09 Machine Learning mit caret

Martin Hanewald

2019-02-19

Packages

```
library(tidyverse)
library(caret)
library(knitr)
library(DT)
library(GGally)
library(Metrics)
```

Überblick

Das `caret` Paket ist der einfachste Weg um eine Vielzahl von Machine-Learning Techniken anzuwenden. Es verfügt über ein einheitliches Interface um zahlreiche Modelle aus anderen Paketen anzusprechen.

Die zentralen FUnktionen sind:

- `createDataPartition()`: Splitting in Training- und Testset
- `train()`: Training von Modellen
- `trainControl()`: Steuerung der Validierungsmethodik und Parametertuning
- `preProcess()`: Optionales Pre-Processing der Daten (Centering, Scaling, etc.)
- `predict()`: Erzeugung der Prediction
- `resamples()`: Erzeugung von Cross-Validation Metriken mit `summary()`und `bwplot()`

Dataset

Wir betrachten einen mitgelieferten Datensatz über den Zusammenhang zwischen Häuserpreisen (`price`) und verschiedenen Hauseigenschaften wie Größe, Lage, Anzahl Badezimmer, etc.

```
data(Sacramento, package='caret')
Sacramento <- Sacramento %>% as_tibble()
# für scrollbaren HTML Output
Sacramento %>% sample_n(20) %>% datatable(options = list(scrollX=T))
```

Show 10 entries Search:

	city	zip	beds	baths	sqft	type	price	latitude
1	GOLD_RIVER	z95670	2	2	1520	Residential	299000	38.62869
2	CITRUS_HEIGHTS	z95621	4	2	1280	Residential	167293	38.715781
3	SACRAMENTO	z95864	3	1	1643	Residential	99000	38.588672
4	GALT	z95632	5	4	3746	Residential	420000	38.271646
5	SACRAMENTO	z95826	3	2	1280	Residential	192067	38.560767
6	SACRAMENTO	z95842	4	2	1578	Residential	195000	38.693071
7	SACRAMENTO	z95841	3	2	1120	Residential	178000	38.658734

8	SACRAMENTO	z95815	2	1	1011	Residential	85000	38.6238
9	SACRAMENTO	z95818	2	1	1144	Residential	299000	38.556844
10	CITRUS_HEIGHTS	z95621	3	2	1343	Residential	284893	38.715853

