm_audit <- audit(lm_expl)

plotPrediction(rf_audit)
```




```r
plotPrediction(svm_audit)
```

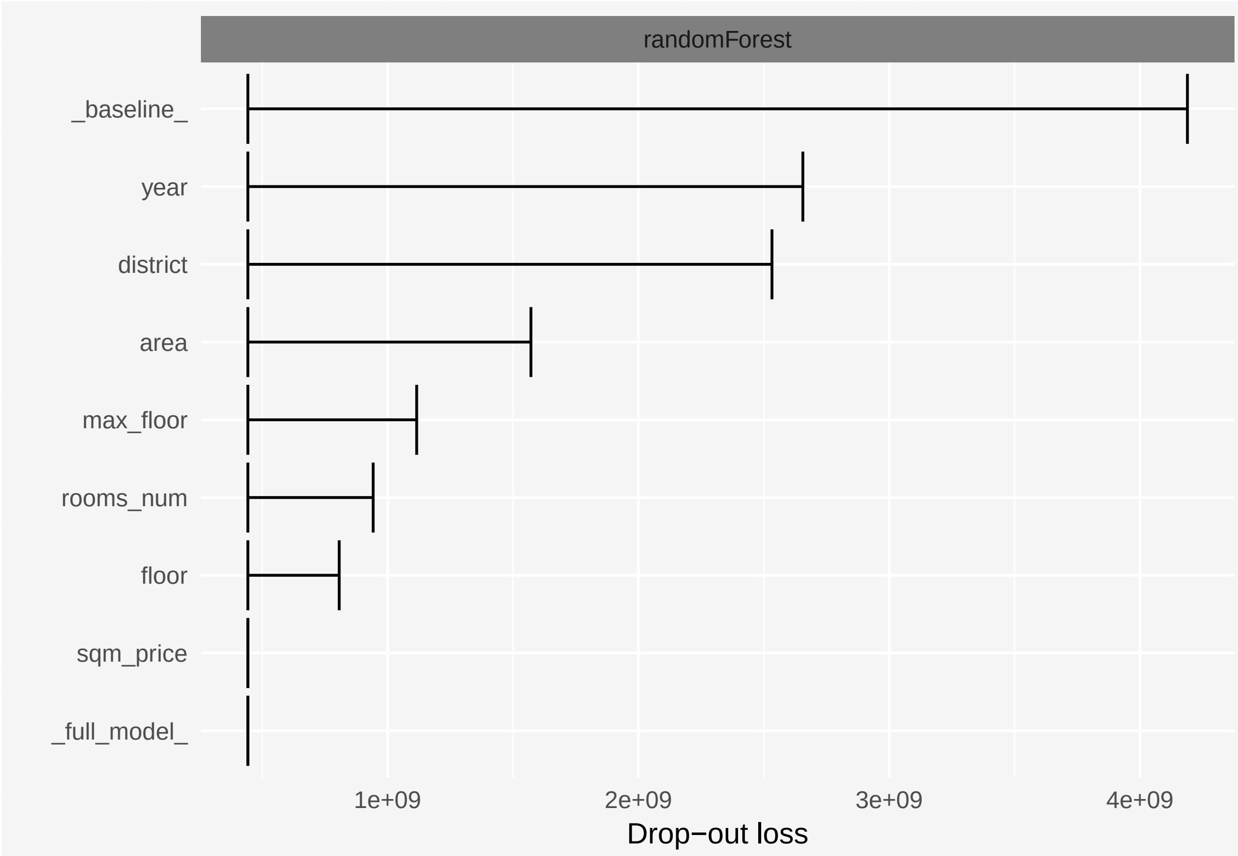



```r
plotPrediction(lm_audit)
```



Bonus 2

```
rf_global <- variable_importance(rf_expl)
plot(rf_global)
```



Exercise 4

```r
library(live)
library(mlr)

houses_similar <- sample_locally2(houses, houses[n_obs, ], "sqm_price", 1000)
houses_similar2 <- add_predictions2(houses_similar, house_rf)
lm_approx <- fit_explanation2(houses_similar2, "regr.lm")

