

# New-York-Aribnb-Price-Analysis.R

kadie

2020-05-21

```
library(dummies)
```

```
## dummies-1.5.6 provided by Decision Patterns
```

```
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 3.6.2
```

```
## -- Attaching packages ----- tidyverse 1.3.0 --
```

```
## <U+2713> ggplot2 3.2.1      <U+2713> purrr  0.3.3
## <U+2713> tibble  2.1.3      <U+2713> dplyr  0.8.4
## <U+2713> tidyr   1.0.0      <U+2713> stringr 1.4.0
## <U+2713> readr   1.3.1      <U+2713> forcats 0.4.0
```

```
## Warning: package 'ggplot2' was built under R version 3.6.2
```

```
## Warning: package 'dplyr' was built under R version 3.6.2
```

```
## -- Conflicts ----- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()     masks stats::lag()
```

```
library(infotheo)
```

```
library(gbm)
```

```
## Warning: package 'gbm' was built under R version 3.6.2
```

```
## Loaded gbm 2.1.5
```

```
library(plotly)
```

```
## Warning: package 'plotly' was built under R version 3.6.2
```

```
##
## Attaching package: 'plotly'
```

```
## The following object is masked from 'package:ggplot2':
##
##   last_plot
```

```
## The following object is masked from 'package:stats':
##
##   filter
```

```
## The following object is masked from 'package:graphics':
##
##   layout
```

```
library(FNN)
```

```
##
## Attaching package: 'FNN'
```

```
## The following object is masked from 'package:infotheo':
##
## entropy
```

```
library(ggplot2)

##### import dataset #####
df <- read.csv("AB_NYC_2019.csv")
df$last_review <- as.character(df$last_review)
df$last_review <- as.Date(df$last_review) #format of date should be mm-dd-yyyy
df$last_review <- difftime(as.POSIXct(Sys.Date()),
                          as.POSIXct(df$last_review),
                          units="days")
df$last_review <- as.numeric(df$last_review)
names(df)[13] <- "days_since_last_review"

## check which columns have missing values
for (i in seq(ncol(df))) {
  print(sum(is.na(df[,i])))
}
```

```
## [1] 0
## [1] 0
## [1] 0
## [1] 0
## [1] 0
## [1] 0
## [1] 0
## [1] 0
## [1] 0
## [1] 0
## [1] 0
## [1] 0
## [1] 10052
## [1] 10052
## [1] 0
## [1] 0
```

```
colnames(df[c(13,14)]) # Check which features have NA values
```

```
## [1] "days_since_last_review" "reviews_per_month"
```