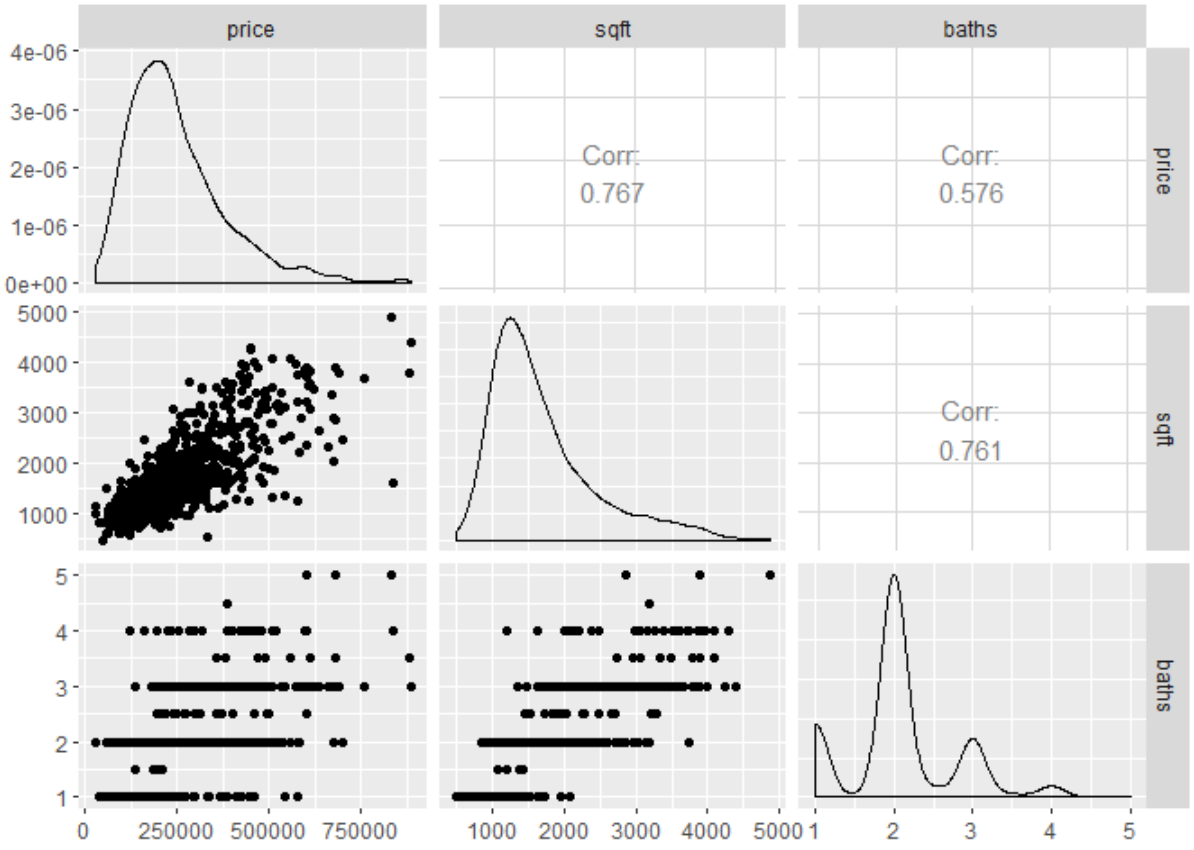
Showing 1 to 10 of 20 entries

Previous 1 2 Next

Exploration

Ein einfacher Pairs Plot zwischen zwei ausgewählten Variabeln `sqft` und `baths` zeigt eine deutliche positivte Korrelation.

```
Sacramento %>% select(price, sqft, baths) %>% ggpairs()
```



Splitting und Training

Wir splitten den Datensatz gemäß der 80/20 Faustregel.

```
in_train <- createDataPartition(Sacramento$price, p = .8, list = FALSE)

training <- Sacramento[ in_train,]
testing  <- Sacramento[-in_train,]
```

Danach fitten wir ein einfaches lineares Modell sowie einen Decision Tree.

Als Sampling Methode wählen wir die Cross-Validation (method='cv') und einige Pre-Processing Schritte für numerische Variablen. Als einzigen Prädiktor wählen wir zunächst nur die Variable `sqft`.

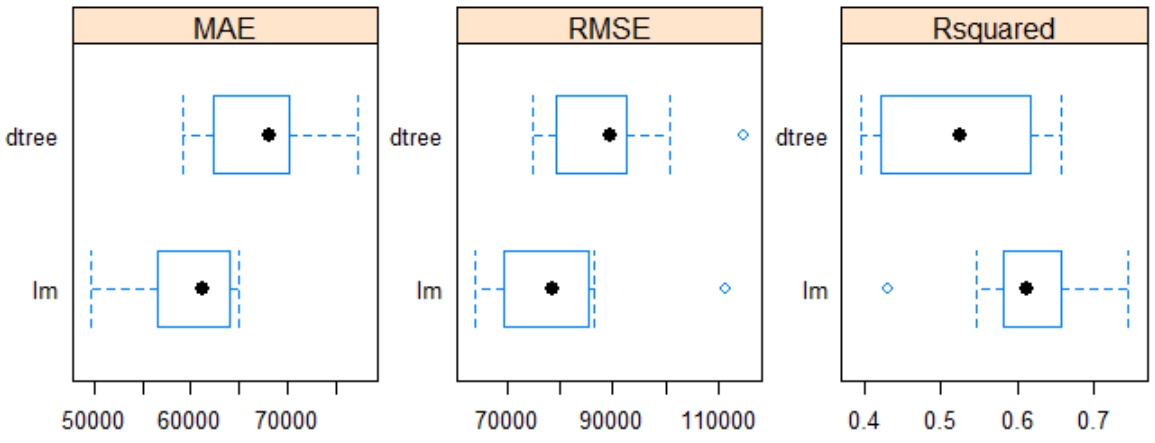
```
fit_lm <- train(price ~ sqft,
             trControl = trainControl(method='cv'),
             preProcess = c('center', 'scale', 'zv'),
             method = 'lm',
             data=training)
```

```
fit_dtree <- train(price ~ sqft,
             trControl = trainControl(method='cv'),
             preProcess = c('center', 'scale', 'zv'),
             method = 'rpart',
             data=training)
#> Warning in nominalTrainWorkflow(x = x, y = y, wts = weights, info =
#> trainInfo, : There were missing values in resampled performance measures.
```

Evaluation

Die Evaluation anhand der Kreuzvalidierungssamples zeigt das lineare Modell eindeutig im Vorteil.

