AirBnB Report

Hanna Damarjian

12/13/2021

I. Introduction.

AirBnB is a booming business that aims to assist tourists with living in residential style. With its focus on accumulating 1 billion guests per year in any of its locations of stay, pricing for travellers and hosts need to be accomodated. This project focuses on determining what factor(s) contribute to the pricing of an AirBnB listing.

II. Methodology.

A. Exploratory Data Analysis.

When initially analyzing the data on Kaggle, it appears that multiple variables have missing entries. When imported onto R, there were four variables that had missing values: **name, host name, last review, and reviews per month** When considering all variables I dropped: 1) "name" because of how sentimental it was, 2)"host_name" because there were too many unique values (levels) for this categorical variable, 3) "neighborhood" because I can use latitude/longitude to reference the specific neighborhood and b/c neighborhood has far too many unique values (categorical variable would be a mess), 4) "host_id" because I can rely upon latitude/longitude and there are too many levels for host_id, and 5)"last_review" since there are 365 days (levels) to consider. If I worked by month, then there would be a relationship between "last_review_ and"reviews_per_month".

Prior to imputation, I did two steps: (1) I converted every character variable into a factor variable prior to imputation. This would allow the imputation to recognize that if a variable is a factor, then the missing entries filled would be labeled as levels of the factor and not characters (or strings). (2) I analyzed the final missing variable, **reviews per month** (which happened to be numeric). About 80% of the data was filled and nearly 20% was missing.

The only column that needed to be imputed was **reviews per month**. I used the MICE random forest method.

B. Feature Selection and Model Building.

After imputation, I re-uploaded the data set and skimmed it using **skim()**. This allowed me to verify that the uploaded data was not being read correctly. That is, R interpreted my factor variables as character variables (strings), so they had to be converted into factors. Now, with the whole data set filled, The final

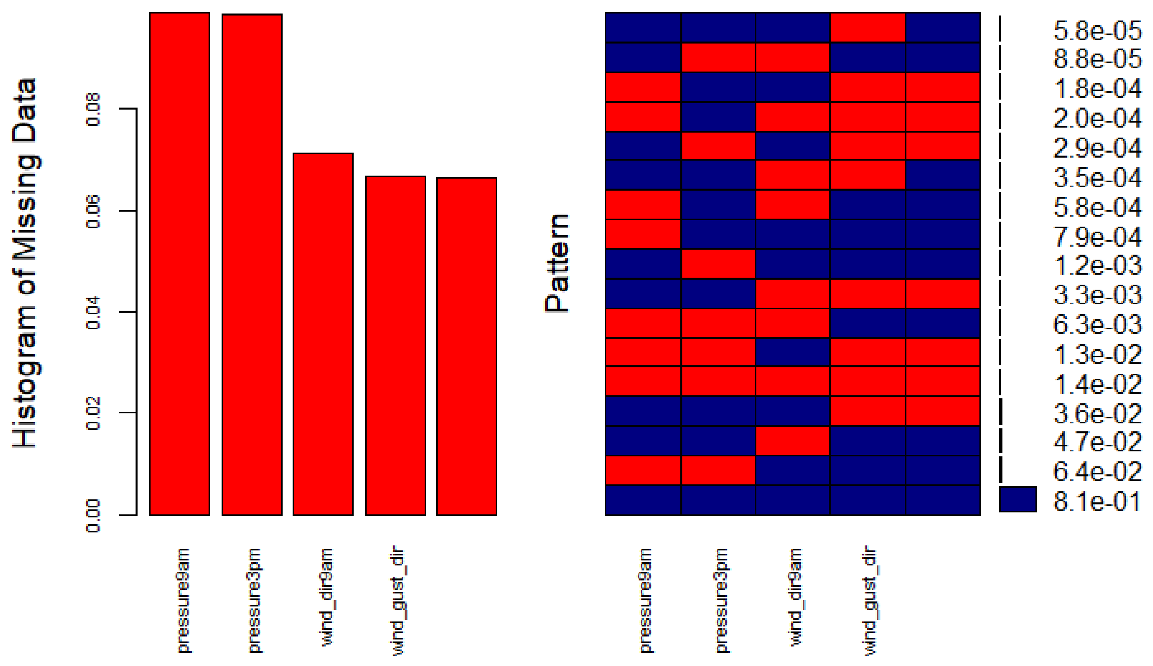

Figure 1: Top Five Missing Entries

data touch-up I made was converted all of the factor variables into dummy variables. This would aid with the feature selection and model selection.

