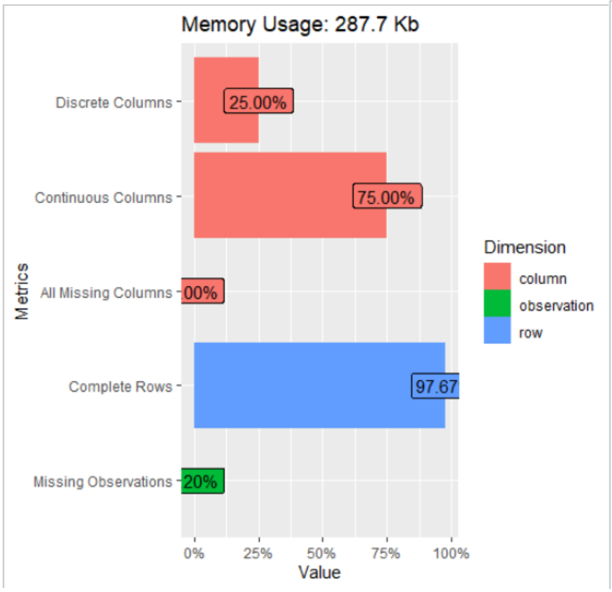
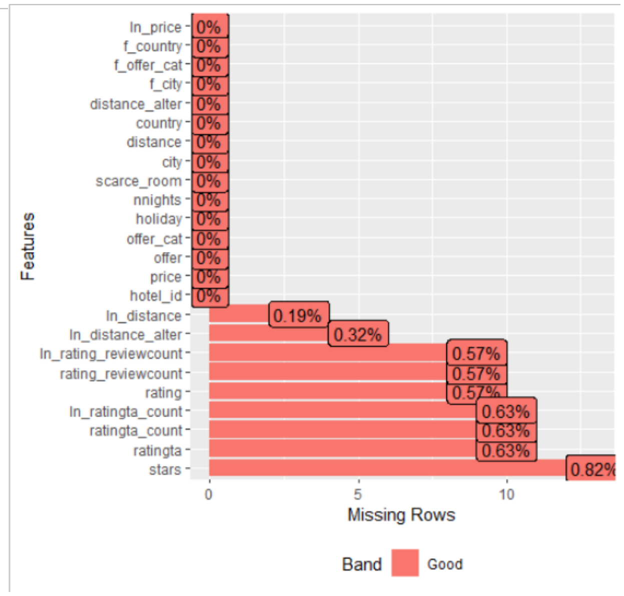
Homework 3 - Data Analysis 3
Option 3

Oscar Leal 1901361

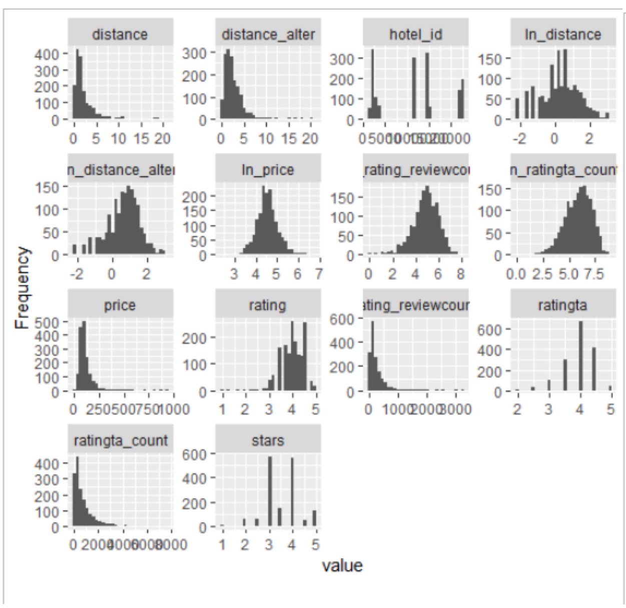
In this assignment the hotels-europe data from the Google Drive repo was used, filtered to hotel rooms. It was filtered to be specific to a date of weekday, in November 2017. Berlin, Munich, Vienna, Budapest, Prague, Warsaw were left out, and some not necessary columns were dropped. Any hotel room that its cost was more than 500, was left out since the mean was 112, many outliers were not going to do good in our training models. Data test/holdout distribution is 70/30.