lm_approx
```

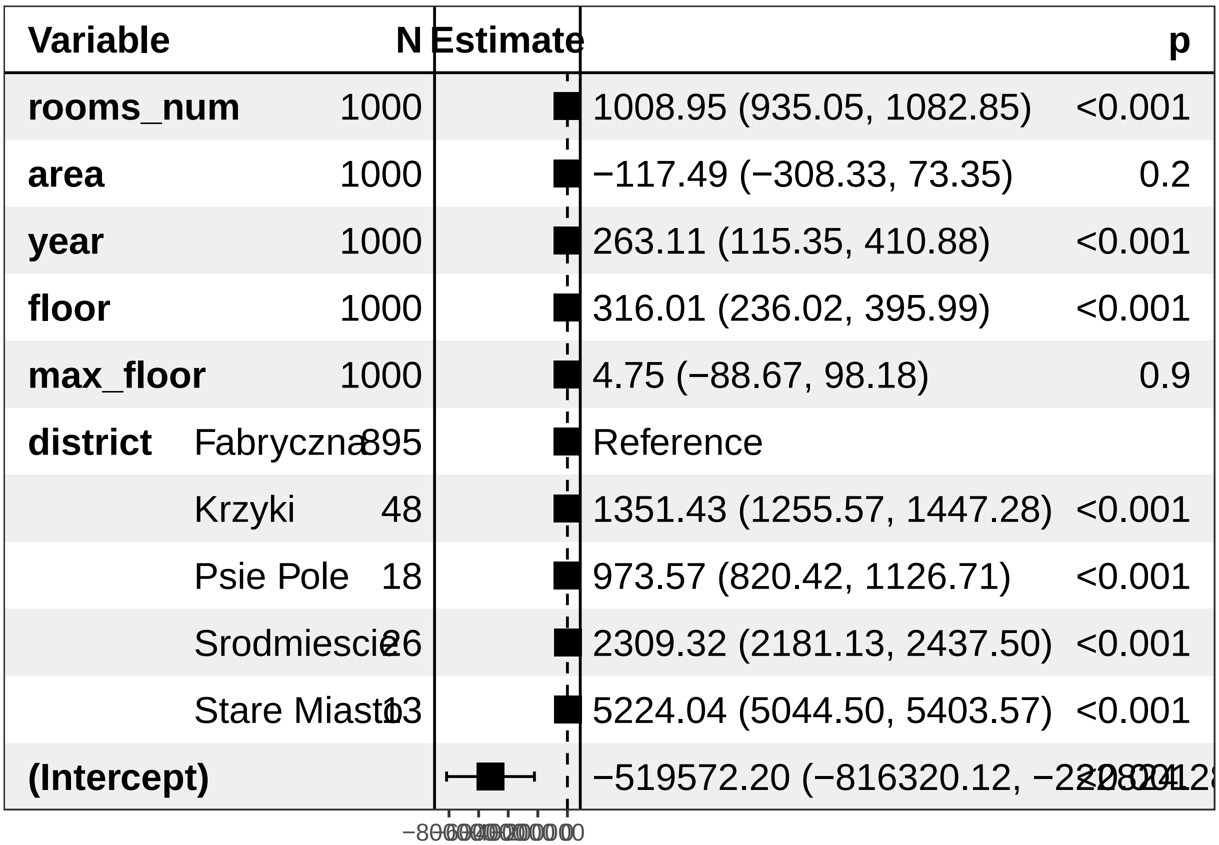
```
## Warning in summary.lm(model_tmp): essentially perfect fit: summary may be
## unreliable
```