1. First Model (Method) - Subset Selection using Backward Elimination.

The first step for this model was to build a training and testing data set. Because the number of rows is 34,226 and it is not divisible for $k \in [3, 10]$, I chose the validation set approach where we have one training and one testing data set. I used 70% of the original data set as the training data set and the final 30% as the testing data set. The best model I chose was the one with four predictors (see results).

2. Second Model (Method) - Lasso.

Similar to the first model, I did the same process. All of the variables were significant (see **Results** also for MSE).

3. Third Model - Ridge.

Results were similar to Lasso.

III. Results.

A. Subset Selection - Backward Elimination.

Final model: $\hat{price} = -31,388.95 - 426.6030 longitude + 0.1757931 availability_{365}$

$+50.24077 neighbourhoodgroup_{Manhattan} - 101.8197 roomtype_{Private}$.

MSE: 36,686.21.

B. Lasso.

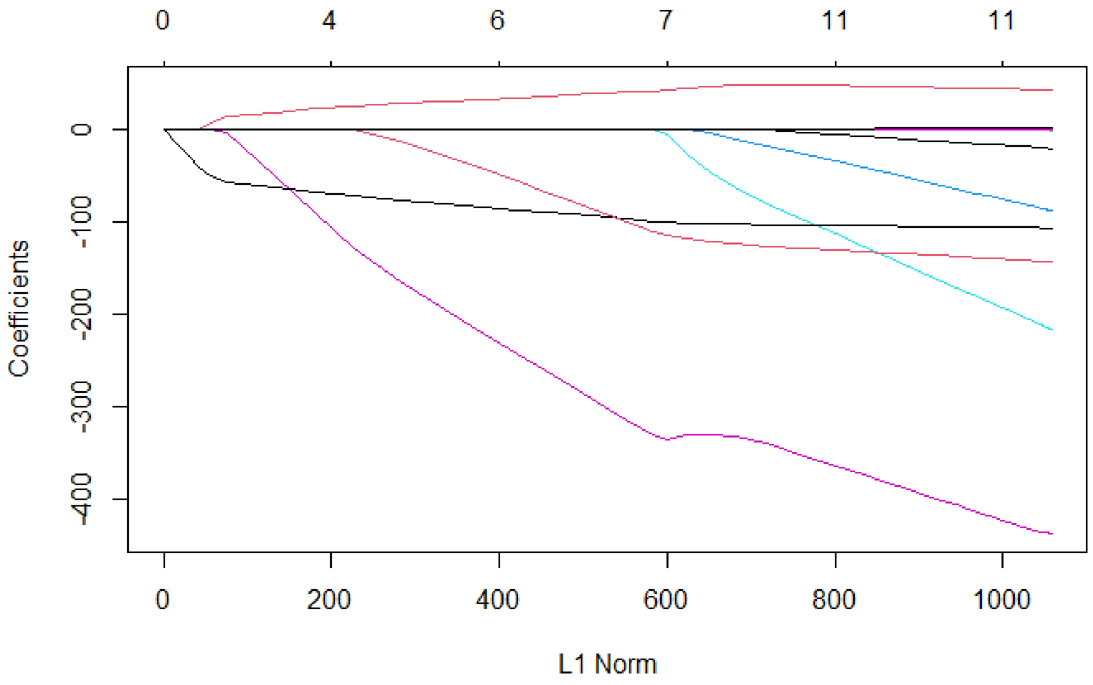

Figure 2: Lasso Model

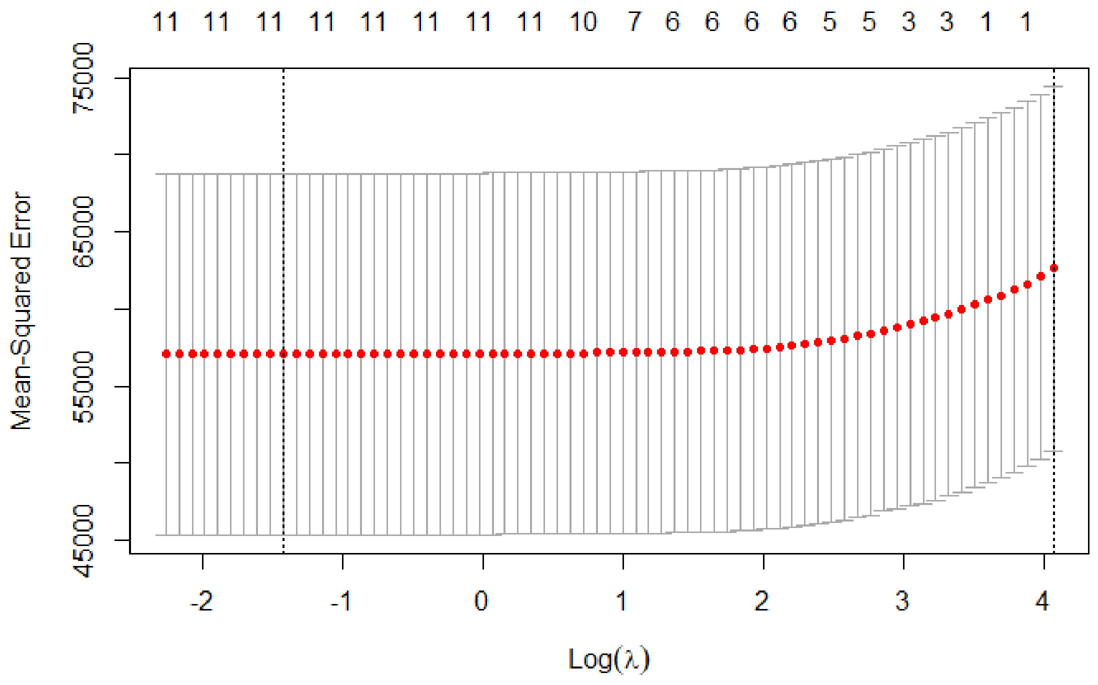

Figure 3: Best Value of Lambda under Lasso

The optimal value of $\lambda \approx 0.1324302$ with $MSE \approx 60,483.29$.

C. Ridge.

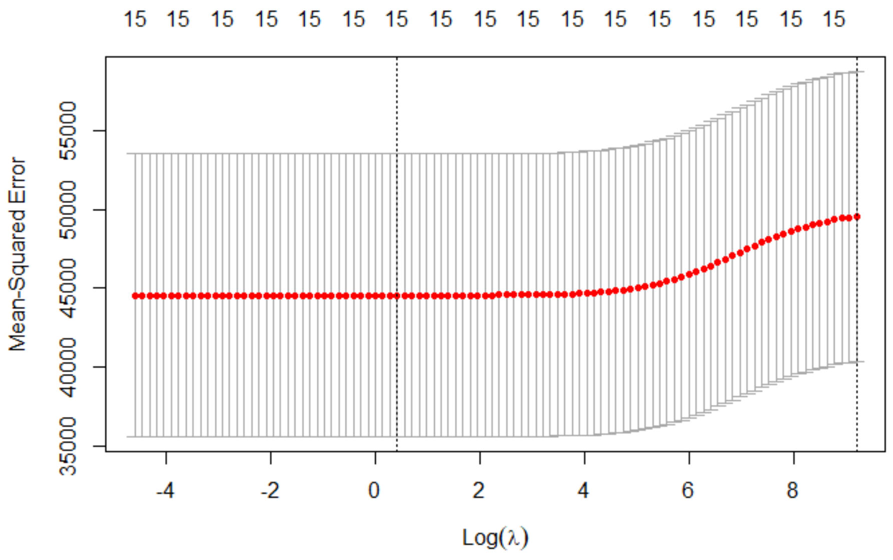

Figure 4: Best Value of Lambda under Ridge

The best value of $\lambda \approx 1.519911$ with $MSE = 60,481.27$.

IV. Conclusion.

Out of the three classification models I have depicted, I chose to use **Subset Selection (Backwards Elimination)** with the **longitude, availability365, neighbourhoodgroupManhattan, and room-typePrivateRoom** variable alone for the tested data. I plan to test out this data set with Kaggle, but need to make some imputations on the test data set on Kaggle.