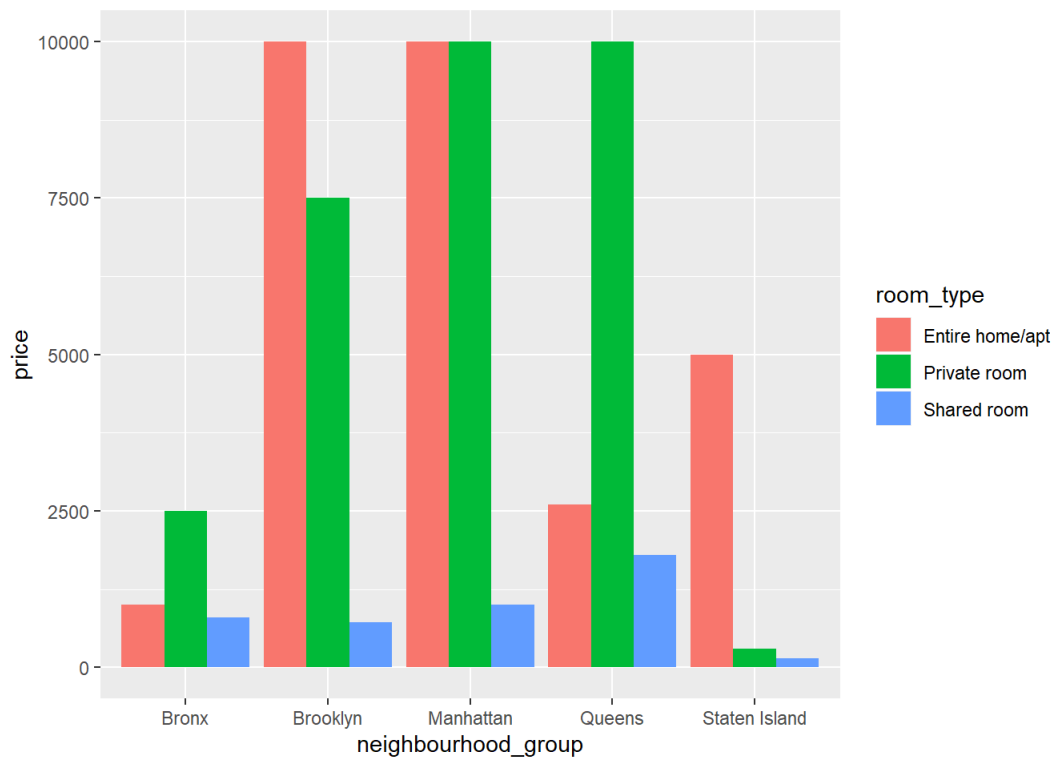
```
df[is.na(df)] <- 0 # Assign 0 to all blank cells

##### Descriptive Stats #####
plot_ly(df, labels = count(df, vars = df$room_type)$vars,
        values = count(df, vars = df$room_type)$n, type = 'pie') %>%
  layout(title = 'Percentage of Each Room Type',
         xaxis = list(showgrid = FALSE, zeroline = FALSE, showticklabels = FALSE),
         yaxis = list(showgrid = FALSE, zeroline = FALSE, showticklabels = FALSE))
```

```
plot_ly(x= df$price, type = "histogram") %>%  
  layout(title = "Distribution of Prices",  
    xaxis = list(range = c(0, 800),title = "in USD"),  
    yaxis = list(title = "Frequency"))
```

```
boroughs <- as.character(unique(df$neighbourhood_group))  
  
plot_ly(x = boroughs,y = sapply(boroughs,function(x){  
  mean(df$price[df$neighbourhood_group == x])}),type = "bar") %>%  
  layout(title = "Average Prices by Borough")
```

```
ggplot(df, aes(x=neighbourhood_group, y=price)) +  
  geom_bar(aes(fill = room_type), position = "dodge", stat = "identity")
```



```
plot_ly(df, y = df$price[df$price<500],  
  color = df$room_type[df$price<500], type = "box")
```

```
plot_ly(df, y = df$price[df$price<500],
        color = df$neighbourhood_group[df$price<500], type = "box")
```

```
##### Cleaning #####
```

```
# MinMax Scaling
scaler <- function(x){
  return((x-min(x))/(max(x)-min(x)))
}

colnames(df) # check column names
```

```
## [1] "id" "name"
## [3] "host_id" "host_name"
## [5] "neighbourhood_group" "neighbourhood"
## [7] "latitude" "longitude"
## [9] "room_type" "price"
## [11] "minimum_nights" "number_of_reviews"
## [13] "days_since_last_review" "reviews_per_month"
## [15] "calculated_host_listings_count" "availability_365"
```

```
for (i in c(7,8,11,12,13,14,15,16)){
  df[,i] <- scaler(df[,i])
}

# Create dummy variables
neighborhood_dummy <- dummy(df$neighbourhood, sep = "_", verbose = T)
```

```
## Warning in model.matrix.default(~x - 1, model.frame(~x - 1), contrasts = FALSE):
## non-list contrasts argument ignored
```

```
## C:/Users/kadie/Desktop/Kadie's File/JHU/FALL 2/Data Analytics/Group/New York Aribnb Price Analysis.R :
## 221 dummy variables created
```

```
room_dummy <- dummy(df$room_type, sep = "_", verbose = T)
```

```
## Warning in model.matrix.default(~x - 1, model.frame(~x - 1), contrasts = FALSE):
## non-list contrasts argument ignored
```

```
## C:/Users/kadie/Desktop/Kadie's File/JHU/FALL 2/Data Analytics/Group/New York Aribnb Price Analysis.R :  
3 dummy variables created
```

```
df <- cbind(df,neighborhood_dummy,room_dummy) # combine the dataframes  
  
# Delete Outliers  
a <- sort(df$price)[round(nrow(df)*0.01, 0)]  
b <- sort(df$price)[round(nrow(df)*0.99, 0)]  
df <- subset(df, price>a & price<b)  
nrow(df) # Check current row numbers
```

```
## [1] 47744
```

```
##### Text Analytics #####  
df_text <- data.frame(name=as.character(df$name),stringsAsFactors = FALSE)  
df_text$length <- nchar(df_text$name)  
summary(df_text$length)
```