```
## Dataset:
##  Observations:  1000
##  Variables:  7
##  Response variable:  sqm_price
## Explanation model:
##  Name:  regr.lm
##  Variable selection wasn't performed
##  Weights present in the explanation model
##  R-squared: 0.8633
```

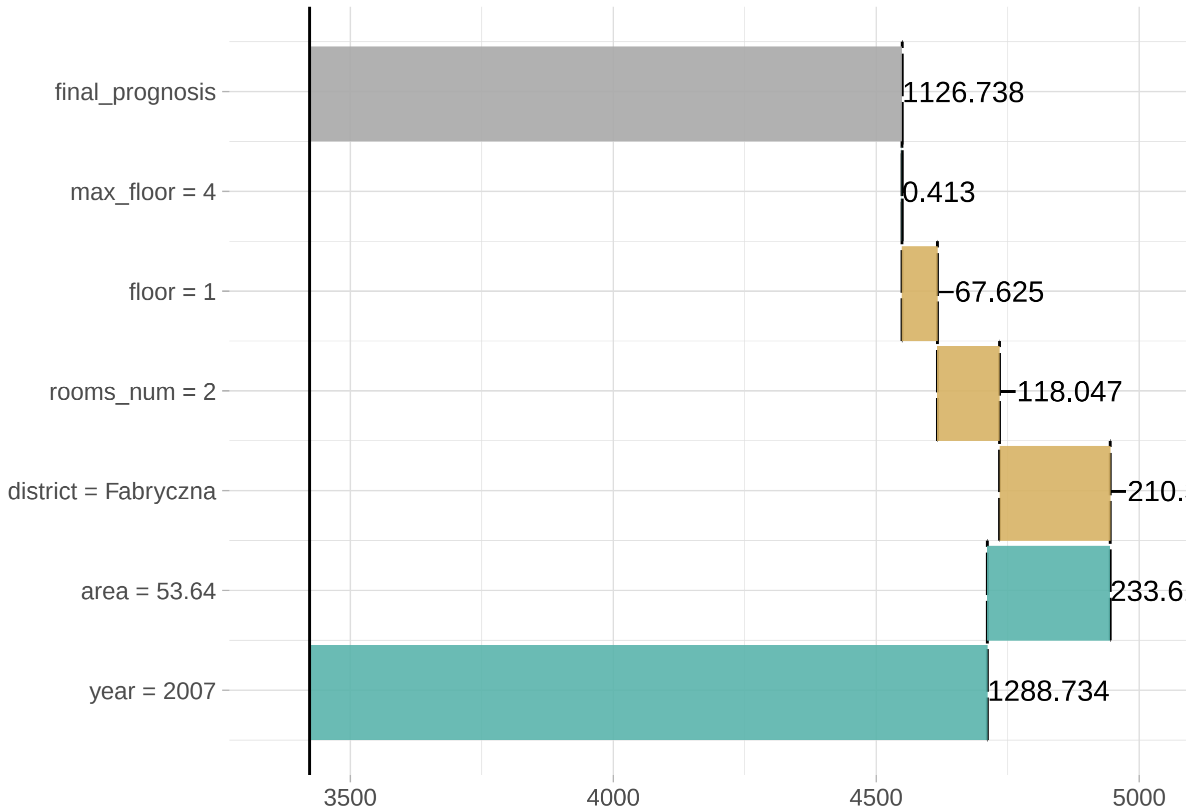
Exercise 5

```r
plot_explanation2(lm_approx, regr_plot_type = "forest")
```

```
## Warning in summary.lm(x): essentially perfect fit: summary may be
## unreliable

## Warning in summary.lm(object): essentially perfect fit: summary may be
## unreliable

## Warning in recalculate_width_panels(panel_positions, mapped_text =
## mapped_text, : Unable to resize forest panel to be smaller than its
## heading; consider a smaller text size
```