Future Direction:

- Consider performing **Sentiment Analysis** on the variable **name**.
- Try another mice method, and try to determine why my PC only takes random forest.
- I would like to try GAM because it appears that the relationship between price and almost any numeric variable is negative exponential. I would consider splines for the price vs. longitude and price vs. latitude plots.
- Consider an Elastic Net Model.
- Utilize other methods for model selection (C_p and BIC).

V. Appendix. All of my code was commented to easily compile this document. My PC is very slow when it comes to compiling code under SVM and random forests.

```
# abb <- read.csv("C:\\Users\\johnd.LAPTOP-35N364TU\\OneDrive\\Desktop\\PA projects\\train.csv")
#
# # Goal: We want to predict the price of an Airbnb room.
# # The ‘Airbnb effect’ is to some extent remarkably similar to
# # gentrification in that it slowly increases the value of an area
# # to the detriment of the indigenous residents,
# # many of whom are pushed out due to financial constraints.
#
# # Cities, popular ones especially, seem to fare the worst.
# # In major cities such as Amsterdam, Barcelona, Edinburgh, and Los Angeles,
# # studies on the ‘Airbnb effect’ have found that over-tourism facilitated
# # by platforms such as Airbnb negatively impacts on house
# # prices and communities.
#
#
#
# # 1. Exploratory Data Analysis:
# # A. Dataframe
# nrow(abb) # 34,226 rows
# length(abb) # 16 columns
#
# # summary(ar) --> summary statistics for each variable
# library(skimr)
# # skim(ar) --> shows the data type, mean, sd, and hist for each variable
#
#
# # B. Missing values (NA)/drop which variables?:
# total_entries <-  sum(is.na(abb)) #  14039 missing entries
#
# percentage_of_filled_entries <-
#   100*(total_entries-sum(is.na(abb)))/total_entries
# percentage_of_filled_entries # about  of cells are filled.
#
# # Which variables (columns) contain NA?
# # What is the percentage of NA's for each variable?
#
# colSums(is.na(abb))/nrow(abb)
# # Missing data: name, host name (9152 unique values),
# # last review, and reviews/month.
#
# # ---------------------------------------------------------------
# # 1) I dropped "name" because of how sentimental it was.
# # NOTE: Sentimental Analysis (for future can be a consideration!)
# # 2) I dropped "host_name" because there were too many unique
# # values (levels) for this categorical variable.
# # 3) "neighborhood" because I can use latitude/longitude to reference
# # the specific neighborhood and b/c neighborhood has far too many
# # unique values (categorical variable would be a mess).
# # 4) "host_id" because I can rely upon latitude/longitude and there
# # are too many levels for host_id.
# # 5) "last_review" since there are 365 days (levels) to consider.
# # If I worked by month, then there would be a relationship between
# # "last_review_ and "reviews_per_month".
# # ---------------------------------------------------------------
#
# abb <- subset(abb, select=-c(name,host_name,host_id,
#                              neighbourhood,last_review))
#
# # We have 11 variables left to consider.
#
#
#
# # C. Change the string variables into factors:
#
# # Change all character variables into factor variables:
# # Use skim(abb) to determine which string variables needed to be factored:
# # skim(abb)
# abb$neighbourhood_group <- as.factor(abb$neighbourhood_group)
# abb$room_type <- as.factor(abb$room_type)
#
#
# # D. Imputation (MICE):
# # Imputating the numeric variable "reviews_per_month" using :
```

```
