Assignment 2 - Predicting Rio Airbnb Prices Technical

A. BOGNAR

Introduction

This is a supplementary paper to *Predicting Rio Airbnb Prices Summary* explaining the technical decisions made in the analysis.

EDA/Feature engineering

For predictive variables the following decisions were made:

- *Units* are similar to *Apartments* in prices and most major characteristics, within the descriptions and advertisements, they are also referred to as apartments. Thus these two categories were used for the analysis.
- From *host verifications* most channels (email, facebook) are listed for nearly all apartments. Government certificates are less common and may be considered to be important, therefore it was included as binary variable.
- Nearly half of the *neighborhoods* consist of less than 100 units. With cross validation and holdout samples, these could not be reliable predicted. I considered adding a dummy variable for these districts, however, I wanted to see how these smaller districts as together compare to other districts, thus they were renamed to a separated category.
- For *bathrooms* values are provided in text format. A few were listed with 0 bathrooms, but upon checking the adds these turned out to be mistakes and were changed to 1 bathroom. Otherwise, numbers were extracted from the texts, some were listed with 1.5, 2.5 bathrooms. These were also kept as the variation may still be relevant.
- Where number of bedrooms was missing (about 7% of data), the median, 1 bedroom was imputed. Flag was generated.
- For *number of nights* many were provided with over a week. Some extreme cases are definitely erroneous. However, even for anything above a week, I considered it may not make a difference for rentees. Further categories 1 week - few weeks may also make sense.
- *Host* related variables were added as rentees might prefer to choose trusted hosts.
- *Number of reviews* seemed to show quadratic association with price, so polynomials were generated.
- For *amenities* considered the below code chunk:

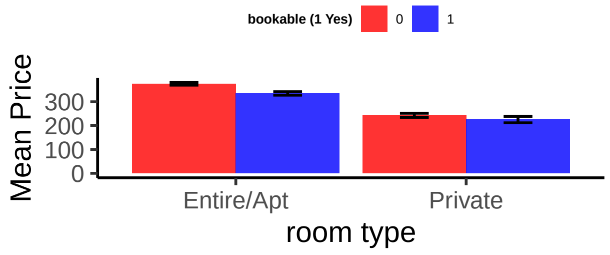
```r
# Dealing with amenities

# First, get all unique amenities

unique_amenities <-  unique(trimws(gsub("[[]" , "" ,gsub("[]]" , ""
                                                        ,gsub("\"" , "" ,unlist(strsplit(as.character(data$am
                                                                                             ",")))))))

# Use this list to find those relevant for prediction

key_words <- c("hdtv", "oven" , "wifi" ,  "refrigerator" ,
                "garage" ,  "pool" ,  "gym" ,
               "grill" , "coffee"  , "dryer" ,
               "washer" ,  "parking" , "sound system" , "air conditioning" ,
               "elevator")

# Double loop to select the list items from all unique possibilities
# This prevents observations not being selected for the item
# if word is part of string eg. "Sony sound system" instead of "sound system".
# Also minding capitalization

for (x in key_words) {
  unique_amenities_mod <- c()

  for (i in seq_along(unique_amenities)) {
    new_item <- ifelse(grepl( x , tolower(unique_amenities[i]), fixed = TRUE) ,
                       tolower( x ) , unique_amenities[i] )
    unique_amenities_mod <- c(unique_amenities_mod , new_item)
  }

  unique_amenities <-  unique(unique_amenities_mod)

}

# Add binary columns for amenities

for(p in key_words) data <- data %>%
      mutate(!! p := +(ifelse((grepl( p, tolower(data$amenities), fixed = TRUE)),1,0)))

# Correct column names

data <- rename(data, c(air_conditioning = `air conditioning`,
                       sound_system = `sound system` ))

# Some key words have synonyms

data <- data %>%
  mutate(  oven = ifelse(data$oven == 1 , 1 ,
                         ifelse((grepl( "stove", tolower(data$amenities), fixed = TRUE)),1,0)) )

data <- data %>%
  mutate(  air_conditioning = ifelse(data$air_conditioning == 1 , 1 ,
                         ifelse((grepl( "AC unit", tolower(data$amenities), fixed = TRUE)),1,0)) )

# Separate television from hdtv

data <- data %>%
  mutate(  television =
             ifelse((grepl( "tv", gsub("hdtv" , "television" ,
                                 tolower(data$amenities)), fixed = TRUE)),1,0) )

# add dummy d_ to column names

dummies <- names(data)[seq(78,93)]

data <- data %>%
  mutate_at(vars(dummies), funs("d"= (.)))

dnames <- data %>%
  select(ends_with("_d")) %>%
  names()
dnames_i <- match(dnames, colnames(data))
colnames(data)[dnames_i] <- paste0("d_", tolower(gsub("[^[:alnum:]_]", "",dummies)))
```

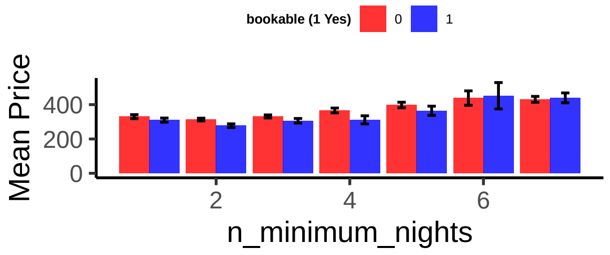
Several steps were required:

