APAN5200: APPLIED ANALYTICS FRAMEWORKS & METHODS I

Kaggle Project Final Report

Keqi Yu

Abstract: In this project, five predictive models have been used to predict the price of a rental in Airbnb. Linear regression model was the first model that has been used and it came with a relatively high rmse. The second model that I have tried was the Decision Tree model and the rmse was a little bit lower than linear regression model. Moreover, the third model I used was regular boosting model gbm model with a lower RMSE than last two model. Then, xgboost model is the fourth model I have used, resulting in higher RMSE, but a lower score in Kaggle. Last but not least, the Random Forest Model is the last model I have tried, and the rmse was a little bit lower than xgboost model, but a lowest score in Kaggle

1. Data Exploration

For the analysisData dataset, there are 41330 observations and 91 variables. There are three class of the variables, respectively integer, character, binary, categorial and numeric. For example, character variables include name, summary, space, description, neighborhood overview, notes, transit, access, interaction, and house_rules. Binary variables are logistic variables with either TRUE or FALSE. In this case, variables like host_is_superhost, host_has_profile_pic, host_ identity_verified, is_location_exact, has_availability, requires_license, instant_bookable, is_business_travel_ready are all binary variables with either TRUE or FALSE. Categorial variables are mostly variables describing location or type, such as street, neighborhood, city, state, room_type, and bed_type. Numerical data are mostly variables describing number or rating, such as bedrooms, beds, number_of_reviews, and review_scores_rating. The numerical variables is the most applicable variables and it is easy to apply into the model we try to create. Furthermore, after carefully looking up the dataset, we can find that the class of the variables can be divided into numerical, binary, character, and factor. Some numerical variables actually belong to factor, with several levels, because the value actually has no meaning. So, I used to_factor_from_numeric function to convert some numerical variables to factor. The variables review_scores_accuracy and the other five variables have been converted to the factor.

```{r}
#read csv
analysis <- read.csv("analysisData.csv")
scoring <- read.csv("scoringData.csv")
```

```{r}
#source("data_manipulation_functions.R")
to_factor_from_numeric <- function(current_var){
  x <- NULL
  if(is.numeric(current_var) && length(unique(current_var)) < 10){
    x <- as.factor(current_var)
  }else{
    x <- current_var
  }
  return(x)
}
analysis_new <- analysis %>% mutate_all(to_factor_from_numeric)
scoring_new <- scoring %>% mutate_all(to_factor_from_numeric)
```

2. Data Cleaning

To get start, the first problem I have encountered is that some variables have many missing values. So, I decided to remove the variables with high rate of missing values. The variables like square_feet, weekly_price and other 5 variables have very high rate of missing values, so I removed them, resulting in 7 variables being removed and 84 variables remained. Moreover, after removing these variables with high rate of NAs, we can find that there are still some variables with a few missing values, such as beds, host_listings_count, host_total_listings_count. So, I impute the missing values with the mean. After fixing the problem of missing values, I found that there are many complex but not useful character variables. These variables like summary, notes are mostly about the description of a rental, which is not useful to include in the model. Some variables are not significant, some variables are not significant. Including unnecessary variables in the model will impact the quality of the model. Thus, I decided to include the variables which I thought it is relevant with the price. Also, for the binary variables with either "t" or "f", I would like to convert to 0 and 1 for the further investigation.

```
##Useful data
```{r}
useful <- c("host_is_superhost","host_has_profile_pic",
            "host_identity_verified","neighbourhood_group_cleansed",
            names(analysis_new)[39:43], names(analysis_new)[45:55],
            names(analysis_new)[58:63], names(analysis_new)[66:72],
            "instant_bookable", names(analysis_new)[76:82])
data <- analysis_new[,useful]
```
```

```
```{r}
for (i in 1:ncol(data)){
  if ("t" %in% data[,i] | "f" %in% data[,i]| "" %in% data[,i]){
    data[,i] <- as.character(data[,i])
    data[which(data[,i]=="f"),i] <- 0
    data[which(data[,i]=="t"),i] <- 1
    data[which(data[,i]==""),i] <- NA
    data[,i] <- as.factor(data[,i])
  }
}
```
```