# library(mice)
#
#
# abb2 <- abb # keep an extra copy of original dataset to compare
# #arNEW <- subset()
# library(VIM) # used for aggregate plot
#
#
# top_rpm_me <- subset(abb, select=c('reviews_per_month'))
# aggr_plot <- aggr(top_rpm_me, col=c("navyblue","red"), numbers=TRUE, sortVars=TRUE,
#                    labels=names(top_rpm_me), cex.axis=1, gap=3, ylab =
#                      c("Histogram of Missing Data","Pattern"))
# # Explain.
#
#
# # Which method to choose to impute?
# hist(abb$reviews_per_month, breaks=25, xlim = c(0,12),
#      main = "Reviews per Month", col = "green",
#      ylab = "Frequency")
#
# # Do not use random forest for numerical because prediction \in  range of variable: https://bmcmedresmethodol.biomedcent
#
#
# # Before imputation: analyze correlation plot of continuous variables:
# abb.subset <- subset(x = abb, select = c(latitude, longitude, price, minimum_nights,
#                                          number_of_reviews,
#                                          calculated_host_listings_count,
#                                          availability_365))
#
# cor <- cor(abb.subset, method = "pearson")
# library(corrplot)
# corrplot(cor, type = "full", order = "hclust", tl.col = "black", tl.srt = 45)
#
#
# # Finally, impute using random forest (what else could I have done differently?):
#
#
#
# imputed_abb <- mice(abb, m=1, method="rf")
# # View(complete(imputed_ar)) check out the dataset and compare to original "abb"
# finished_imputed_abb <- complete(imputed_abb,1)
# dim(finished_imputed_abb)
# sapply(finished_imputed_abb, function(x) sum(is.na(x)))
#
# write.csv(finished_imputed_abb,
#           "C:/Users/johnd.LAPTOP-35N364TU/OneDrive/Desktop/Predictive Analytics/abb_imp.csv")
#
#
#
#
#
# # Use imputed data set:
#
# abb <- read.csv("C:/Users/johnd.LAPTOP-35N364TU/OneDrive/Desktop/Predictive Analytics/abb_imp.csv")
# sum(is.na(abb))
#
#
# # 1. Factorize the variables:
# abb$neighbourhood_group <- as.factor(abb$neighbourhood_group)
# abb$room_type <- as.factor(abb$room_type)
#
# names(abb)
#
# # 2. Omit the row-counter column:
# abb <- subset(abb, select=-c(X))
# abb <- subset(abb, select = -c(id))
#
# # 3. Create dummy variables:
# library(fastDummies)
# abb_dummy <- dummy_cols(abb, select_columns =
#                       c('neighbourhood_group','room_type'),
#                       remove_selected_columns=TRUE)
#
#
#
# # B. GAM
# plot(abb$latitude,abb$price) # try spline
# plot(abb$longitude,abb$price) # try spline
# plot(abb$minimum_nights,abb$price) # try -exponential
# lines(abb$minimum_nights,exp(-1*abb$price))
# plot(abb$number_of_reviews,abb$price) # exponential
# plot(abb$reviews_per_month,abb$price) # exponential
# plot(abb$calculated_host_listings_count,abb$price) # exponential
# plot(abb$availability_365,abb$price) # no pattern
# ```
#
#
# # A. Subset Selection - Backwards Elimination:
# library(leaps)
# # Create model using Backward SS (and perform CV using the validation set approach):
# set.seed(123)
# train <- sample(c(TRUE, FALSE), nrow(abb_dummy),replace = TRUE)
# test <- (!train)
# regfit.best <- regsubsets(price ~ .,
#                           data = abb_dummy[train,], nvmax = 1000, method="backward")
#
#
# test.mat <- model.matrix(price ~ ., data = abb_dummy[test,])
#
# val.errors <- rep(NA, 7)
#
# for (i in 1:7) {
# coefi <- coef(regfit.best, id = i)
# pred <- test.mat[,names(coefi)] %*% coefi
# val.errors[i] <- mean((abb$price[test] - pred)^2) #RSS
# }
#
# val.errors
# which.min(val.errors)
# coef(regfit.best, 4)
# # I chose MSE is 36,686.21 where the model includes four variables. Model is:
# #                 (Intercept)                    longitude              availability_365
# #               -3.138895e+04                -4.266030e+02                 1.757931e-01
# # neighbourhood_group_Manhattan      `room_type_Private room`
# #                5.024077e+01                -1.018197e+02
#
#
```

```
