R Notebook

Verena Haunschmid

Loading data, libraries, ...

```r
library(readr)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
library(magrittr)
library(lime)
```

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

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

```r
library(caret)
```

```
## Loading required package: lattice

## Loading required package: ggplot2
```

```r
path_train <- "C:/Users/veroa/OneDrive/PhD/Data/kaggle-house-prices/train.csv"

data_train <- read_csv(path_train)
```

```
## Parsed with column specification:
## cols(
##   .default = col_character(),
##   Id = col_integer(),
##   MSSubClass = col_integer(),
##   LotFrontage = col_integer(),
##   LotArea = col_integer(),
##   OverallQual = col_integer(),
##   OverallCond = col_integer(),
##   YearBuilt = col_integer(),
##   YearRemodAdd = col_integer(),
##   MasVnrArea = col_integer(),
##   BsmtFinSF1 = col_integer(),
##   BsmtFinSF2 = col_integer(),
##   BsmtUnfSF = col_integer(),
##   TotalBsmtSF = col_integer(),
```

```
##   `1stFlrSF` = col_integer(),
##   `2ndFlrSF` = col_integer(),
##   LowQualFinSF = col_integer(),
##   GrLivArea = col_integer(),
##   BsmtFullBath = col_integer(),
##   BsmtHalfBath = col_integer(),
##   FullBath = col_integer()
##   # ... with 18 more columns
## )

## See spec(...) for full column specifications.
```

```
colnames(data_train) <- tolower(colnames(data_train))
data_train <- data_train %>% mutate_if(is.character, factor)
```

Split the data into training and test data. I also converted the tibble to a data.frame because at the beginning I had some issues and an answer on stackoverflow can state that caret gets confused with the class `tibble`.

```
set.seed(0987)
idx <- sample(1:nrow(data_train), 0.8 * nrow(data_train), replace = FALSE)
x_train <- data.frame(data_train[idx, ])
x_test <- data.frame(data_train[-idx, ])
```

Then I prepared the features I wanted to use. I removed `mssubclass` and `mszoning` since I couldn't figure out what they were. And I also removed all features that had NA's in them. There are better ways to deal with NA's but I wanted to keep it as simple as possible.

```
features_na <- colnames(x_train)[unique(which(is.na(x_train), arr.ind = TRUE)[, 2])]
features <- colnames(x_train[,!(colnames(x_train) %in% c("id", "mssubclass", "mszoning", "saleprice"))])
features <- features[!(features %in% features_na)]
target <- "saleprice"
```

This left me with

```
length(features)
```

```
## [1] 58
```

features.

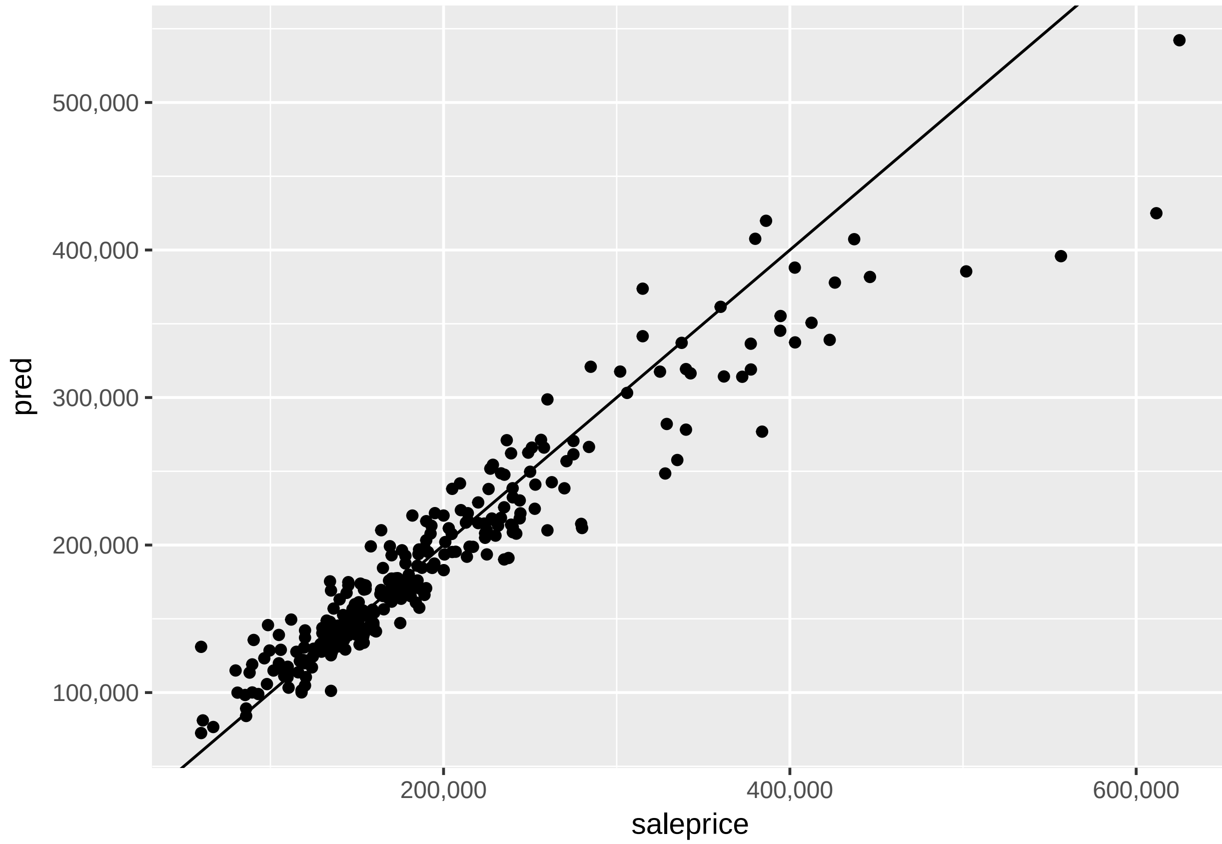
```
rf_model <- train(x_train[,features], x_train[,target], method='rf')
```