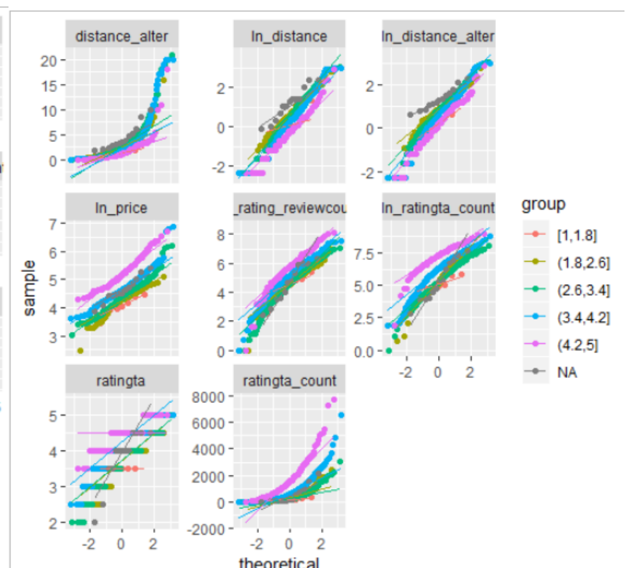
```
> skimr::skim(data)
── Data Summary ────────────────────────
                           Values
Name                       data
Number of rows             1576
Number of columns          24
_______________________
Column type frequency:
  character                3
  factor                   3
  numeric                  18
________________________
Group variables            None

── Variable type: character ────────────────────────────────────────────
  skim_variable n_missing complete_rate   min   max empty n_unique whitespace
1 offer_cat             0             1    10    13     0        5          0
2 city                  0             1     6     8     0        6          0
3 country               0             1     6    14     0        5          0

── Variable type: factor ───────────────────────────────────────────────
  skim_variable n_missing complete_rate ordered n_unique top_counts
1 f_city                0             1 FALSE          6 Ber: 394, Pra: 380, Mun: 297, Vie: 258
2 f_offer_cat           0             1 FALSE          5 15-: 637, 0% : 392, 1-1: 392, 50%: 100
3 f_country             0             1 FALSE          5 Ger: 691, Cze: 380, Aus: 258, Hun: 170

── Variable type: numeric ──────────────────────────────────────────────
   skim_variable         n_missing complete_rate      mean       sd      p0       p25       p50       p75      p100 hist
 1 hotel_id                      0         1      11312.    7551.    1745     2889.    11616.    14909.    22842
 2 price                         0         1        108.      59.6     12       71        93       127       492
 3 offer                         0         1          0.751    0.432    0        1         1         1         1
 4 holiday                       0         1          0        0        0        0         0         0         0
 5 nnights                       0         1          1        0        1        1         1         1         1
 6 scarce_room                   0         1          0.375    0.484    0        0         0         1         1
 7 distance                      0         1          2.29     2.88     0        0.7       1.4       2.73     21
 8 stars                        13         0.992      3.55     0.723    1        3         3.5       4         5
 9 rating                        9         0.994      4.01     0.461    1        3.7       4         4.4       5
10 rating_reviewcount            9         0.994    232.     283.       1       69       144       294.     3234
11 ratingta                     10         0.994      3.96     0.509    2        3.5       4         4.5       5
12 ratingta_count               10         0.994    694.     825.       1      155       408       921      7717
13 distance_alter                0         1          2.76     2.75     0        1.1       2.1       3.5      21
14 ln_price                      0         1          4.56     0.483    2.48     4.26      4.53      4.84      6.20
15 ln_rating_reviewcount         9         0.994      4.89     1.15     0        4.23      4.97      5.69      8.08
16 ln_ratingta_count            10         0.994      5.88     1.28     0        5.04      6.01      6.83      8.95
17 ln_distance                   3         0.998      0.283    1.08    -2.30    -0.357     0.336     1.03      3.04
18 ln_distance_alter             5         0.997      0.631    0.933   -2.30     0.0953    0.742     1.25      3.04
>
```