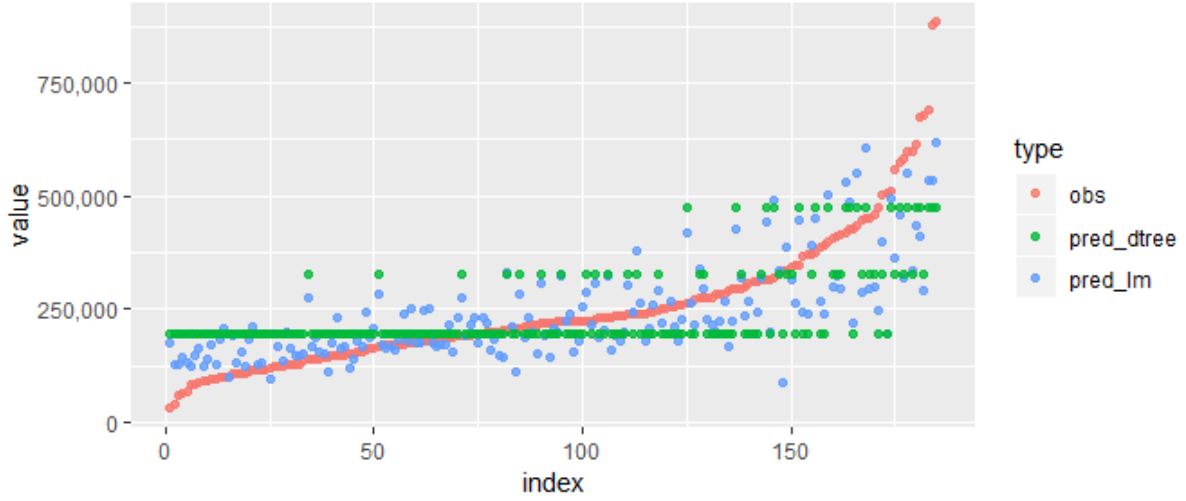
```
res <- resamples(list(lm = fit_lm, dtree = fit_dtree))
res %>% summary()
#>
#> Call:
#> summary.resamples(object = .)
#>
#> Models: lm, dtree
#> Number of resamples: 10
#>
#> MAE
#>           Min.  1st Qu.   Median     Mean  3rd Qu.     Max. NA's
#> lm    49662.66 56608.57 61138.46 59482.88 63998.12 64976.63    0
#> dtree 59116.95 62521.25 68131.29 67158.75 69759.41 77328.92    0
#>
#> RMSE
#>           Min. 1st Qu.   Median     Mean  3rd Qu.     Max. NA's
#> lm    63948.38 70316.6 78445.73 79636.29 85436.34 111279.2    0
#> dtree 74841.75 79334.8 89396.70 88917.08 92501.02 114739.1    0
#>
#> Rsquared
#>            Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
#> lm    0.4305718 0.5831811 0.6119248 0.6102292 0.6510016 0.7443224    0
#> dtree 0.3947227 0.4225706 0.5244397 0.5151145 0.6057425 0.6563568    0
res %>% bwplot(scales='free')
```



Auch die visuelle Inspektion der Prediction auf Basis des Testdatensatzes zeigt die Schwächen des Decision Tree Ansatzes.

```
ans <- tibble(obs = testing$price,
              pred_lm = predict(fit_lm, testing),
              pred_dtree = predict(fit_dtree, testing))

ans %>% arrange(obs) %>%
    mutate(index = row_number(obs)) %>%
    gather(type, value, -index) %>%
    ggplot(aes(x=index, y =value, color=type)) + geom_point(alpha=.8) +
    scale_y_continuous(labels = scales::comma)
```



Abschließende Beurteilung der Güte auf dem Testset anhand der Performance-Kennzahl MAPE (Mean Absolute Percentage Error).

```
ans %>%
    summarise(mape_lm = mape(obs, pred_lm),
              mape_dtree = mape(obs, pred_dtree),
              mase_lm = mase(obs, pred_lm),
              mase_dtree = mase(obs, pred_dtree))
#> # A tibble: 1 x 4
#>   mape_lm mape_dtree mase_lm mase_dtree
#>     <dbl>      <dbl>   <dbl>      <dbl>
#> 1   0.320      0.415    2.52       2.92
```

Aufgabe: Verbessere obiges Modell durch Variation

1. Probiere ein Ensemble Modell aus (siehe caret Doku)
2. Füge zusätzliche Variablen hinzu