# My Kaggle Project Report

Dinghao Xu

November 29, 2021

**Abstract**

Used Random Forest model, Ranger model, and XGBoost. The XGBoost gave me the best results with a score of 66.94153 on the public board and 61.54859 on the private board .

## 1 Initial exploration

When I first get the data, I have no idea about how to deal with it. I am new to R studio, which means I just learned for two months. To begin with, I used linear regression to select several features and used the adjusted R-squared to see if the value increased or not.

I used str to see the category of each feature. And I was trying to figure which features I should use to get the lowest RMSE score.

I used the linear regression model to see the numeric and integer features and selected several features like accommodates, bathrooms, bedrooms, beds, security_deposit, guests_included, extra_people, minimum_nights, number_of_reviews,review_scores_rating, and review_scores_accuracy.

For the features, I thought the beds, bathrooms, bedrooms, security_deposit, guests_included, extra_people, minimum_nights, property_type, and cleaning_fee would be included for a basic model.

Then, I noticed that there's NA in beds and security_deposit. Thus, I substituted the NA with the mean value of the rest. After running a NA test for the whole dataset, I found that I needed to clean five columns with NA because of 83 columns with zero NA and 3 with NA larger than 15 percent. And the NA's included can not be predicted in the scoring data. And NA's in host_response_rate and host_response_time can not be cleaned by simply selecting them out and substituting for the mean value. Then, I added a NA string to clean all types of NA with a typo or multiple words. After the cleaning step, the model selections are applied.

```
## Library packages
library(tidyverse)
library(dplyr)
library(caTools)
library(caret)
library(readr)
library(fastDummies)
## READ IN DATA HERE
```

```
df = read.csv('analysisData.csv')
## INITIAL EXPLORATION STARTS HERE
str(data)
sum(is.na(data$cleaning_fee))
# result: 6366 NAs
df[df=="?"] <- NA
table(colSums(is.na(df)))
#result:0     1     2     6  5958 38157 40972 41126
#       83     1     1     2     1     1     1     1

# Make dummy variables
dummy_cols(df$property_type,remove_first_dummy = TRUE)
# Cleaning the NAs
mean(df$beds,na.rm=TRUE)
df$beds[which(is.na(cdf$beds))]<- as.integer("1.577189")
mean(df$security_deposit,na.rm=TRUE)
df$security_deposit[which(is.na(df$security_deposit))]<- as.integer("156.9591")
mean(df$cleaning_fee,na.rm=TRUE)
df$cleaning_fee[which(is.na(df$cleaning_fee))]<- as.integer("64.36677")
mean(df$square_feet,na.rm=TRUE)
df$square_feet[which(is.na(df$square_feet))]<- as.integer("690.3318")
mean(df$reviews_per_month,na.rm=TRUE)
df$reviews_per_month[which(is.na(df$reviews_per_month))]<- as.numeric("1.351986")

## INITIAL EXPLORATION ENDS HERE
```

## 2 Models and Feature Selection

By using linear regression, I can compare with different variable selections. Then, I used the random forest to test the impact with other variables. I tried several ntrees for the random forest, and the RMSE score on Kaggle and my test data shows that the higher ntree value(up to 1000) will generate a lower RMSE.

And I tried several combination of the features, the results showed out that the combination of "accommodates+bathrooms+bedrooms+beds+ security_deposit+guests_included+extra_people +minimum_nights+number_of_reviews+review_scores_rating +review_scores_accuracy" genreated the lowest RMSE on the kagggle public board.

I tried the tuned method several times, but my computer could not handle that method. Thus, I tried the Ranger method to see whether my RMSE score lowered.

At this time, I added the features property type, cancellation policy, and zip code, cleaning_fee, square_feet, host_acceptance_rate, host_total_listings_count, and weekly_price. I loaded the package "fastDummies" to create the dummy variables for the property type and cancellation policy. And then, I clean the zipcode, cleaning fee, square_feet, and host_acceptance_rate for the ranger model.