Moreover, when we applied the Decision Tree model, Random Forest Model and Boosting model, we found that these models require all the variables to be numerical or factor. So, I convert all the variables to numeric. For character data, we convert them to factor first, then to numeric.

```r
#Data type conversion to numeric type
datanum <- data
num.index <- c(1:3,5:8,10:27,36:41)
for (i in num.index){
  datanum[,i] <- as.numeric(datanum[,i])
}
char.index <- c(4,9,28:35)
for (i in char.index){
  datanum[,i] <- as.factor(datanum[,i])
  datanum[,i] <- as.numeric(datanum[,i])
}
```

3. Model and Feature Selection

After data exploration and data cleaning process, the first model I used is the linear regression model, since it is the first model we studied in the class, and it is easy to apply. After running the linear regression model, the RMSE we got is 142.293. After I submitted to Kaggle, the score we got is 106.32416. After we tried the linear regression, the second model I have tried is the decision tree model. We will examine a tree with default cp of 0.01. The result we got is much lower than the linear regression, which is 133.1909 and 86.37468 score in the Kaggle.

Moreover, after testing the linear regression and decision tree model, I think the boosting model would be better. Like bag and forest models, boosting models are ensemble models that derive predictions from a number of trees. The key difference is that in boosting, trees are grown sequentially, each tree is grown using information from previously grown trees. Thus, boosting can be seen as a slow learning evolutionary model. Since we are predicting a numerical variable, price, the distribution is set to 'gaussian'. At first, I tried the regular boosting model Gradient Boosting Machine(gbm) model. The result RMSE we got is 127.2261, which is lower than both linear regression and decision tree model. After submitting to Kaggle, the score is 87.59981. I am still confused why this happens when the RMSE is lower, the result score is higher. Then, I also tried the Boosting with cross-validation. However, since it takes too much time to run, I gave up. As a result, instead of boosting with cv, I tried the Boosting model with xgboost. XGBoost is an optimized distributed gradient boosting library designed to be highly efficient, flexible, and portable. The algorithm is also a bit picky about the format of variables used. All factor class variables need to be dummy coded and fed into the model as a matrix. After converting all the variables to numeric and running the xgboost model, the RMSE is 140.4818, which is lower than the linear regression, but is higher than the decision tree and other boosting model. However, after I submitted to Kaggle, the score is the lowest compared to others, which is 72.32. I still don't know the reason behind this situation. This is the best result so far, which it is the final result I submitted on Kaggle.

Furthermore, I also tried to run the Random Forest model, but at that time, I didn't have enough time to run and submit on Kaggle. Then, I tried this model after the submission deadline. Random Forest models consider a subset of predictors (default is p/3 for numerical in this case) for constructing each tree. We got 138.0336 RMSE in R, and 71.61806 score in Kaggle. In addition, we also tried to run Tuned Random Forest Model with ranger, but it takes too much time, so that I still have time to finish it.

4. Model Comparison

Model	RMSE on scoring data in R	Score on Kaggle	Notes
Model 1: Linear Regression	142.293	106.32416	
Model 2: Decision Tree	133.1909	86.37468	
Model 3: gbm model	127.2261	87.59981	
Model 4: xgboost model	140.4818	72.32	Final Kaggle result RMSE
Model 5: Random Forest	138.0336	71.61806	The lowest score

5. Discussion

After all the analysis so far, I think the process of selecting and filtering variables is the most important step in the whole project. Since I selected the variables by personal perspectives, the result model may not be the best. Maybe I delete the significant variables and include the unnecessary data in the model.

Furthermore, I found that for scoringData dataset, we should do the same cleaning process. In addition, there are still some missing values for binary variables host_is_superhost, host_has_profile_pic, host_identity_verified. So, I impute the missing value with the value, which is more frequent occurred, respectively 0, 1, 0.

```r
```{r}
str(data1)
data1 %>% select_if(~any(is.na(.)))
data1$host_is_superhost[is.na(data1$host_is_superhost)] <- "0"
data1$host_has_profile_pic[is.na(data1$host_has_profile_pic)] <- "1"
data1$host_identity_verified[is.na(data1$host_identity_verified)] <- "0"
data1 %>% select_if(~any(is.na(.)))
```
```

6. Future investigation and improvement

For future improvement, I would spend more time on feature selections since it is the most important step in the analysis. Then, I will try more models to test which the model is the best. Moreover, I will try to figure out why a higher RMSE in R get a lower score in Kaggle.