```r
plot_explanation2(lm_approx, regr_plot_type = "waterfall")
```

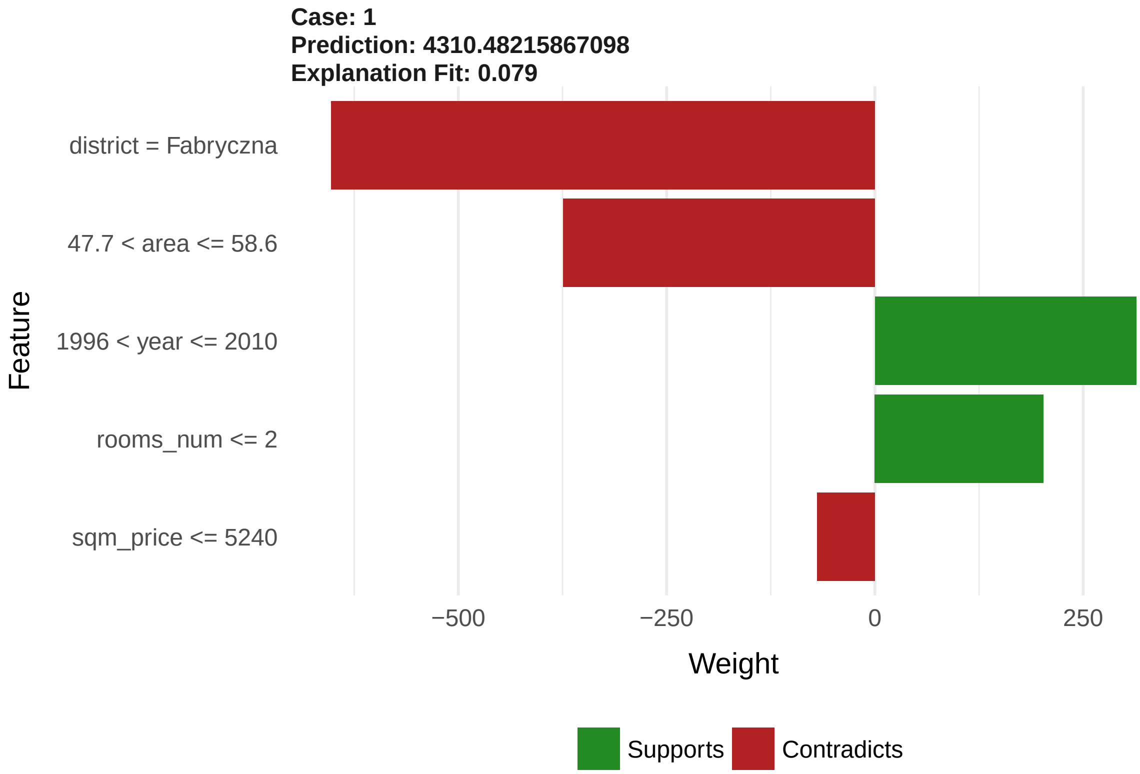


Exercise 6

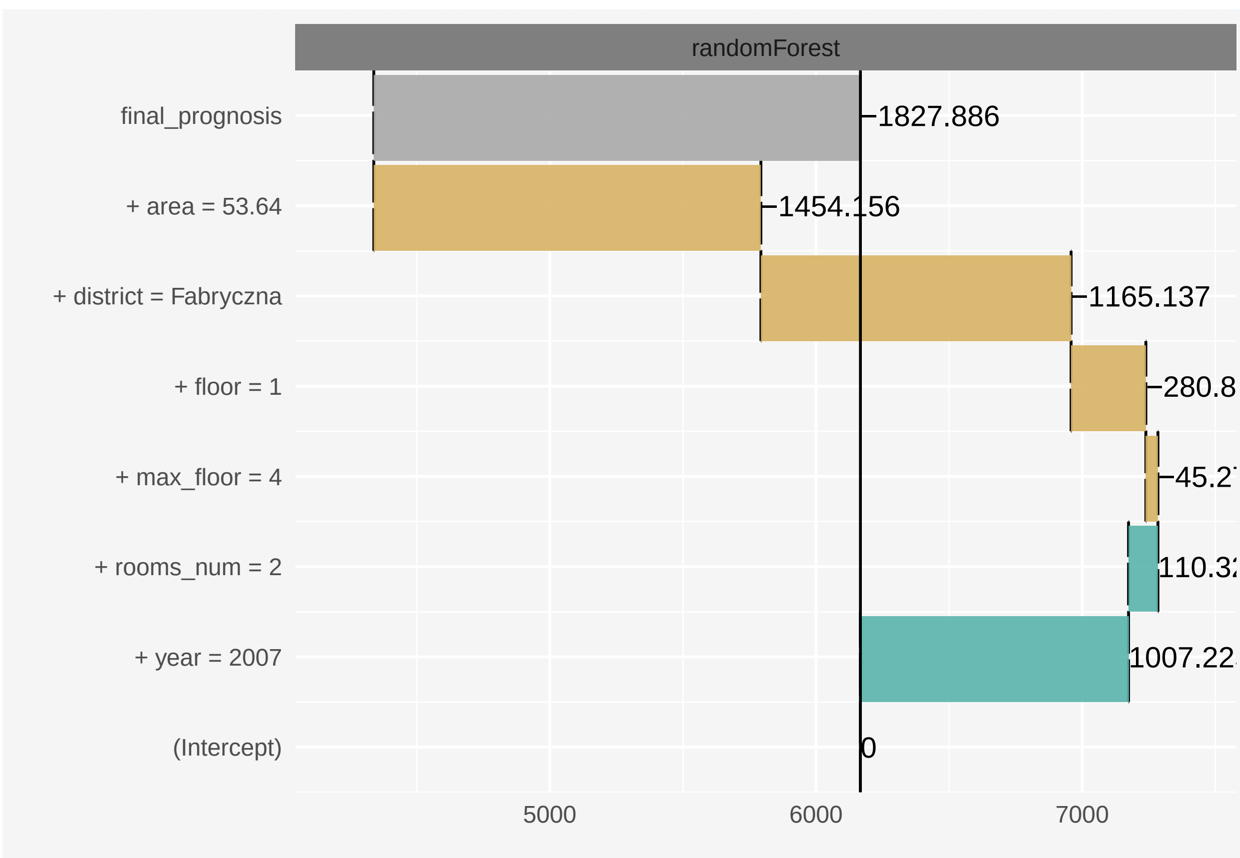
```
library(lime)
```