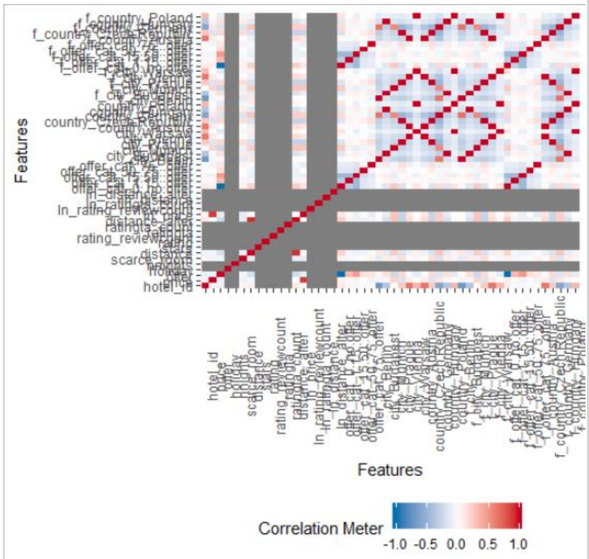


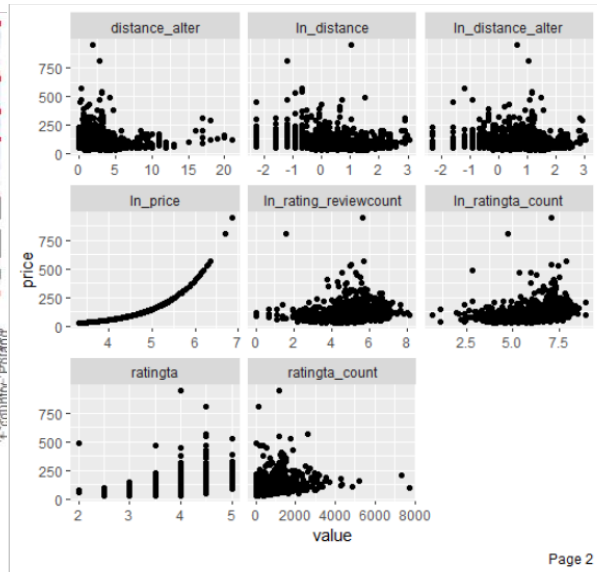
Distribution of variables, qqplots by group of stars



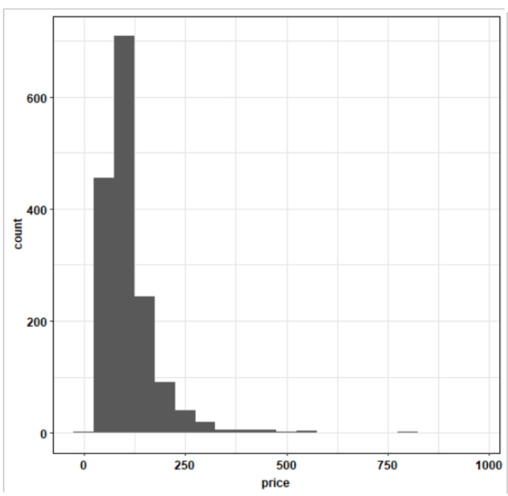


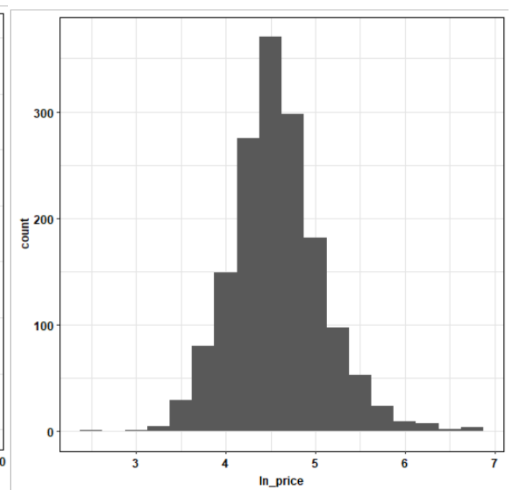
Correlation Plot and scatterplot by price