The ranger model lowered my RMSE to 73.39, and I selected 800 as the number of trees because the RMSE scorer goes up for 500 and 1000. Thus, I set 800 as my final number of trees.

The third method I used is the XGBoost method. The strange thing is that when I cleaned the model in XGBoost, the RMSE score went up. After I remove all the variables cleaned, the score goes lower. There might be something wrong with that. Then, I just selected the variables and tried for several n rounds, and the RMSE score will be significantly decreased from the other models. Thus, I chose XGBoost as the method that went through my final submission and scored 66.94153 on the public board and 61.54859 on the private board.

```
## MODELING AND FEATURE SELECTION STARTS HERE
## Variables Selection STARTS HERE
setwd('~/Desktop/APAN5200/KAGGLE')
df<- read.csv('analysisData.csv')
library(caret)
library(tidyverse)
library(dplyr)
library(caTools)
library(randomForest)

lm0<-lm(price ~ accommodates+bathrooms+bedrooms+beds+security_deposit
        +guests_included+extra_people+minimum_nights+number_of_reviews
        +review_scores_rating+review_scores_accuracy,data=train)
summary(lm0)

lm1<-lm(price ~ accommodates+bathrooms+bedrooms+beds+security_deposit
        +guests_included+extra_people+minimum_nights+number_of_reviews
        +review_scores_rating+review_scores_accuracy+property_type
        +cleaning_fee+square_feet+availability_30+number_of_reviews
        +review_scores_accuracy+review_scores_rating+is_location_exact
        +reviews_per_month+review_scores_cleanliness+review_scores_checkin
        +review_scores_location+review_scores_value,data=df)

summary(lm1)

lm2<-lm(price ~ accommodates+bathrooms+bedrooms+beds+security_deposit
        +guests_included+extra_people+minimum_nights+number_of_reviews+property_type
        +cleaning_fee+availability_30+number_of_reviews+review_scores_accuracy
        +review_scores_rating+host_acceptance_rate+host_total_listings_count
        +zipcod, data=df)

summary(lm2)

lm3<-lm(price ~ accommodates+bathrooms+bedrooms+beds+security_deposit
        +guests_included+extra_people+minimum_nights+number_of_reviews
        +review_scores_rating+review_scores_accuracy+property_type
        +cleaning_fee+square_feet+availability_30+number_of_reviews
        +review_scores_accuracy+review_scores_rating+is_location_exact
        +reviews_per_month+review_scores_cleanliness+review_scores_checkin
```

```
        +review_scores_location+review_scores_value+zipcode,data=df)
summary(lm3)
str(df)

lm4<- lm(price ~ accommodates+bathrooms+bedrooms+beds+security_deposit
         +guests_included+extra_people+minimum_nights+number_of_reviews
         +review_scores_rating+review_scores_accuracy+property_type
         +cleaning_fee+availability_30+number_of_reviews+review_scores_accuracy
         +review_scores_rating+is_location_exact+reviews_per_month
         +review_scores_cleanliness+review_scores_checkin
         +review_scores_location+review_scores_value+zipcode
         +host_response_rate+host_location+host_listings_count
         +neighbourhood_group_cleansed+neighbourhood_cleansed
         +instant_bookable+require_guest_phone_verification+street,data=df)


summary(lm4)
## Variables Selection ENDS HERE

## MODEL 1: Random Forest STARTS HERE
setwd('~/Desktop/APAN5200/KAGGLE')
df<- read.csv('analysisData.csv')
library(caret)
library(tidyverse)
library(dplyr)
library(caTools)
library(randomForest)
df[df=="?"] <- NA
colSums(is.na(df))
df$beds[which(is.na(df$beds))]<- as.integer("1")
mean(df$security_deposit,na.rm=TRUE)
df$security_deposit[which(is.na(df$security_deposit))]<- as.integer("156.7857")
set.seed(1031)
split <- sample.split(df$price, SplitRatio = .8)
train <- df[split,]
test <- df[!split,]
