Finding Hotel Deals in Vienna using Machine Learning

Bruno Helmeczy

12/02/2021

```
# PACKAGES
library(caret)
library(dplyr)
library(readr)
library(data.table)
library(ggthemes)
library(ggplot2)
library(ggridges)
library(moments)
library(rattle)
library(knitr)
library(gridExtra)
#library(huxtable)
library(tidyverse)


# Load Data
FileSource <- "https://raw.githubusercontent.com/BrunoHelmeczy/Prediction_Projects-CEU_DA3/main/Find_Ho

df <- read_csv(paste0(FileSource,"hotelbookingdata.csv"))

# filter Time Period & Location -> Christmas Time in Vienna
df <- df[df$year == 2017 & df$month == 12 & df$holiday == 1,]
df <- df[df$s_city == "Vienna",]

# Check for singular value columns -> Remove
ColsUniques <- rbindlist(lapply(1:length(df), function(x) {
  tl <- list()
  tl[[’name’]] <- colnames(df)[x]
  tl[[’distinct’]] <- nrow(unique(df[,x]))
  return(tl)
}))

df[,which(ColsUniques$distinct == 1)] <- NULL
```

Executive Summary

This report looks to find the 5 best “Deals” in Vienna, during holidays in December 2017. I look to answer “What are the 5 best offers in town during the 2017 year-end holiday period ?” by applying a Linear Regression, a Classification & Regression Tree, & a Random Forest model on data obtained from Trip Advisor. I define ‘best deals’ as those with largest negative residuals vs their expected price according to my selected model. I found the Random Forest model to vastly outperform both the OLS & CART model specification, reaching a 68.3% Adjusted R-squared value, 16.2 euros Mean Absolute Error & 24.5 euros RMSE. Finally, the model found 5 deals that represent bargains between 44-55 euros, in addition to being diverse in terms of pricing, stay length, star rating, distance from the centre & accommodation type.

Introduction

Vienna is a popular destination during the (pre)-christmas period, famous for its christmas market, & beautiful scenery, glamorous reminders of the glorious age of the Austria-Hungarian Monarchy. As such a conveniently distanced, & suitable destination for couples' & families' from neighbouring countries, to relax. I look to answer "Which 5 hotels offer the best 'Deal' during 2017 year-end holiday period ?" by applying a Linear Regression, Classification & Regression Tree, & a Random Forest model, on data obtained from Trip Advisor. I define 'Best Deal' as the hotel offering the most-value for money. In this context, after modelling hotels' prices based on variables related to hotel quality, distance from the city centre & guest reviews, I define the 'best deals' as those observations with largest negative residuals vs their predicted level price according to my selected model.

Data Cleaning, Association Patterns, & Variable Transformations

Cleaning: My raw data comprises the hotels europe dataset available by clicking here. This dataset includes hotel prices & various other features from 46 european cities on 10 different dates, & is scraped from a price comparison website. I filtered the dataset for a 2017, December, holiay noted period, for deals under 400 euros per night, netting 676 observations. Note that I included both 1-night & 4-night deals, as they reporesent different offerings, possibly with different value for money, even if offered by the same hotel. Furthermore, it is a realistic scenario for travellers to decide on the holidays' length, based on the deals that they would find most to their liking. To avoid comparing apples & oranges however, I calculated the price per night for every deal, which represents my target variable. I cleaned the data 1st by Converting distance strings, & review ratings to numeric vectors; 2nd by Imputing missing Review Scores for a hotel with the datasets' median, & adding flag variable columns to note hotels without reviews; & 3rd by converting accommodation type, offer type & neighbourhood variables to a series of dummy variables, resulting in 44 predictor variables used.

Distributions & Association Patterns: Please see the most important quantitative variables' summary statistics below, as well as all variables' histograms in the appendices. One may observe skewness with long right tails for prices per night, both distance variables, & the Nr. of reviews posted on Trip Advisor, while for the alternative rating measure skewness with long left tail is visible. Based on the summary statistics & histograms I changed the target variable to its' log-form. Also due to observed summary statistics, histograms & association patterns, I log-transformed both distance measures & review counts, while taking the exponent of guest reviews. Finally, I pivoted neighbourhoods, offer categories, & accommodation type variables to dummy tables.

```r
# Plot Variable Distributions & Frequency tables ----
Hists <- lapply(df %>% select(-matches("id")) %>% colnames(),function(x) {
  if(is.numeric(df[,x][[1]]) ) {
    df %>% ggplot(aes_string(x = x)) +
      geom_histogram( color = "red", fill = "blue") + theme_tufte() +
      labs(title = paste0("Vienna Hotels ",x," Distribution"))
  } else if (is.character(df[,x][[1]])) {
    df %>% ggplot(aes_string(x = x ) ) +
      geom_bar(color = "red", fill = "blue") + coord_flip() + theme_tufte() +
      labs(title = paste0("Vienna Hotels ",x," Distribution"))
  }
})

# Summary table -----
summstats <- c("min","median","mean","max","sd","skewness")
NrCols <- c("price","pricepernight","Nrnights","starrating","center1distance","center2distance",
             "rating2_ta","rating2_ta_reviewcount","guestreviewsrating")
```

```
summtable <- list()
table <- data.frame()
for (i in summstats) {
  summtable[[i]] <- cbind(mapply( eval(parse(text = i)), df[,NrCols]))
  if (length(table) == 0) {
    table <- cbind(summtable[[i]])
  } else {table <- cbind(table, summtable[[i]])}
}
colnames(table) <- summstats
table <- as.data.frame(round(table,1))
Vars <- rownames(table)
rownames(table) <- NULL
table <- as.data.frame(cbind(Vars,table))

table %>% knitr::kable()
```