```
##
## Attaching package: 'lime'

## The following object is masked from 'package:DALEX':
##
##     explain

## The following object is masked from 'package:dplyr':
##
##     explain
```

```
lime_rf <- lime(houses, house_rf_mlr)
lime_explanation <- lime::explain(houses[n_obs, ], lime_rf, n_features = 5)
plot_features(lime_explanation)
```

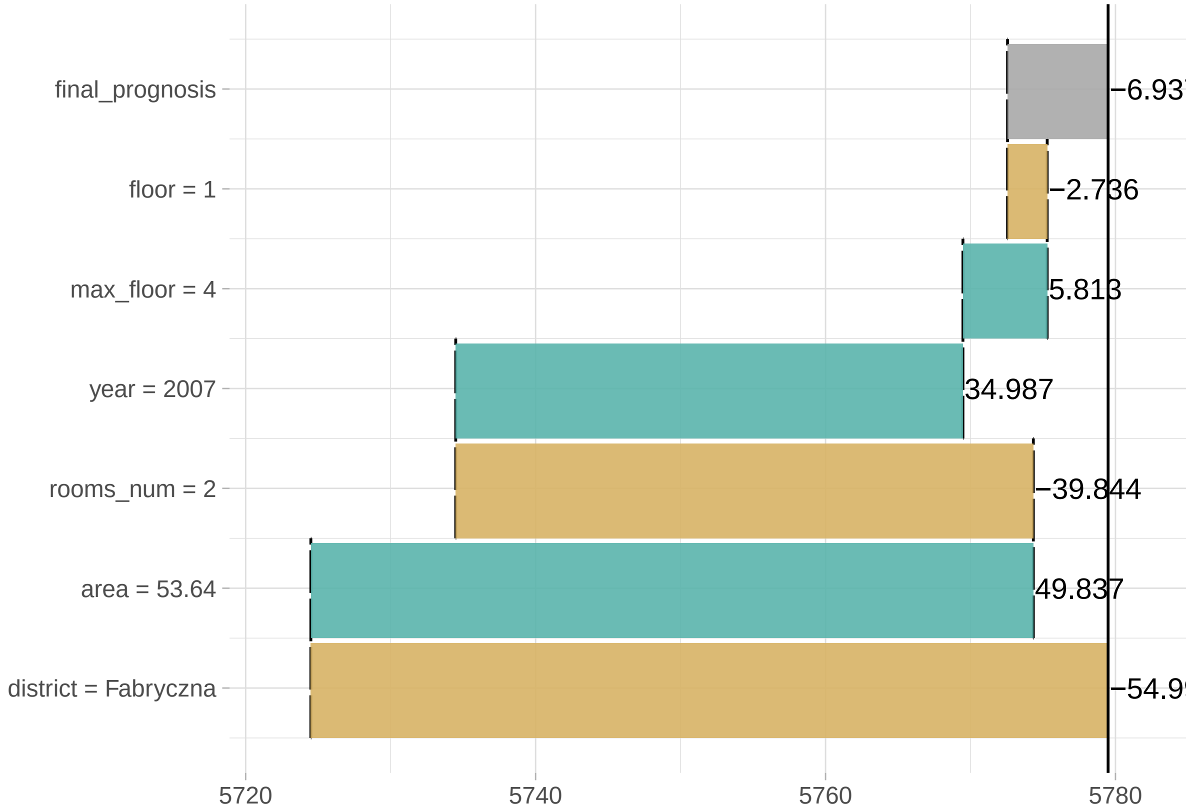