forest = randomForest(price ~ accommodates+bathrooms+bedrooms+beds+security_deposit
                      +guests_included+extra_people+minimum_nights
                      +number_of_reviews+review_scores_rating+review_scores_accuracy,
                      train, ntree = 1000)
pred_train = predict(forest)
rmse_train_forest = sqrt(mean((pred_train - train$price)^2)); rmse_train_forest
pred_forest = predict(forest, newdata= test)
rmse_forest = sqrt(mean((pred_forest - test$price)^2)); rmse_forest

scoringData = read.csv('scoringData.csv')
colSums(is.na(scoringData))
```

```r
scoringData$beds[which(is.na(scoringData$beds))]<- as.integer("1")
mean(scoringData$security_deposit,na.rm=TRUE)
scoringData$security_deposit[which(is.na(scoringData$security_deposit))]<- as.integer("157.654")
pred = predict(forest, newdata= scoringData)
submissionFile = data.frame(id=scoringData$id, price=pred$predictions)
write.csv(submissionFile, 'submission.csv', row.names=FALSE)
## MODEL 1: Random Forest ENDS HERE

## MODEL 2: Ranger Model STARTS HERE

library(caret)
library(tidyverse)
library(dplyr)
library(caTools)
library(caret)
library(ranger)
library(readr)
library(fastDummies)
library(dplyr)

set.seed(1031)
na_strings <- c("NA", "N A", "N / A", "N/A", "N/ A", "Not Available", "NOt available")
df<- read.csv('analysisData.csv',na=na_strings)
scoringData = read.csv('scoringData.csv')
scoringData$zipcode <- as.character(scoringData$zipcode)
combinedData <- bind_rows(df, scoringData)
#Data cleanning
combinedData$zipcode <- substr(combinedData$zipcode, 1, 5)
combinedData$zipcode <- as.factor(combinedData$zipcode)
combinedData$zipcode <- forcats::fct_lump_n(combinedData$zipcode, 40)

dummy_cols(combinedData$property_type,remove_first_dummy = TRUE)
mean(combinedData$beds,na.rm=TRUE)
combinedData$beds[which(is.na(combinedData$beds))]<- as.integer("1.577189")
mean(combinedData$security_deposit,na.rm=TRUE)
combinedData$security_deposit[which(is.na(combinedData$security_deposit))]<- as.integer("156.959
mean(combinedData$cleaning_fee,na.rm=TRUE)
combinedData$cleaning_fee[which(is.na(combinedData$cleaning_fee))]<- as.integer("64.36677")
mean(combinedData$square_feet,na.rm=TRUE)
combinedData$square_feet[which(is.na(combinedData$square_feet))]<- as.integer("690.3318")
mean(combinedData$reviews_per_month,na.rm=TRUE)
combinedData$reviews_per_month[which(is.na(combinedData$reviews_per_month))]<- as.numeric("1.351
summary(df$host_acceptance_rate)

mean(parse_number(combinedData$host_acceptance_rate),na.rm=TRUE)
combinedData$host_acceptance_rate[which(is.na(combinedData$host_acceptance_rate))]<- as.numeric(
mean(parse_number(combinedData$host_response_rate),na.rm=TRUE)
```

```r
combinedData$host_response_rate[which(is.na(combinedData$host_response_rate))]<- as.numeric(("93


dummy_cols(combinedData$cancellation_policy,remove_first_dummy = TRUE)

# numeric variables median imputation
numeric_predictors <- which(colnames(combinedData) != "price" &
                              sapply(combinedData, is.numeric))

imp_model_med <- preProcess(combinedData[,numeric_predictors], method = 'medianImpute')
combinedData[,numeric_predictors] <- predict(imp_model_med, newdata=combinedData[,numeric_predic

## split back the Data
train <- combinedData[!is.na(combinedData$price),]
scoringData <- combinedData[is.na(combinedData$price),]

forest_ranger = ranger(price~accommodates+bathrooms+bedrooms+
                         beds+security_deposit+guests_included
                       +extra_people+minimum_nights+number_of_reviews
                       +property_type+cleaning_fee+square_feet
                       +availability_30+review_scores_accuracy
                       +review_scores_rating+host_acceptance_rate
                       +host_total_listings_count+zipcode+weekly_price,
                       data = train,