To save some time I've stored the model on disk:

```
# load model, train idx
load("rf_model.RData")
```

Model performance

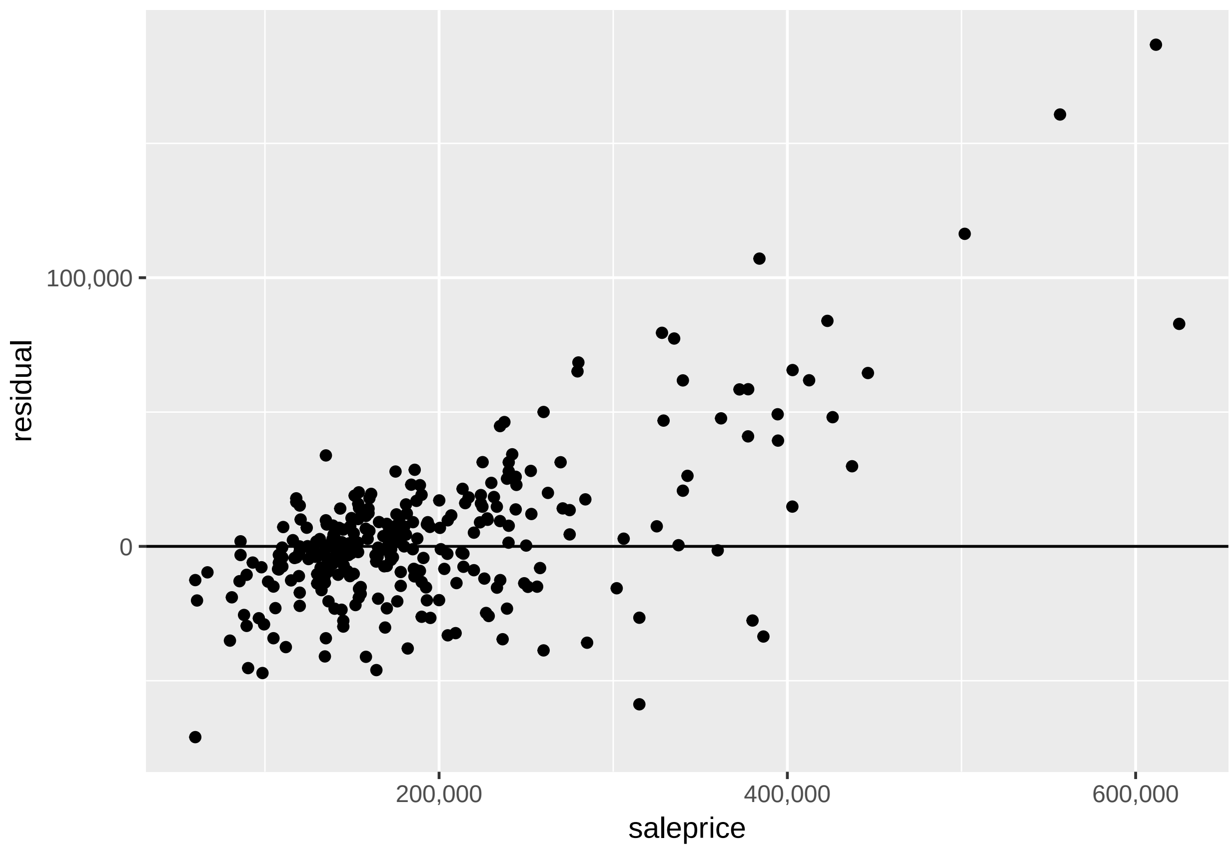
```
preds <- predict(rf_model, x_test)

ggplot(data.frame("saleprice"=x_test$saleprice, "pred"=preds), aes(saleprice, pred)) +
  geom_point() +
  geom_abline(slope=1, intercept=0) +
  scale_y_continuous(labels = scales::comma) +
  scale_x_continuous(labels = scales::comma)
```