Vars	min	median	mean	max	sd	skewness
price	30.0	262.0	359.1	1546	300.3	1.4
pricepernight	30.0	128.0	147.4	386	67.8	1.2
Nrnights	1.0	1.0	2.4	4	1.5	0.2
starrating	1.0	3.5	3.4	5	0.7	-0.3
center1distance	0.0	1.5	1.7	13	1.6	3.1
center2distance	0.5	3.5	3.7	13	1.6	1.5
rating2_ta	2.5	4.0	4.0	5	0.4	-0.7
rating2_ta_reviewcount	0.0	233.0	471.6	3262	613.5	1.9
guestreviewsrating	1.0	4.0	4.0	5	0.5	-1.3

```
# Check 4 Transformations & Dummy Feature Engineering ->
  # Log:
    # Pricepernight (y)
    # rating2_ta_reviewcount
    # center2distance
    # center1distance
    # rating_reviewcount
  # Exp:
    # guestreviewsrating
  # Dummy tables:
    # Accommodationtype
    # Neighbourhood

df <- df %>% mutate(
  pricepernight_ln          = log(pricepernight),
  rating2_ta_reviewcount_ln = log(rating2_ta_reviewcount+1),
  center2distance_ln        = log(center2distance+1),
  center1distance_ln        = log(center1distance+1),
  rating_reviewcount_ln     = log(rating_reviewcount+1),
  guestreviewsrating_exp    = exp(guestreviewsrating))


Boxes_Scatters <- lapply(df %>% select(-matches("id|pricepernight|flag")) %>%
                           select(-price) %>% colnames(),function(x) {
```

```
  if (is.character(df[,x][[1]]) | x %in% c("starrating", "rating2_ta","Nrnights")) {
    plot <- df %>% ggplot()  +
      geom_density_ridges(aes_string(y = x, x = "pricepernight_ln", group = x),color = "red", fill = "bl
      theme_tufte() +
      labs(title = paste0("Vienna Hotels log-Price Distr. by ",x))

  } else if(is.numeric(df[,x][[1]]) ) {
    plot <- df %>% ggplot(aes_string(x = x,y = "pricepernight_ln")) +  theme_tufte() +
      geom_smooth(method="loess", color="black", size = 1) +
      labs(title = paste0("Vienna Hotels log-Price Distr. by ",x))
    if (length(unique(df[,x][[1]])) == 2) {
      plot <- plot + geom_point( color = "red", shape = 3)

    } else {
      plot <- plot + geom_point( color = "blue", )
    }
  }
})


# Dummy Tables 4: offer_cat, accommodationtype, neighbourhood ----
  # Get vector of raw factor levels & Dataframe of dummies if in level

# Offer Category
Levels <- levels(factor(unlist(df$offer_cat)))
Dummies1 <- as.data.frame(do.call(rbind, lapply(lapply(df$offer_cat, factor, Levels), table)))
colnames(Dummies1) <- paste0("p",Dummies1 %>% colnames())

# Accommodation Type
Levels <- levels(factor(unlist(df$accommodationtype)))
Dummies2 <- as.data.frame(do.call(rbind, lapply(lapply(df$accommodationtype, factor, Levels), table)))

# Neighbourhood
Levels <- levels(factor(unlist(df$neighbourhood)))
Dummies3 <- as.data.frame(do.call(rbind, lapply(lapply(df$neighbourhood, factor, Levels), table)))

# City Actual
Levels <- levels(factor(unlist(df$city_actual)))
Dummies4 <- as.data.frame(do.call(rbind, lapply(lapply(df$city_actual, factor, Levels), table)))
colnames(Dummies4) <- paste0("city_",colnames(Dummies4))
  #summary(Dummies4) -> 97% writes Vienna -> drop, no use

df <- cbind(df,Dummies1,Dummies2,Dummies3) %>%
  select(-c(city_actual,offer_cat,‘p0% no offer‘))

# Var name cleaning ------
colnames(df) <- gsub("+","",
                     gsub("%","",
                          gsub("17._","",
                               gsub("-","_",
                                    gsub(" ","_",df %>% colnames())))))

colnames(df)[27] <- "p75_offer"
```

```
df$p75_offer <- NULL
  # 75%+ offers = 0.2% of observations -> Model returns NA coeffs due to singularity
  # Same for Vacation_home_Condo -> 0.2%
df$Vacation_home_Condo <- NULL

df <- df %>% rename(
  c1dist_ln = center1distance_ln,
  c2dist_ln = center2distance_ln
) %>% select(-price)
```

Modelling

Model Comparisons: With cleaning & transformations out of the way, 3 models were specified, each using the same group of predictor variables & predicting the natural logarithm of prices per night. Given the goal of this analysis is to find best deals in sample, no holdout data was sampled, however I performed 10-fold cross-validation. Please see the models' resulting summary statistics below, having already transformed log-price predictions to level-forms. One can observe the Regression Tree to beat the Multiple-Linear Regression model in terms of Mean absolute error (24.5 vs 30.9) & Root Mean Square error (35.3 vs 43.7), despite boasting slightly lower Adjusted R-squared (52.7% vs 56.6%). The Random Forest model however, using 7 variables at a time & a minimum node size of 5, vastly outperforms both, boasting ca. 16% higher adjusted R-squared (68.8% vs 52.7%), over 8 euros lower MAE (16.3 vs 24.5) & over 10 euros less RMSE (24.8 vs 35.3). As such, the Random Forest is the clear winner & therefore is my chosen final model.

```
# Model Specifications -------------
predictors <- df %>%
  select(-matches("id|pricepernight|accommodationtype|neighbourhood|rating2_ta_reviewcount$|center2dista

# Sample vs Holdout -> No holdout -> In sample prediction
  # 10-fold CV, insted of 5 though -> not soooo many observations
#nrow(df)

# train control is 5 fold cross validation
train_control <- trainControl(method = "cv",
                              number = 10,
                              verboseIter = FALSE)

#formula(paste0("pricepernight_ln ~ ", paste0(predictors, collapse = " + ")))

#### OLS ####
set.seed(1234)
  ols_model <- train(
    formula(paste0("pricepernight_ln ~ ", paste0(predictors, collapse = " + "))),
    data = df,
    method = "lm",
    trControl = train_control
  )

ols_model_coeffs <-  ols_model$finalModel$coefficients
ols_model_coeffs_df <- data.frame(
  "variable" = names(ols_model_coeffs),
  "ols_coefficient" = ols_model_coeffs
) %>%
  mutate(variable = gsub("‘","",variable))
```

```
#### CART ####
set.seed(1234)
  cart_model <- train(
    formula(paste0("pricepernight_ln ~ ", paste0(predictors, collapse = " + "))),
    data = df,
    method = "rpart",
    trControl = train_control,
    tuneGrid= expand.grid(cp = 0.001))

#### Random Forest ####

# set tuning
tune_grid <- expand.grid(
  .mtry = c( 5, 6, 7 ),
  .splitrule = "variance",
  .min.node.size = c(5, 10, 15)
)

set.seed(1234)
  rf_model_1 <- train(
    formula(paste0("pricepernight_ln ~ ", paste0(predictors, collapse = " + "))),
    data = df,
    method = "ranger",
    trControl = train_control,
    tuneGrid = tune_grid,
    importance = "impurity"
  )

#------------------ Model Comparison ####

final_models <-
  list("OLS" = ols_model,
       "CART" = cart_model,
       "Random_forest" = rf_model_1)


results <- resamples(final_models) %>% summary()

models <- c("ols_model","cart_model","rf_model_1")
Predictions <-data.frame(sapply(models,function(x) {
  tl <- list()
  model <- eval(parse(text = x))
  tl[[x]] <- predict(model,newdata = df)
  res <- tl[[x]] - df$pricepernight_ln
  StDev <- sd(res)
  tl[[x]] <- exp(tl[[x]]) * exp((StDev^2)/2)

  return(tl)
}))

Rsq <- NULL
for (i in 1:3) {
  Rsq[models[i]] <- round(results[[3]][[’Rsquared’]][i,4],3)
}


SumStatTable <- as.data.frame(cbind(Rsq,rbindlist(lapply(Predictions, function(x) {
  tl <- list()
  tl[[’MAE’]] <- round(MAE(x,df$pricepernight),3)
  tl[[’RMSE’]] <- round(RMSE(x, df$pricepernight),3)
  tl[[’RMSE_norm’]] <- round(tl[[’RMSE’]]/mean(df$pricepernight),3)
  return(tl)
}))))
rownames(SumStatTable) <- names(final_models)

SumStatTable %>% kable()
```