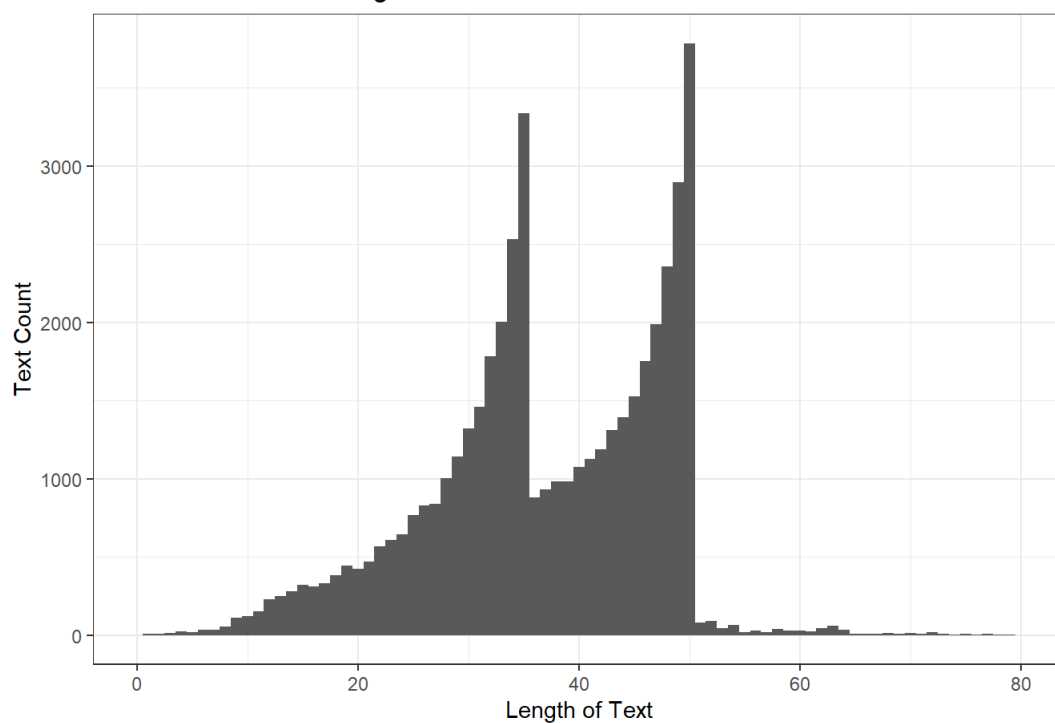
```
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.  
##      0.00   31.00   37.00   37.08   46.00   179.00
```

```
ggplot(df_text, aes(x=length)) +  
  theme_bw()+  
  geom_histogram(binwidth = 1)+  
  labs(y = "Text Count", x = "Length of Text",  
       title = "Distribution of Text Length")+  
  xlim(0,80)
```

```
## Warning: Removed 26 rows containing non-finite values (stat_bin).
```

```
## Warning: Removed 2 rows containing missing values (geom_bar).
```