Residual plot

This plot shows that the residuals are not uniformly distributed and that the model performs worse on more expensive houses.

```
ggplot(data.frame("saleprice"=x_test$saleprice, "residual"=x_test$saleprice - preds),
       aes(saleprice, residual)) +
  geom_point() +
  geom_abline(slope=0, intercept=0) +
  scale_y_continuous(labels = scales::comma) +
  scale_x_continuous(labels = scales::comma)
```



Building an explainer

For building an explainer you pass the **training data** and the model to the function `lime()`.

```
explainer <- lime(x_train, rf_model)
```

Explaining selecting samples

```
x_expl <- x_test[1:8,]
explanation <- explain(x_expl, explainer, n_features = 2)
```

```
head(explanation)
```

```
##   model_type case   model_r2 model_intercept model_prediction    feature
## 1 regression    1 0.01804252        182683.8         171027.0 electrical
## 2 regression    1 0.01804252        182683.8         171027.0  grlivarea
## 3 regression    2 0.18974200        169980.9         207139.9  bsmtunfsf
## 4 regression    2 0.18974200        169980.9         207139.9  grlivarea
## 5 regression    3 0.19910279        169786.0         210048.7 fireplaces
## 6 regression    3 0.19910279        169786.0         210048.7  grlivarea
##   feature_value feature_weight              feature_desc
## 1             5       413.4585       electrical = SBrkr
## 2          1262    -12070.2307 1114 < grlivarea <= 1454
```

```
## 3           434      -790.9498   216 < bsmtunfsf <= 468
## 4          1786     37950.0359         1764 < grlivarea
## 5             2       659.6091           1 < fireplaces
## 6          2090     39603.0166         1764 < grlivarea
##
## 1             2, 20, 4, 80, 9600, 2, NA, 4, 4, 1, 3, 1, 25, 2, 3, 1, 3, 6, 8, 1976, 1976, 2, 2, 9, 9
## 2             2, 20, 4, 80, 9600, 2, NA, 4, 4, 1, 3, 1, 25, 2, 3, 1, 3, 6, 8, 1976, 1976, 2, 2, 9, 9
## 3          3, 60, 4, 68, 11250, 2, NA, 1, 4, 1, 5, 1, 6, 3, 3, 1, 6, 7, 5, 2001, 2002, 2, 2, 13, 14,
## 4          3, 60, 4, 68, 11250, 2, NA, 1, 4, 1, 5, 1, 6, 3, 3, 1, 6, 7, 5, 2001, 2002, 2, 2, 13, 14,
## 5 8, 60, 4, NA, 10382, 2, NA, 1, 4, 1, 1, 1, 17, 5, 3, 1, 6, 7, 6, 1973, 1973, 2, 2, 7, 7, 4, 240, 4
## 6 8, 60, 4, NA, 10382, 2, NA, 1, 4, 1, 1, 1, 17, 5, 3, 1, 6, 7, 6, 1973, 1973, 2, 2, 7, 7, 4, 240, 4
##   prediction
## 1   169271.1
## 2   169271.1
## 3   214555.3
## 4   214555.3
## 5   220004.6
## 6   220004.6
```

Visualising the model explanations

```r
plot_features(explanation)
```

Using scaled data

```r
x_train_sc <- x_train
num_feat <- sapply(x_train_sc, is.numeric)
num_feat[names(num_feat) == "saleprice"] <- FALSE

sc_center <- apply(x_train_sc[,num_feat], 2, mean)
sc_sd <- apply(x_train_sc[,num_feat], 2, sd)

x_train_sc[,num_feat] <- scale(x_train_sc[,num_feat])

x_test_sc <- x_test
x_test_sc[,num_feat] <- scale(x_test_sc[,num_feat], center = sc_center, scale = sc_sd)
```

```r
rf_model_sc <- train(x_train_sc[,features], x_train_sc[,target], method='rf')
```

```r
load('rf_model_sc.RData')
```

```r
explainer_sc <- lime(x_train_sc, rf_model_sc)
x_expl_sc <- x_test_sc[1:8,]

explanation_sc <- explain(x_expl_sc, explainer_sc, n_features = 3)

plot_features(explanation_sc)
```