	Rsq	MAE	RMSE	RMSE_norm
OLS	0.566	30.855	43.738	0.297
CART	0.527	24.472	35.336	0.240
Random_forest	0.688	16.295	24.777	0.168

```
  # Model Choice : Random Forest -> mtry = 7 , Min.NodeSize = 5
```

Final Model: Please see the final models’ summary statistics below, re-estimated on the complete dataset. Fitted to the complete dataset, though Adjusted R-squared slightly decreased to 68.3%, both MAE & RMSE improved marginally. Importantly, the Root Mean Square Error only represents 16.6% of the average daily room price. With that, the model is ready to find the 5 best deals in town for what presumably is the christmas period.

```
#### Final Model Re-Estimation ####
train_control <- trainControl(method = "none",verboseIter = FALSE)

# set tuning for final model
tune_grid <- expand.grid(
  .mtry = c(7),
  .splitrule = "variance",
  .min.node.size = c(5)
)

set.seed(1234)
  rf_model_final <- train(
    formula(paste0("pricepernight_ln", paste0(" ~ ",paste0(predictors, collapse = " + ")))),
    data = df,
    method = "ranger",
    trControl = train_control,
    tuneGrid = tune_grid,
    importance = "impurity"
  )

# rf_model_final$finalModel
# summary(rf_model_final)

#### Summary of RF_2_Final Model ####
```

```
df <- df %>%
  mutate(predicted_price_ln = predict(rf_model_final, newdata = df))

df <- df %>% mutate(res = predicted_price_ln - pricepernight_ln)
StDev <- sd(df$res)
df$predicted_price <- exp(df$predicted_price_ln) * exp((StDev^2)/2)
Rsq <- round(rf_model_final$finalModel$r.squared,3)

SumStatsFinalModelTable <-
  cbind(Rsq,rbindlist(lapply(df[c("predicted_price")], function(x) {
    tl <- list()
    tl[[’MAE’]] <- round(MAE(x,df$pricepernight),3)
    tl[[’RMSE’]] <- round(RMSE(x, df$pricepernight),3)
    tl[[’RMSE_norm’]] <- round(tl[[’RMSE’]]/mean(df$pricepernight),3)
    return(tl)
  })))

modelname <- "RF Final Model"
SumStatsFinalModelTable <- cbind(modelname,SumStatsFinalModelTable)

SumStatsFinalModelTable %>% kable()
```

modelname	Rsq	MAE	RMSE	RMSE_norm
RF Final Model	0.683	16.179	24.45	0.166

The 5 Best Deals: Please see the 5 best deals according to the final model, as well as the 5 hotels provided as best deals' according to the reference model provided for this course. Interestingly, the 5 best deals seem quite diverse in nature & price, incorporating 2 hotels in Innere Stadt of different star rating for 1 night, 1 apartment just over a mile from the centre that is quite budget-friendly, & 2 apartments further from the centre, however 1 of them for 4 nights. Thus, if one sees this analysis' purpose to be finding a diverse variety of alternatives that represent good value-for-money, then the final Random Forest model achieves that purpose, each deal offering a daily bargain between 44-55 euros according to our model. Meanwhile, the deals offered by the 5 hotels provided as best deals' according to a reference model only represented bargains of 11-18 euros, 1 of them even being over-priced by 28 euros according to the final model.

```
#### Residual Analysis 2 Find BEST DEALS ----
  # i.e. Largest Negative residuals
    # largest as in ? Percent / Money saved / total money saved
      # Money saved / day
df$res <- round(df$pricepernight - df$predicted_price,1)

# Characterize best deal hotels
  # Hotel id, Star rating, Pricepernight, res, Nrnights,
    # rating2_ta, exp(rating2_ta_reviewcount_ln)-1, guestreviewsrating_exp,
    # +1s: neighbourhood, distance to centre

# Hotels ranked by Best Deals according to our model
Deals <- df %>% select(hotel_id,starrating,pricepernight, res,Nrnights,
                       accommodationtype,neighbourhood, center1distance,
                       rating2_ta,rating2_ta_reviewcount) %>%
  arrange(res)
```

```
Deals <- rbind(Deals[c(1:5),],
               Deals[Deals$hotel_id %in% c(21912, 21975, 22344, 22080, 22184),])

names <- c("Hotel_id","Stars","Avg.Price",
           "Resid.","Nights","Type",
           "Where?","Miles fr Center","TA_Rating","Nr.Ratings")
colnames(Deals) <- names
Deals %>% kable()
```