# # B. Lasso

# # Why do you perform 2-fold cross validation?

#

# set.seed(123)

# x2 <- model.matrix(price~.,data=abb_dummy)[,-1] # omits the first intercept column.

# y2 <- abb_dummy$price

# train2 <- sample(1:nrow(x2), .7*nrow(abb_dummy)) # 57 since the above was 43

# test2 <- (-train2)

# y.test2=y2[test2]

# # Create Lasso model:

# library(glmnet)

# ## Loading required package: Matrix

# ## Loaded glmnet 4.1-2

# grid <- 10^(seq(10,-2,length=100))

# lasso.mod <- glmnet(x2[train2,],y2[train2],alpha=1,lambda=grid)

#

# plot(lasso.mod)

#

#

# # Perform cross-validation on lambda using built-in function "cv.glmnet()":

# cv.out=cv.glmnet(x2[train2,], y2[train2], alpha=1)

# plot(cv.out)

#

# # Minimum value of lambda:

# bestlam <- cv.out$lambda.min

# bestlam

#

# cv.out$lambda.1se

#

# lasso.pred <- predict(lasso.mod,s=bestlam, newx=x2[test2,])

# mean((lasso.pred - y.test2)^2)

#

#

# lasso.mod <- glmnet(x2,y2,alpha=1, lambda = bestlam)

#

# coef(lasso.mod)[,1]

# # Use Lasso - MSE is 60483.29

#

# # (Intercept)                              latitude
# #                       -2.578007e+04                         -2.030132e+02
# #                           longitude                        minimum_nights
# #                       -4.615259e+02                         -1.766886e-01
# #                   number_of_reviews                     reviews_per_month
# #                       -2.958136e-01                         -2.740601e-01
# #      calculated_host_listings_count                      availability_365
# #                       -1.516214e-01                          1.960911e-01
# #            neighbourhood_group_Bronx        neighbourhood_group_Brooklyn
# #                        0.000000e+00                         -2.770325e+01
# #       neighbourhood_group_Manhattan          neighbourhood_group_Queens
# #                        3.494691e+01                         -5.516106e+00
# # `neighbourhood_group_Staten Island`         `room_type_Entire home/apt`
# #                       -1.315513e+02                          1.063793e+02
# #             `room_type_Private room`              `room_type_Shared room`
# #                        0.000000e+00                         -3.368191e+01
# # ----------------------------------------------------------
#
# # Ridge Method
#
# set.seed(123)
# abb.train <- abb_dummy[train2,]
# abb.test <- abb_dummy[test2,]
# train.mat <- model.matrix(price~.,data=abb.train)
# test.mat <- model.matrix(price~.,data=abb.test)
# grid <- 10^seq(from=4,to=-2,length=100)
# library(Matrix) # needed for Ridge
# # Build Ridge model:
# ridge.mod <- cv.glmnet(train.mat, abb.train[,"price"],alpha=0,lambda=grid,thresh=1e-12)
#
# plot(ridge.mod)
# # cv.out <- cv.glmnet(x_matrix, y_resp, alpha=0,lambda=best_lambda from grid)
# # Best Lambda:
# lambda.best <- ridge.mod$lambda.min
# lambda.best
#
# ridge.pred <- predict(ridge.mod, newx=test.mat,s=lambda.best)
# mean((abb.test[,"price"]-ridge.pred)^2)
# # 60481.27 --> MSE is very large compared to Lasso and Backward Elimination
#
```

```
# coef(ridge.mod)[,1]
#        #                  (Intercept)                         (Intercept)
#        #                -1.203514e+03                        0.000000e+00
#        #                     latitude                           longitude
#        #                 3.049935e+00                       -1.663037e+01
#        #               minimum_nights                   number_of_reviews
#        #                 5.572719e-03                       -5.302572e-03
#        #            reviews_per_month      calculated_host_listings_count
#        #                -1.099509e-01                        8.907601e-03
#        #             availability_365            neighbourhood_group_Bronx
#        #                 3.116248e-03                       -1.413315e+00
#        # neighbourhood_group_Brooklyn        neighbourhood_group_Manhattan
#        #                -9.525402e-01                        1.645822e+00
#        #   neighbourhood_group_Queens `neighbourhood_group_Staten Island`
#        #                -1.364472e+00                       -6.136071e-01
#        #  `room_type_Entire home/apt`              `room_type_Private room`
#        #                 2.620622e+00                       -2.478820e+00
#        #      `room_type_Shared room`
#        #                -1.697011e+00
```