```
plot(rf_expl_sp)
```

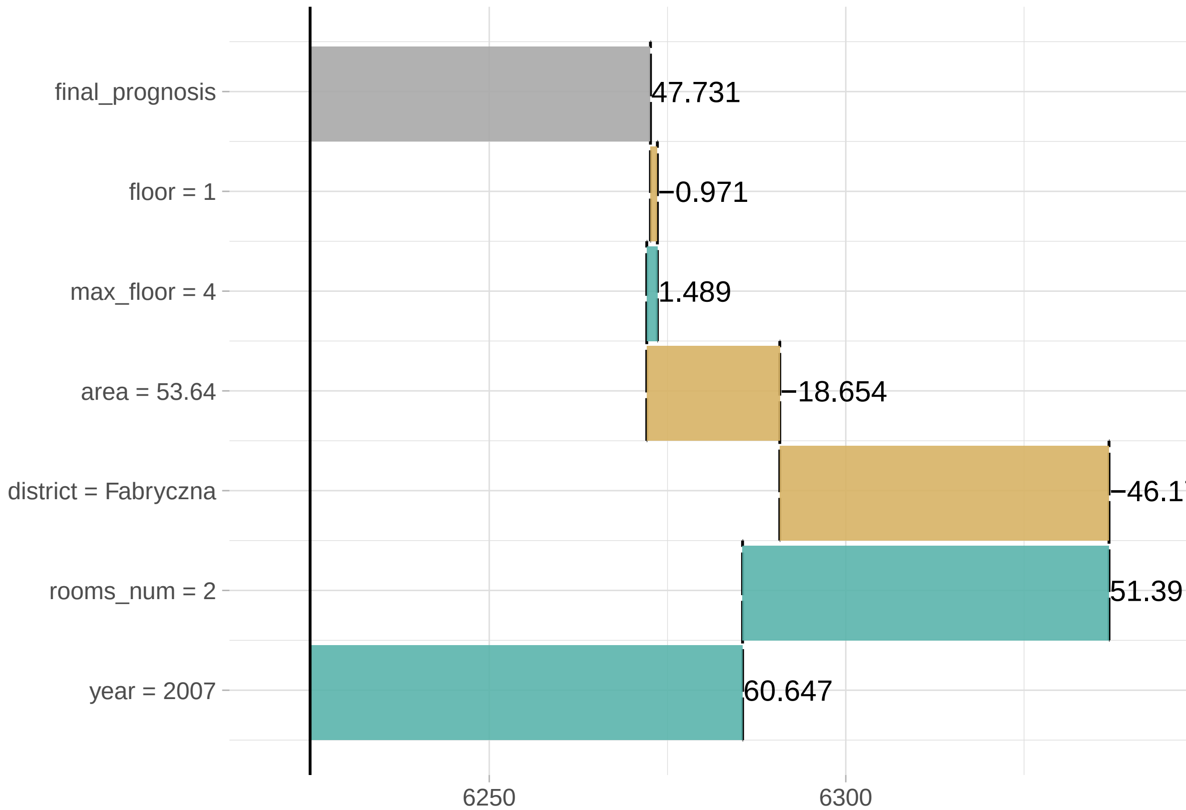


Bonus 3

```
houses_similar3 <- add_predictions2(houses_similar, house_svm)
houses_similar4 <- add_predictions2(houses_similar, house_lm)
lm_approx2 <- fit_explanation2(houses_similar3)
lm_approx3 <- fit_explanation2(houses_similar4)