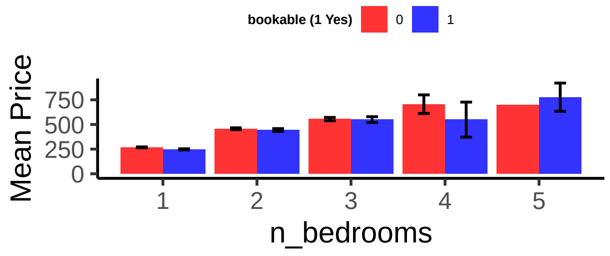
1. List all unique items provided
2. Of these select those that are not too common, but also not too special, so they make sense as predictors
3. Reformat text to make sure no observation is misscharacterized due to capitalization or multiple words used
4. Consider synonyms
5. For televisions, I grouped HDTVs separately from standard televisions

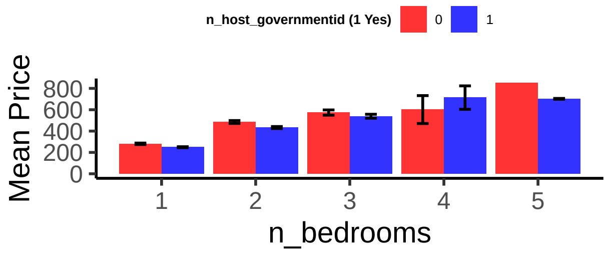
Interactions

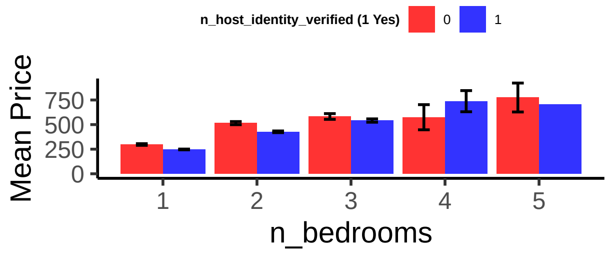
Possible interactions were chosen based graphical checks such as the ones below. As well as one theory based decision: I thought the number of reviews may have different effect on price for *superhosts* since they are trusted members with possible following.