	Hotel_id	Stars	Avg.Price	Resid.	Nights	Type	Where?	Miles fr Center	TA_Rating	Nr.Ratings
1	21905	3.5	70	-54.4	1	Apartment	Alsergrund	1.2	4.0	0
2	21980	4.0	146	-47.3	1	Hotel	Innere Stadt	0.1	4.0	714
3	21982	3.0	123	-46.4	1	Hotel	Innere Stadt	0.5	4.5	985
4	22360	3.0	104	-45.2	4	Apartment	Vienna	2.8	4.0	0
5	21939	3.0	95	-44.6	1	Apartment	Favoriten	2.5	4.0	0
65	22184	3.0	85	-18.6	1	Hotel	Leopoldstadt	0.7	4.0	827
79	21975	4.0	197	-17.2	1	Hotel	Innere Stadt	0.1	4.5	211
84	22344	3.0	65	-16.7	4	Hotel	Vienna	3.9	4.0	12
106	22344	3.0	57	-14.8	1	Hotel	Vienna	3.9	4.0	12
150	21912	4.0	95	-10.7	1	Hotel	Alsergrund	1.1	4.0	359
151	22080	3.0	65	-10.7	1	Hotel	Josefstadt	1.1	3.0	85
588	21975	4.0	264	28.0	4	Hotel	Innere Stadt	0.1	4.5	211

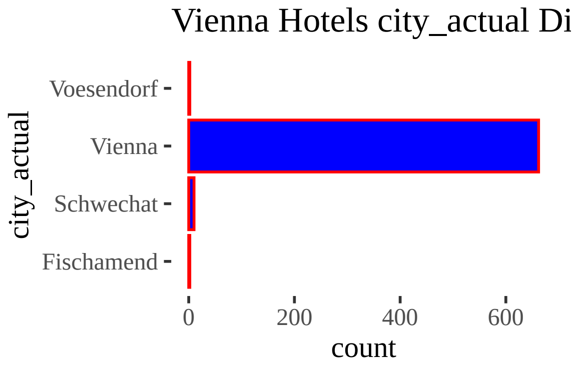
Conclusion / Summary

This report investigated the best hotel deals available in Vienna, during the 2017 December holiday period, using data originally scraped from a price comparison website, & also considering apartments, not just hotels, while sticking to a daily budget of max 400 euros, yet being flexible about staying for 1, or 4 nights. After comparing performance of an Ordinary Least Squares, a Classification & Regression Tree, & a Random Forest model, I found Random Forest to vastly outperform OLS & CART. Meanwhile, re-estimated on the whole dataset, the final model performed consistently, reaching adjusted R-Squared of 68.3%, mean absolute error of 16.2 euros, & root mean squared error of 24.45 euros. Using this model, I found the 5 'best deals' for the period by looking at largest negative residuals, each deviating between 45-55 euros from their respective predicted prices. Interestingly, the deals found were quite diverse in both nature & pricing, while the 5 best deals according to the reference model provided for this course represented bargains of about 11-18 euros.

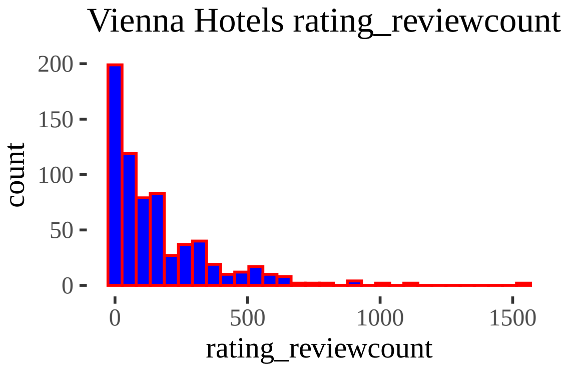
Appendices

1) Variable Histograms

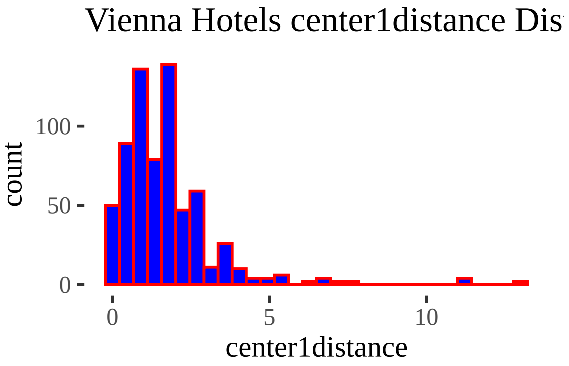
```
Hists
```


Vienna Hotels city_actual Di
city_actual
Voesendorf
Vienna
Schwechat
Fischamend
0
200
400
600
count

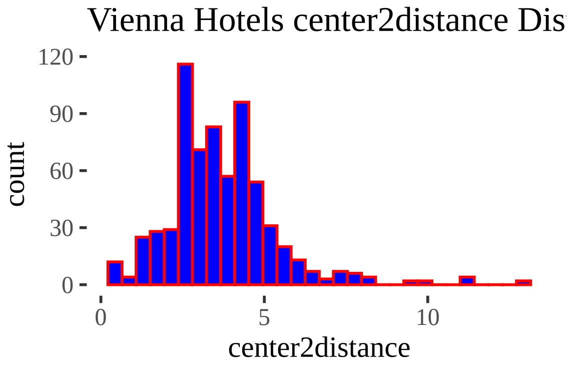
[[1]]


Vienna Hotels rating_reviewcount
count
200
150
100
50
0
0
500
1000
1500
rating_reviewcount

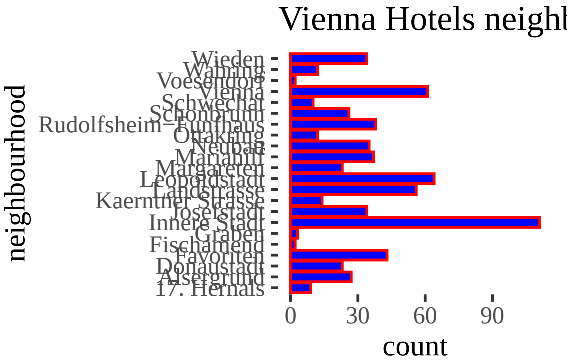
[[2]]