plot_explanation2(lm_approx2, "waterfall")
```

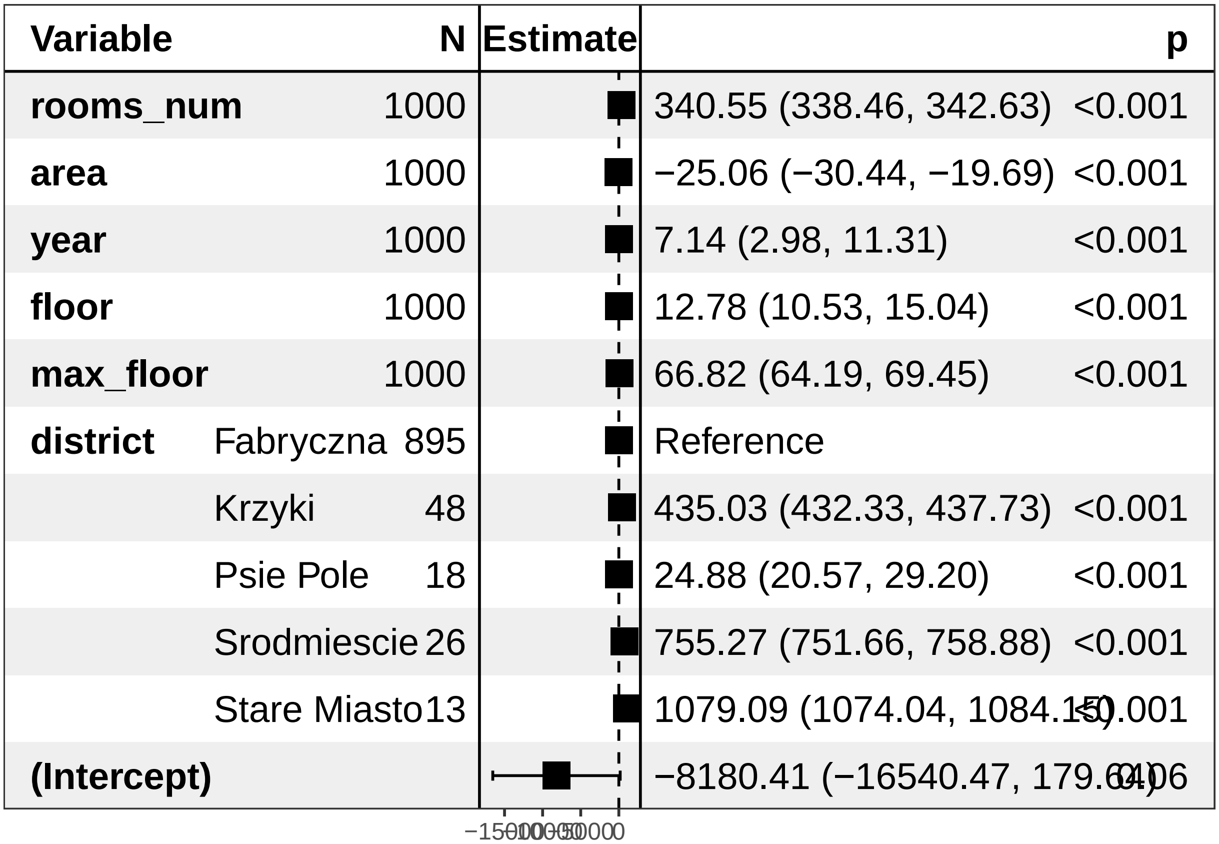



```
plot_explanation2(lm_approx3, "waterfall")
```




```
plot_explanation2(lm_approx2, "forest")
```

```
## Warning in summary.lm(x): essentially perfect fit: summary may be
## unreliable

## Warning in summary.lm(object): essentially perfect fit: summary may be
## unreliable

## Warning in recalculate_width_panels(panel_positions, mapped_text =
## mapped_text, : Unable to resize forest panel to be smaller than its
## heading; consider a smaller text size
```




```r
plot_explanation2(lm_approx3, "forest")
```

```
## Warning in summary.lm(x): essentially perfect fit: summary may be
## unreliable