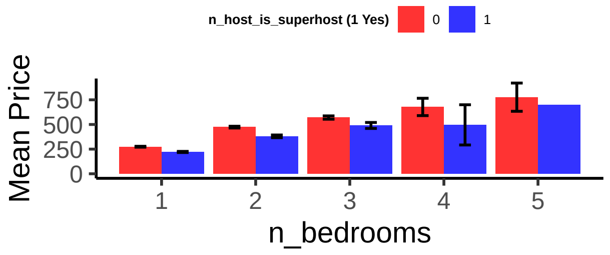






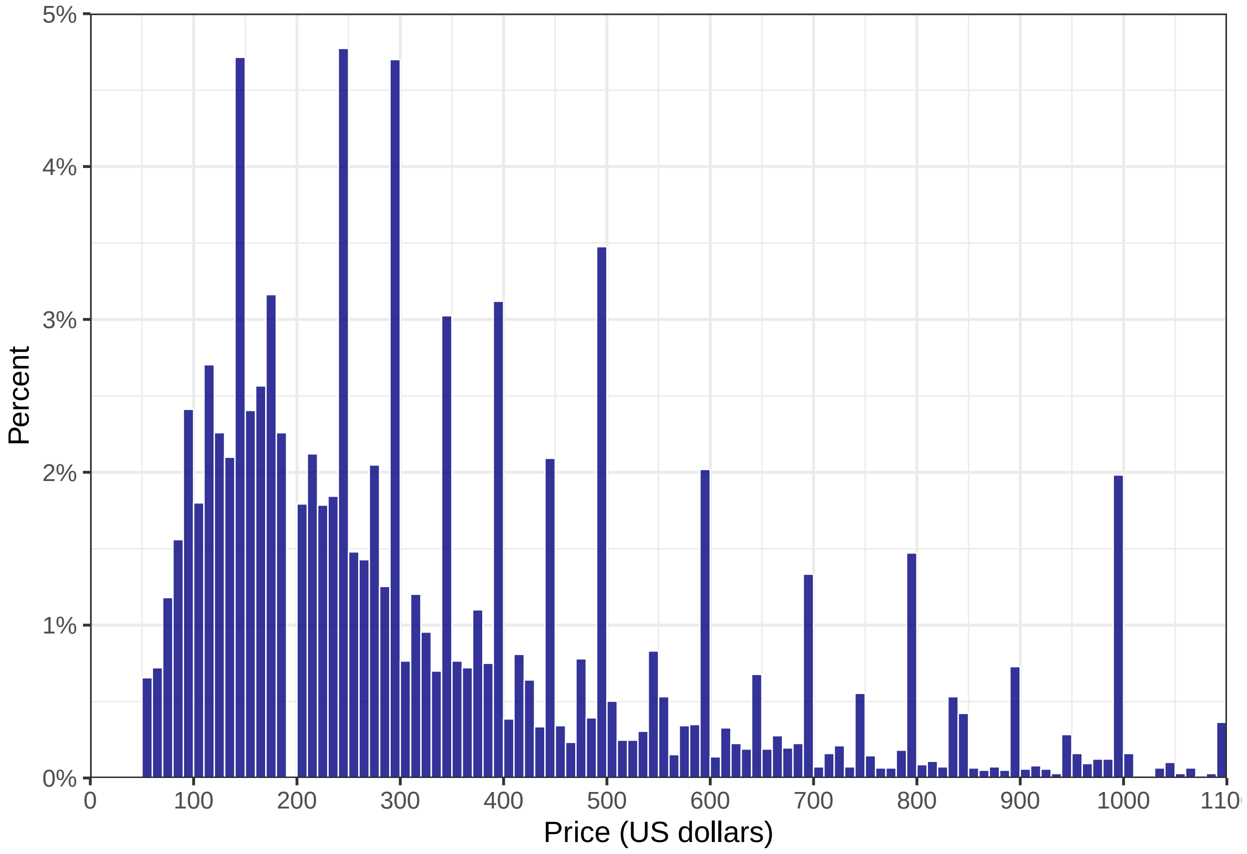






Label Engineering/Model Decisions

For label engineering, log transformation was considered, as per below histogram distribution of price is close to lognormal.



I verified the outcome by comparing my preferred OLS model with level-log outcome after transforming back the log results. It turns out log model performs somewhat worse.

level	log
187.2454	190.7424

Below are the results in test sample for all Linear models created. As mentioned none outperforms Random Forest, but Model 7-8 comes close. I highlighted in the report model 3 as it likely less overfitted, easier to explain and the results are still close.

Table 1: Cross-validated Test RMSE (Prediction error)

Model	Coefficients	R_squared	BIC	Training_RMSE	Test_RMSE
M1	2	0.1370437	148998.9	213.9724	213.9576
M2	8	0.2744357	147150.8	196.2010	196.3461
M3	32	0.3439254	146268.8	186.5692	187.2454
M4	34	0.3649134	145930.5	183.5607	184.5322
M5	36	0.3663022	145925.0	183.3599	184.3637
M6	76	0.3721116	146196.1	182.5175	184.5104
M7	92	0.3990801	145862.9	178.5548	180.6277
M8	140	0.4083356	146139.1	177.1744	180.1893

Tuning parameters

For LASSO: default tuning to find optimal Lambda. For Random Forest: 5 random variables chosen for decorrelation, minimum 20 observations stopping rule, default 500 trees.

OLS coefficients

For reference below are the coefficients of OLS model 3 referred in the working paper (Test RMSE, not holdout).

Table 2: Models 3 Airbnb

	Model 3
(Intercept)	175.9***
	(11.9)
n_accommodates	17.6***
	(1.98)
n_bedrooms	78.2***
	(5.00)
f_property_typeEntire/Unit	-30.9***
	(7.92)
f_property_typeRoom	-78.4***
	(8.81)
n_number_of_reviews	-0.838***
	(0.048)
n_minimum_nights	9.12***
	(1.19)
n_host_is_superhost	-40.8***
	(3.95)
f_neighbourhood_cleansedBotafogo	-103.2***
	(10.3)
f_neighbourhood_cleansedCamorim	-66.4***
	(16.6)
f_neighbourhood_cleansedCatete	-131.0***
	(14.0)
f_neighbourhood_cleansedCentro	-127.7***
	(10.3)
f_neighbourhood_cleansedCopacabana	-79.4***
	(7.09)
f_neighbourhood_cleansedFlamengo	-107.2***
	(12.2)
f_neighbourhood_cleansedGávea	-11.1
.	(25.1)
f_neighbourhood_cleansedGlória	-112.3***
.	(22.1)
f_neighbourhood_cleansedIpanema	2.05
.	(8.30)
f_neighbourhood_cleansedJacarepaguá	-19.7
.	(13.0)
f_neighbourhood_cleansedJardimBotânico	-18.6
.	(27.5)
f_neighbourhood_cleansedLagoa	29.7
.	(23.5)
f_neighbourhood_cleansedLaranjeiras	-112.8***
.	(15.4)
f_neighbourhood_cleansedLeblon	33.9***
.	(10.2)
f_neighbourhood_cleansedLeme	-43.2**
.	(14.8)
f_neighbourhood_cleansedRecreiodosBandeirantes	-68.7***
.	(12.2)
f_neighbourhood_cleansedSantaTeresa	-132.3***
.	(10.6)
f_neighbourhood_cleansedsmallneighboorhoods	-96.2***
.	(9.72)
f_neighbourhood_cleansedTijuca	-88.6***
.	(17.4)
n_instant_bookable	3.88
.	(3.87)
n_bathrooms_text	56.1***
.	(4.73)
n_host_governmentid	2.19
..	(5.23)
n_host_since	0.018***
..	(0.002)
n_host_identity_verified	-57.8***
..	(6.05)
RMSE	186.6