Price and Ln Price distributions





Variables

```
basic_vars <- c(
  "offer", "f_offer_cat", "f_city",
  "f_country","distance", "ln_distance", "holiday", "nnights",
  "ln_distance", "ln_distance_alter", "stars")

# reviews
rating <- c("rating", "ratingta" ,"ratingta_count","ln_rating_reviewcount",
            "rating_reviewcount","ln_ratingta_count")

predictormodforest <- basic_vars
predictors <- c(basic_vars, rating)
```

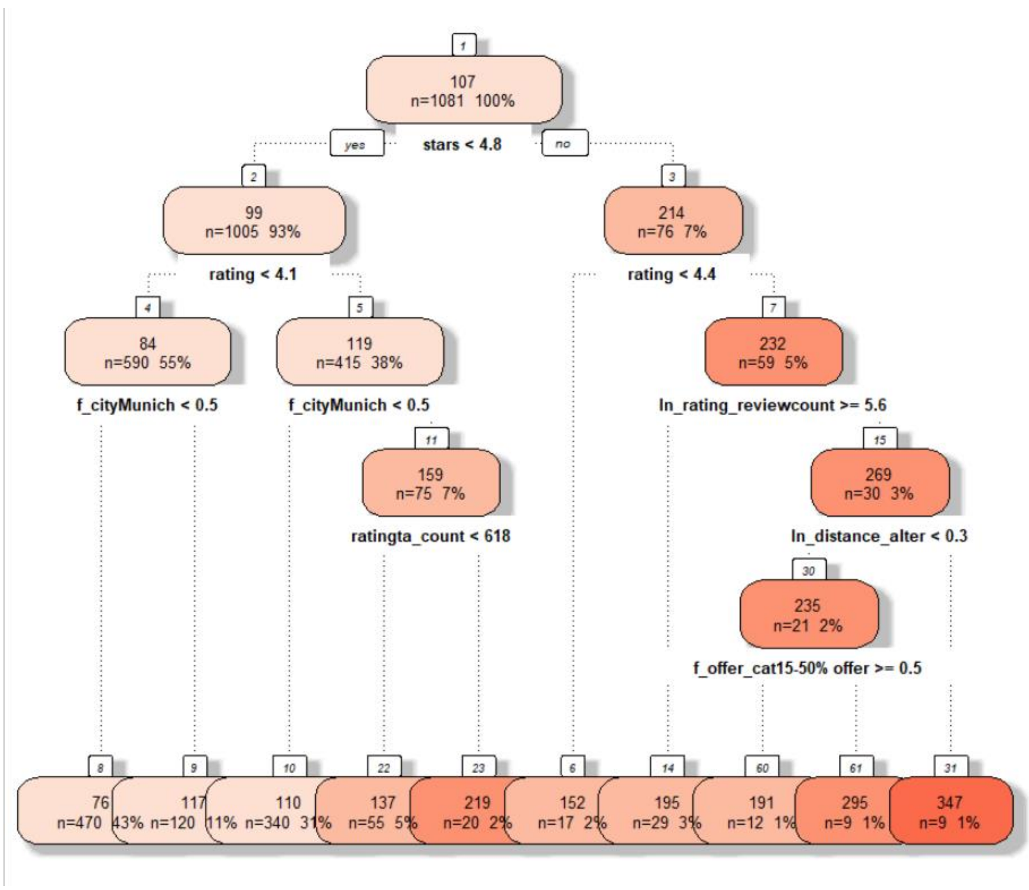
It was decided to create logs of distance, rating_reviewcount, and factors of city, country and offer_cat.