Vienna Hotels center1distance Dis
count
100
50
0
0
5
10
center1distance

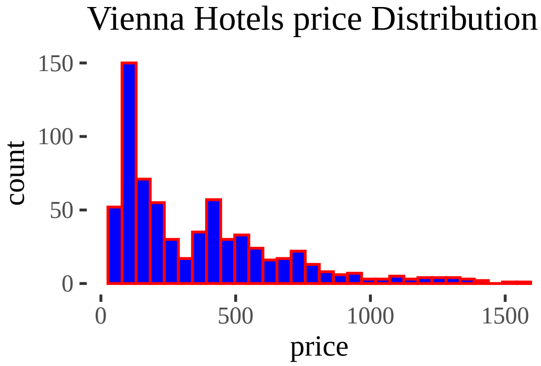
[[3]]


Vienna Hotels center2distance Dis
count
120
90
60
30
0
0
5
10
center2distance

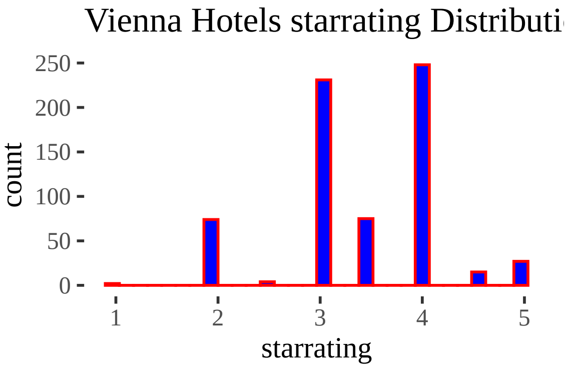
[[4]]


Vienna Hotels neighl
neighbourhood
Wieden
Wahring
Voesendorf
Vienna
Schwechat
Schonbrunn
Rudolfsheim−Funfhaus
Ottakring
Neubau
Mariahilf
Margareten
Leopoldstadt
Landstrasse
Kaerntner Strasse
Josefstadt
Innere Stadt
Graben
Fischamend
Favoriten
Donaustadt
Alsergrund
17. Hernals
0
30
60
90
count

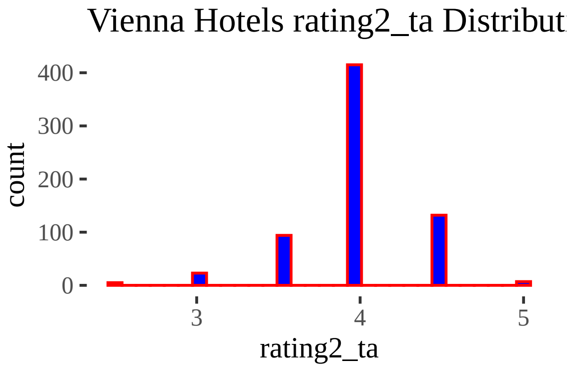
[[5]]


Vienna Hotels price Distribution
count
150
100
50
0
0
500
1000
1500
price

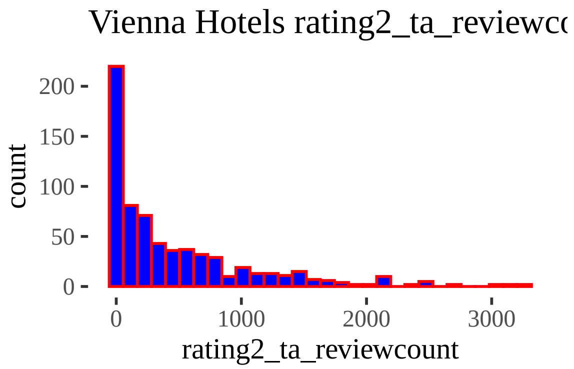
[[6]]


Vienna Hotels starrating Distributi
count
250
200
150
100
50
0
1
2
3
4
5
starrating

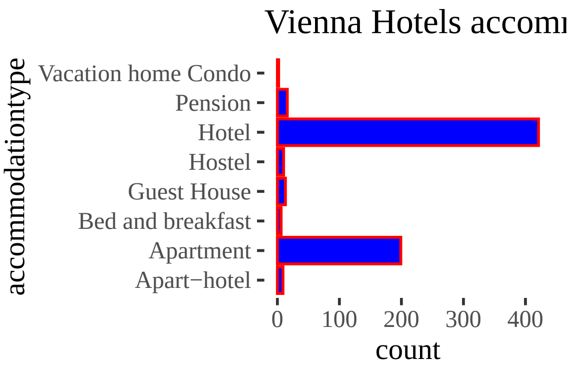
[[7]]


Vienna Hotels rating2_ta Distribut
count
400
300
200
100
0
3
4
5
rating2_ta

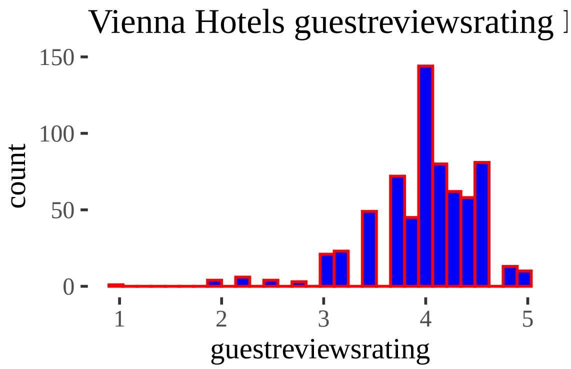
[[8]]



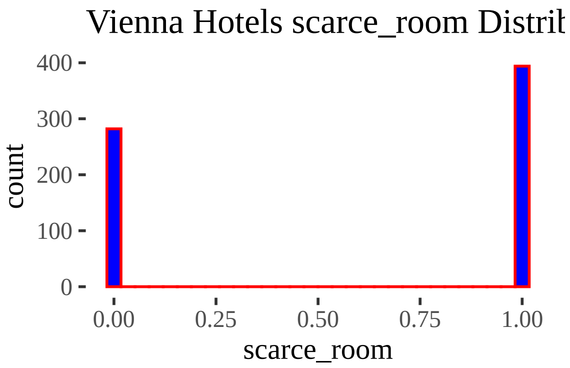
[[9]]