## Warning in summary.lm(object): essentially perfect fit: summary may be
## unreliable

## Warning in recalculate_width_panels(panel_positions, mapped_text =
## mapped_text, : Unable to resize forest panel to be smaller than its
## heading; consider a smaller text size
```

Variable		N	Estimate		p
rooms_num		1000		−439.23 (−439.23, −439.23)	<0.001
area		1000		9.38 (9.38, 9.38)	<0.001
year		1000		12.38 (12.38, 12.38)	<0.001
floor		1000		4.54 (4.54, 4.54)	<0.001
max_floor		1000		17.12 (17.12, 17.12)	<0.001
district	Fabryczna	895		Reference	
	Krzyki	48		277.96 (277.96, 277.96)	<0.001
	Psie Pole	18		−412.57 (−412.57, −412.57)	<0.001
	Srodmiescie	26		569.83 (569.83, 569.83)	<0.001
	Stare Miasto	13		1956.82 (1956.82, 1956.82)	<0.001
(Intercept)				−18275.81 (−18275.81, −18275.81)	<0.001

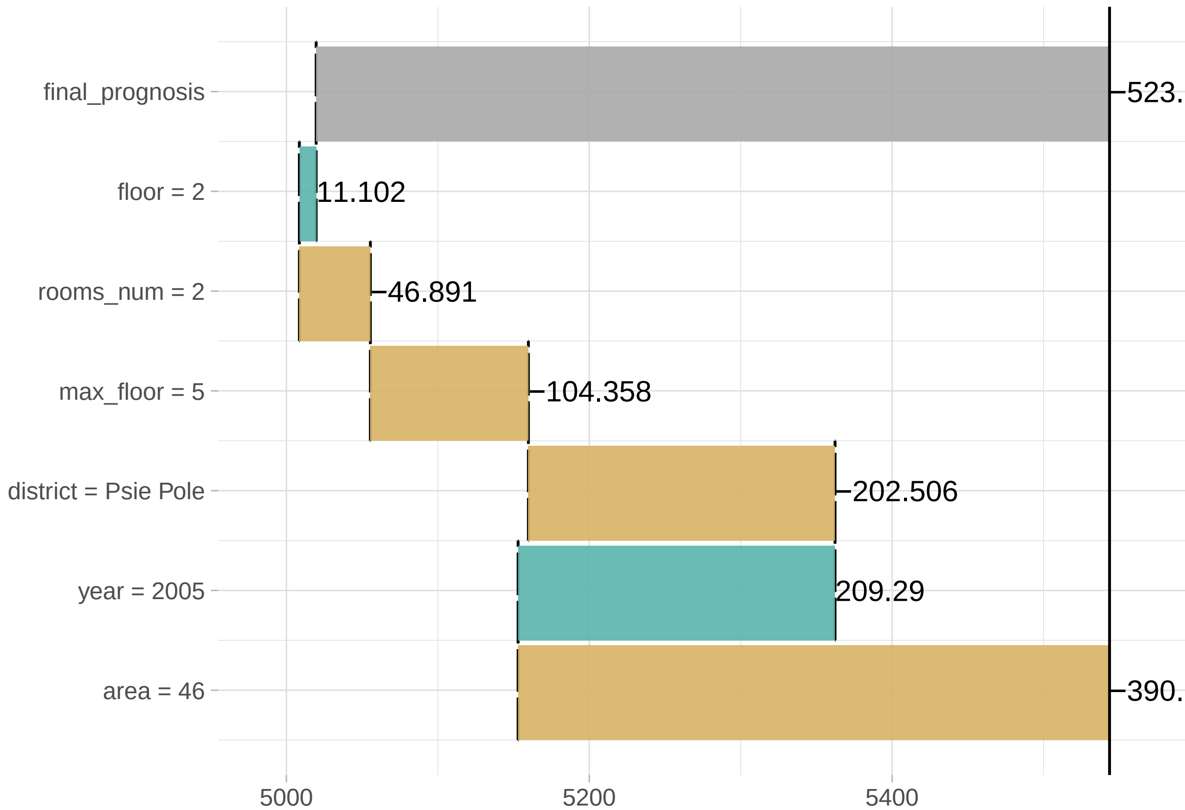
Bonus 4

```r
library(live)
library(mlr)
n_obs2 <- 5830
houses_similar <- sample_locally2(houses, houses[n_obs2, ], "sqm_price", 1000)
houses_similar2 <- add_predictions2(houses_similar, house_rf)
n_obs2expl <- fit_explanation2(houses_similar2)
n_obs2expl
```

```
## Warning in summary.lm(model_tmp): essentially perfect fit: summary may be
## unreliable
```

```
## Dataset:
##  Observations:  1000
##  Variables:  7
##  Response variable:  sqm_price
## Explanation model:
##  Name:  regr.lm
##  Variable selection wasn't performed
##  Weights present in the explanation model
##  R-squared: 0.8071
```

```r
plot_explanation2(n_obs2expl, "waterfall")
```




```
plot_explanation2(lm_approx, regr_plot_type = "waterfall")
```