OLS settings

```
system.time({
  ols_model <- train(
    formula(paste0("price ~", paste0(predictors, collapse = " + "))),
    data = data_train,
    method = "lm",
    trControl = train_control,
    na.action = na.omit
  )
})
```

CART settings and plot

```
system.time({
  cart_model <- train(
    formula(paste0("price ~", paste0(predictors, collapse = " + "))),
    data = data_train,
    method = "rpart",
    tuneLength = 10,
    trControl = train_control,
    na.action = na.omit
  )
})
cart_model
```



Random Forests settings

Basic:

```r
system.time({
  rf_model_1 <- train(
    formula(paste0("price ~", paste0(predictormodforest, collapse = " + "))),
    data = data_train,
    method = "ranger",
    trControl = train_control,
    tuneGrid = tune_grid,
    importance = "impurity",
    na.action = na.omit
  )
})
rf_model_1
```

Advanced:

```r
system.time({
  rf_model_2 <- train(
    formula(paste0("price ~", paste0(predictors, collapse = " + "))),
    data = data_train,
    method = "ranger",
    trControl = train_control,
    tuneGrid = tune_grid,
    importance = "impurity",
    na.action = na.omit
  )
})
```

GBM settings

```r
gbm_grid <-  expand.grid(interaction.depth = c(1, 5, 10), # complexity of the tree
                         n.trees = 250, # number of iterations, i.e. trees
                         shrinkage = c(0.05, 0.1), # learning rate: how quickly the algo
                         n.minobsinnode = 20 # the minimum number of training set sample
)


set.seed(1234)
system.time({
  gbm_model <- train(formula(paste0("price ~", paste0(predictors, collapse = " + "))),
                     data = data_train,
                     method = "gbm",
                     trControl = train_control,
                     verbose = FALSE,
                     tuneGrid = gbm_grid,
                     na.action = na.pass)
})
gbm_model
```

Results of training:

```
Models: OLS, CART, Random forest (smaller model), Random forest, GBM (basic tuning)
Number of resamples: 5

MAE
                                  Min.  1st Qu.   Median     Mean  3rd Qu.     Max. NA's
OLS                           26.68276 26.68802 27.70635 27.95555 28.77453 29.92611    0
CART                          27.40136 28.76438 29.43759 29.54784 30.00277 32.13312    0
Random forest (smaller model) 24.86282 26.05095 28.56415 28.03096 30.33618 30.34073    0
Random forest                 21.84071 22.99050 24.45311 23.80608 24.52427 25.22179    0
GBM (basic tuning)            23.25023 23.51355 24.12374 24.62940 25.98976 26.26971    0

RMSE
                                  Min.  1st Qu.   Median     Mean  3rd Qu.     Max. NA's
OLS                           41.08775 42.42468 42.88760 43.50395 45.37675 45.74294    0
CART                          42.76359 44.55548 45.08180 45.50218 46.96586 48.14419    0
Random forest (smaller model) 34.84957 36.14272 46.25731 43.54161 48.45918 51.99927    0
Random forest                 38.88338 38.89113 39.13358 39.41327 40.04634 40.11192    0
GBM (basic tuning)            35.63074 36.00101 40.95952 40.14539 43.37941 44.75626    0

Rsquared
                                   Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
OLS                           0.4747178 0.4784610 0.4897678 0.4961562 0.4952434 0.5425908    0
CART                          0.4149075 0.4265405 0.4524818 0.4548739 0.4742775 0.5061624    0
Random forest (smaller model) 0.4266882 0.4415999 0.4684794 0.4911516 0.5468240 0.5721665    0
Random forest                 0.5331069 0.5438348 0.6085176 0.5872392 0.6204626 0.6302742    0
GBM (basic tuning)            0.4851004 0.4972980 0.5907511 0.5745487 0.6454595 0.6541347    0
```

RMSES on data_test:

```
> rmses
$OLS
[1] 66.10324

$CART
[1] 67.9512

$`Random forest (smaller model)`
[1] 63.612

$`Random forest`
[1] 67.06771

$`GBM (basic tuning)`
[1] 69.27824

>
```

Conclusions:

We can assume from the results of the training that the advanced tuned random forest was the winner as its mean and median are the lowest (39.41, 38.89). But surprisingly by actually predicting on the data_test (holdout) the one that predicted the best was the basic tuned random forest. (63.612).

This can prove the point that overfitting a model with a big set of variables will most of the time perform better in the training set, but not in the holdout set since it’s capturing the noise of the training set.