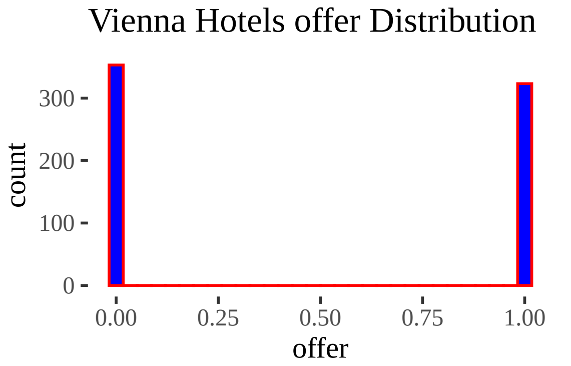
[[10]]



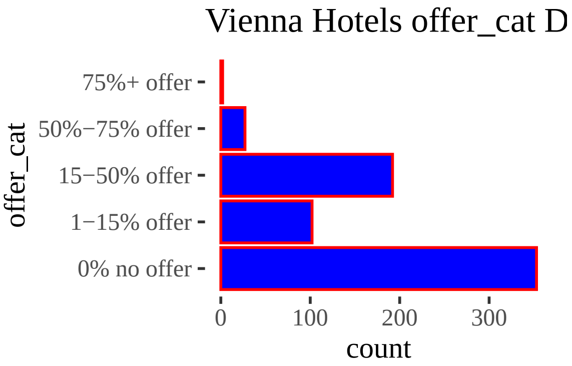
[[11]]



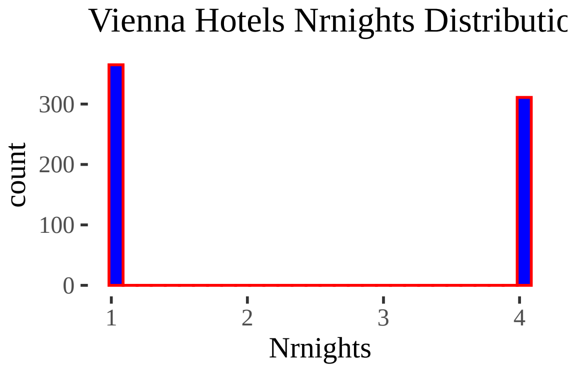
[[12]]



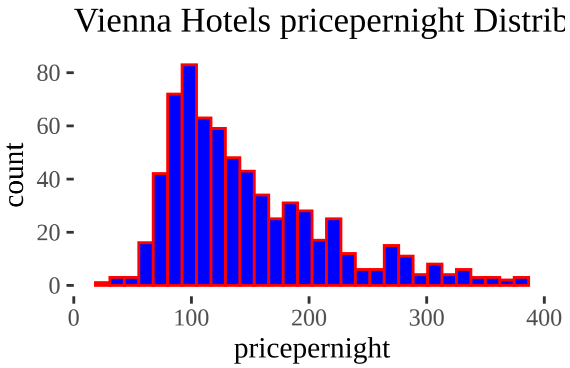
[[13]]



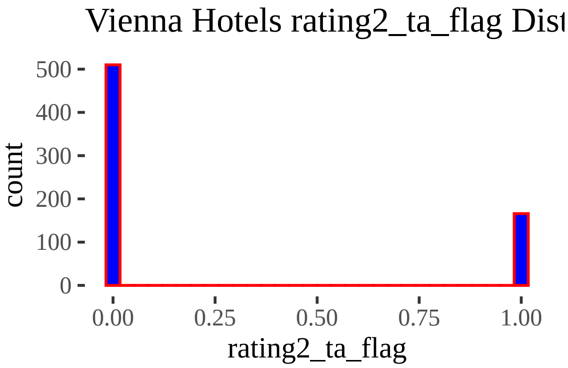
[[14]]



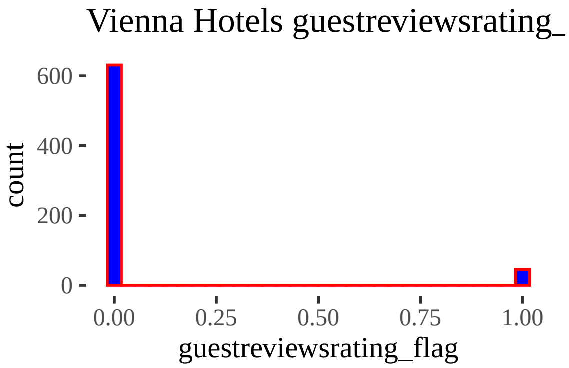
[[15]]



[[16]]



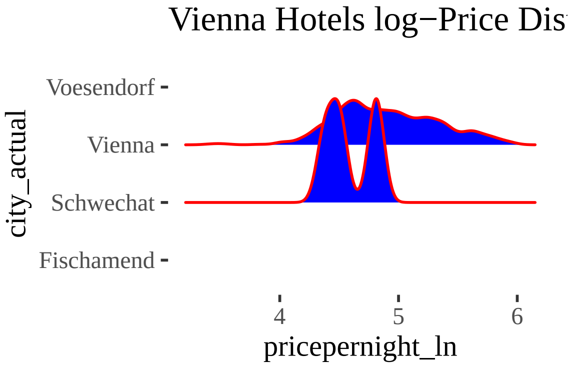
[[17]]



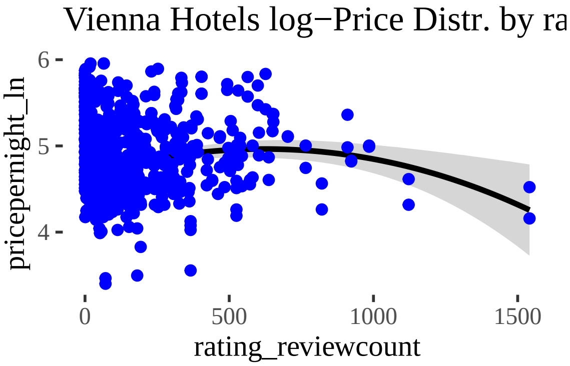
[[18]]