Distribution of Text Length



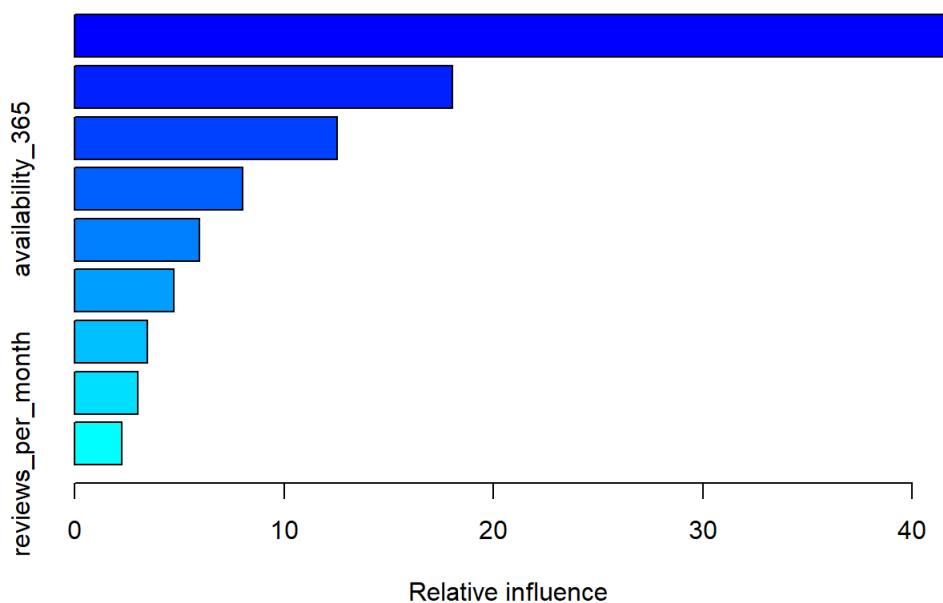
```
##### KNN Regression #####
set.seed(123)
train_index <- sample(1:nrow(df),0.8*nrow(df))
x_train <- df[train_index,11:ncol(df)] # Only columns 11-240 are used
x_test <- df[-train_index,11:ncol(df)] # Only columns 11-240 are used
y_train <- df[train_index,10]
y_test <- df[-train_index,10]
for (i in c(10,25,50,100)){
  knn_pred <- knn.reg(train=x_train, y=y_train, test=x_test, k=i)
  print(sqrt(mean((knn_pred$pred - y_test)^2)))
}
```

```
## [1] 79.55545
## [1] 78.94545
## [1] 79.55915
## [1] 80.32846
```

```
##### Gradient Boosting Machine #####
set.seed(123)
gbm_df <- df[,7:16]
x_train_gbm <- sample(1:nrow(gbm_df),0.8*nrow(gbm_df))
gbm <- gbm(price~., data = gbm_df[x_train_gbm,],distribution = "gaussian",
  n.trees = 150,shrinkage = 0.1, interaction.depth = 17)
print(gbm)
```

```
## gbm(formula = price ~ ., distribution = "gaussian", data = gbm_df[x_train_gbm,
##      ], n.trees = 150, interaction.depth = 17, shrinkage = 0.1)
## A gradient boosted model with gaussian loss function.
## 150 iterations were performed.
## There were 9 predictors of which 9 had non-zero influence.
```

```
summary(gbm)
```



```
##                                var    rel.inf
## room_type                     room_type 41.848913
## longitude                     longitude 18.057069
## latitude                      latitude 12.540113
## availability_365              availability_365 8.035680
## minimum_nights                minimum_nights 5.974985
## calculated_host_listings_count calculated_host_listings_count 4.759926
## number_of_reviews             number_of_reviews 3.468975
## days_since_last_review        days_since_last_review 3.040707
## reviews_per_month            reviews_per_month 2.273631
```

```
predicprice <- predict(gbm, newdata = gbm_df[-x_train_gbm,], n.trees = 150)
sqrt(mean((predicprice-gbm_df[-x_train_gbm,]$price)^2))
```

```
## [1] 74.07903
```