                       num.trees = 800)
pred_train = predict(forest_ranger, data = train, num.trees = 800)
rmse_train_forest_ranger = sqrt(mean((pred_train$predictions - train$price)^2))
rmse_train_forest_ranger

pred_forest_ranger = predict(forest_ranger, data= scoringData, num.trees = 800)
submissionFile = data.frame(id=scoringData$id, price=pred_forest_ranger$predictions)
write.csv(submissionFile, 'submissionRanger.csv', row.names=FALSE)
## MODEL 2: Ranger Model ENDS HERE

## MODEL3: XGBoost Model STARTS HERE
setwd('~/Desktop/APAN5200/KAGGLE')

library(caret)
library(tidyverse)
library(dplyr)
library(caTools)
library(caret)
library(rpart)
library(rpart.plot)
library(randomForest)
library(ranger)
library(xgboost)
```

```
library(readr)
library(gbm)
library(fastDummies)
library(dplyr)
library(vtreat)

# Read the Data
set.seed(617)
df<- read.csv('analysisData.csv')
scoringData = read.csv('scoringData 3.csv')

# CLEAN THE ZIPCODE
df$zipcode<-as.numeric(df$zipcode)
scoringData$zipcode<-as.numeric(scoringData$zipcode)

#Model

train= df[,c('room_type','accommodates','bathrooms',
             'bedrooms','beds','bed_type','security_deposit',
             'guests_included','extra_people','minimum_nights','maximum_nights',
             'number_of_reviews','property_type','cleaning_fee',
                'availability_30',
             'review_scores_rating','review_scores_accuracy','review_scores_cleanliness',
             'review_scores_checkin','review_scores_communication','review_scores_location'
             ,'review_scores_value','cancellation_policy','reviews_per_month',
             'calculated_host_listings_count','host_acceptance_rate',
             'host_total_listings_count','zipcode','price')]


trt = designTreatmentsZ(dframe = train, varlist = names(train)[1:28])

train_input = prepare(treatmentplan = trt,
                      dframe = df)
test_input = prepare(treatmentplan = trt,
                     dframe = scoringData)

xgboost = xgboost(data = as.matrix(train_input),
                  label = df$price,
                  nrounds=82,
                  verbose = 0)

xgboost$evaluation_log

#Predict the Scoring Data using XGboost
pred= predict(xgboost, newdata=as.matrix(test_input))
(rmse_xgboost <- RMSE(pred, scoringData$price))
```

```r
submissionFile = data.frame(id=scoringData$id, price=pred)
write.csv(submissionFile, 'submissionXGBOOST.csv', row.names=FALSE)

## MODEL3: XGBoost Model ENDS HERE

## MODELING AND FEATURE SELECTION ENDS HERE
```

# 3 Model Comparison

| Model from previous section | RMSE on training data | RMSE for test data holdout or CV | RMSE on Kaggle | Other notes |
|---|---|---|---|---|
| Model 1: Random Forest | 80.66462 | 79.13791 | 85.03685 | |
| Model 2: Ranger | 33.94413 | | 73.39 | Added several features selected for model 1 |
| Model 3: XGBoost | | | 66.94153 | Final Kaggle RMSE: 61.54859 |

# 4 Discussion

The XGBoost gave me the best results with a score of 66.94153 on the public board and 61.54859 on the private board.

The XGBoost model was designed to handle a large amount of dirty data, and I verified that by using other methods like Random Forest and Ranger model. The RMSE scores of the other two models will be higher than the XGBoost model. Thus, I think the XGBoost model is the top-performing model for the 'cleaned-up data.

I can use the model to predict the prices between the predictor variables and Airbnb prices.

# 5 Future directions

If I had more time, I would try more variables in the XGBoost model and do more essential cleanings. I cleaned the zipcode in the XGBoost model, and the score went up a lot. However, I cleaned others as well, which led to an increase in the final RMSE score. Also, I will try to use the lassoModel to select the variables instead of using the linear regression model.