2) Scatterplot Associations & Ridge Distributions

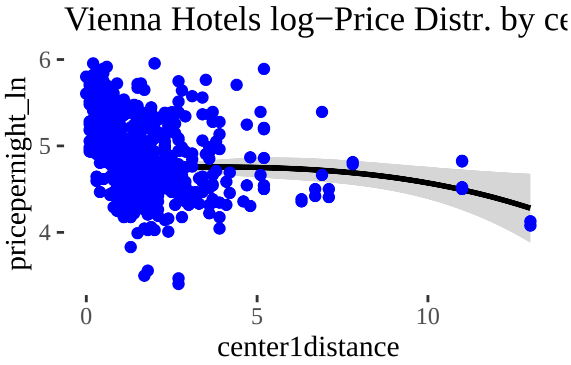
```
Boxes_Scatters
```



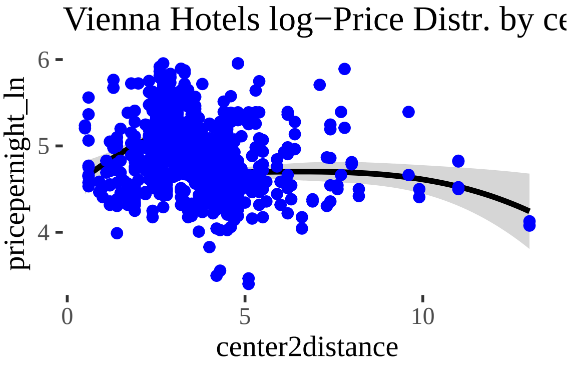
[[1]]



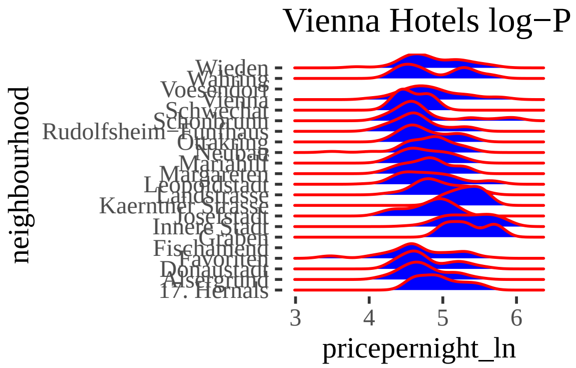
[[2]]



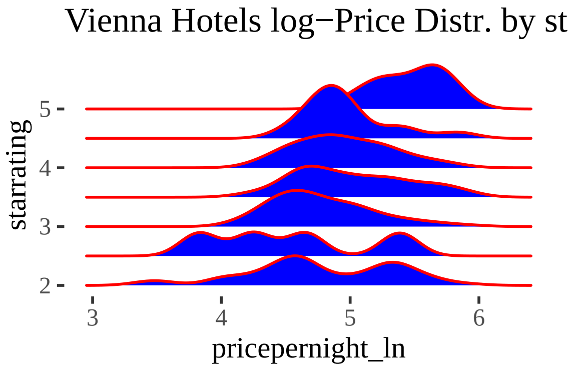
[[3]]



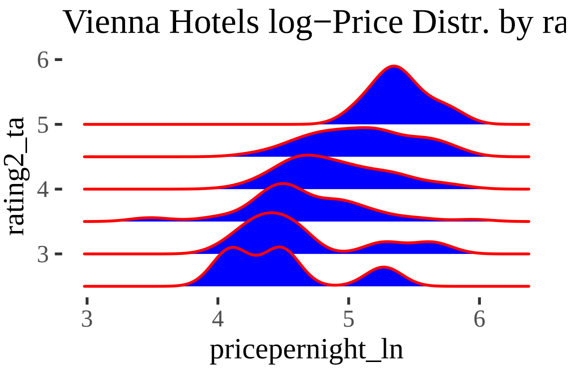
[[4]]



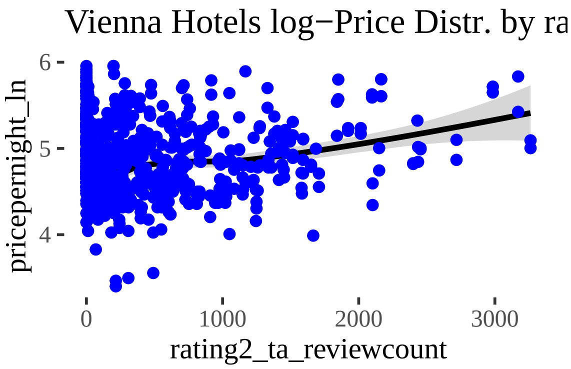
[[5]]



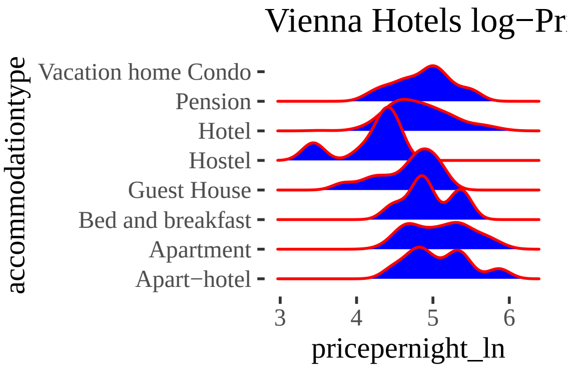
[[6]]



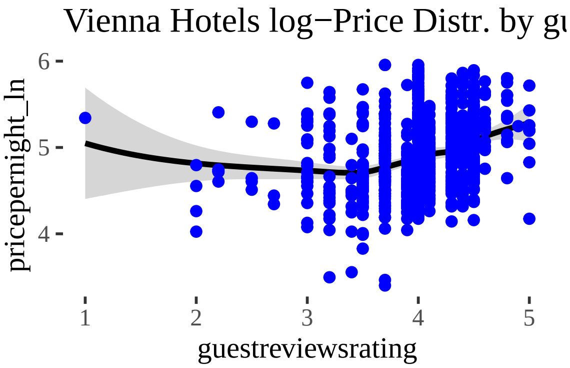
[[7]]



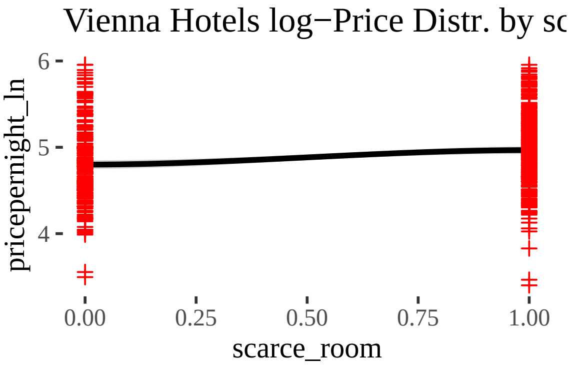
[[8]]



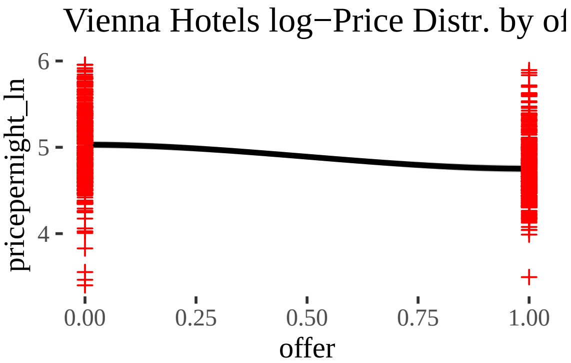
[[9]]



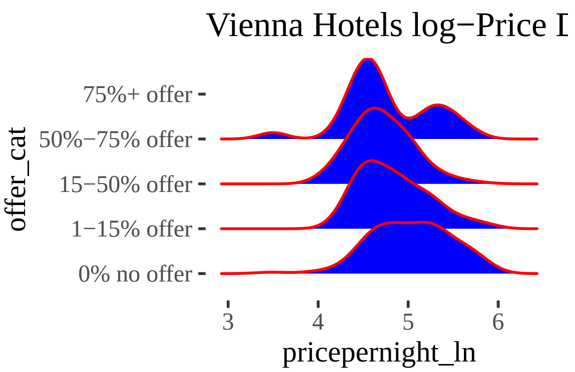
[[10]]



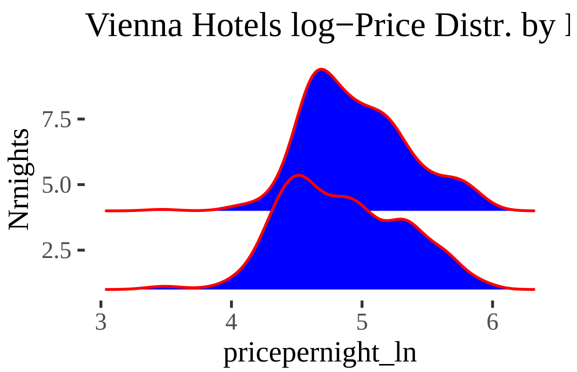
[[11]]



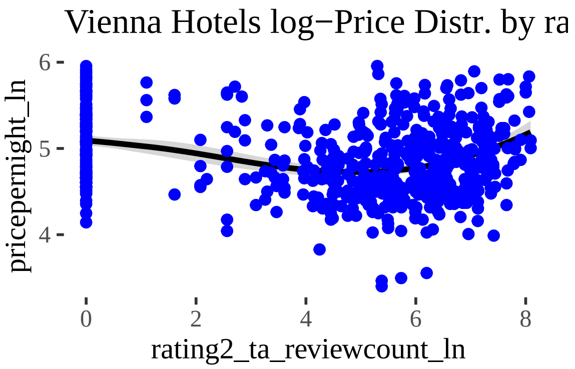
[[12]]



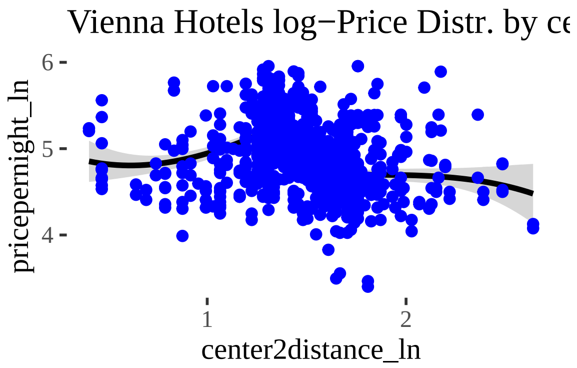
[[13]]



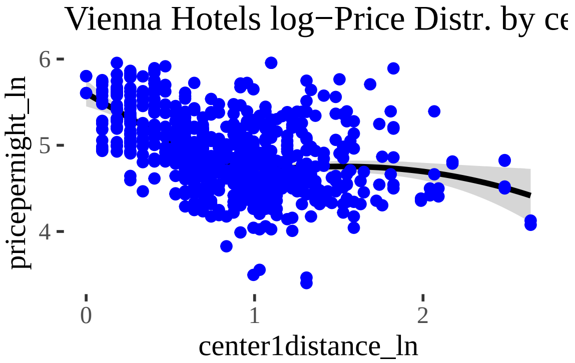
[[14]]



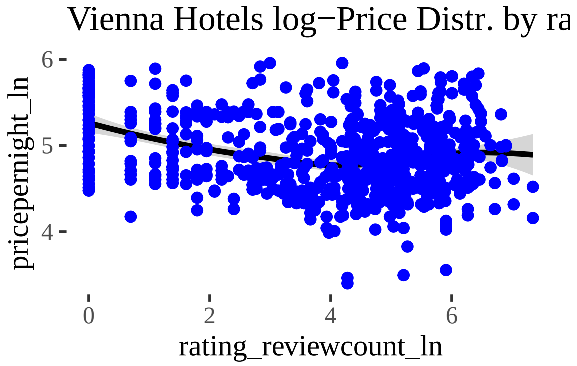
[[15]]



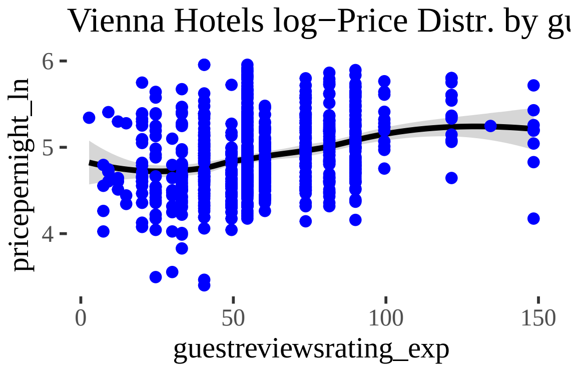
[[16]]



[[17]]



[[18]]



[[19]]