DATA-606 Airbnb Project

Gaurav, Ryan, Shora

02/02/2020

Introduction

Airbnb is an accommodation sharing community for both host and traveler. It is an online platform where hosts can list and give access to their properties or free space, and travelers can rent any listing of their choice. The information like price, location, and picture is already available on the platform but if required hosts and travelers can contact each other using the Airbnb platform and ask for more detail. This flexibility of fast information sharing makes Airbnb one of the most popular platform for affordable and short-stay accommodation. The primary objective of this project is to explore the available Airbnb listing in Vancouver based on price, minimum night, room type and location. We will also use various sampling and modeling techniques to determine if a model can be created to predict the nightly price of a stay based on the available information.

Dataset

The dataset is sourced from an independent, non-commercial, entity named Inside Airbnb. Inside Airbnb collects data about Airbnb listings from various cities and releases it so people can examine how Airbnb is being used around the world. For this report, the data for Vancouver, Canada, was used. The dataset includes data on the owner of the property and about the property itself. Each entry in the dataset is a listing. Each listing includes data on the owner of the listing and data about the listing itself, including, the nightly charge, the type of property, the neighbourhood where the listing is located, etc.

The variables of interest include, price, minimum_nights, room_type, property_type, instant_bookable, and neighbourhood. Price is the nightly charge for staying in the listing. Minimum_nights is the minimum number of nights a renter is required to book in order to stay in the listing. For example, a listing with a minimum nights of seven, means a renter must book to stay a minimum of one week. A longer stay can be booked, but not less than seven nights. Room_type describes the type of listing being provided, this includes four categories: entire home/apartment (apt), shared room, private room, and hotel room. Property_type describes the type of property where a renter will stay, this includes 21 categories, such as: house, apartment, townhouse, camper/RV, boat, etc. Instant_bookable is describes whether a listing can be booked without waiting for the owner to review and approve your application to rent the listing. Instant_bookable is a binary value that can be either “True” or “False.” Neighbourhood is the physical location in Vancouver where the listing is found, based on the boundaries defined by the city. There are 29 in the dataset.

Data Wrangling

1. Remove unused columns from the dataset which will make the data set liter and easy to process.

2. A regular expression is used to remove formatting information like “$” and “,” from the price column

3. Airbnb’s primary purpose is to provide affordable short stay so there is some data point which has an extreme value which goes against the company primary objective so we can remove this outlier.

4. Price more than 500 is quite rare and people paying such a high price will usually prefer a resort or hotel with brand name famous for their services so we can remove these values.

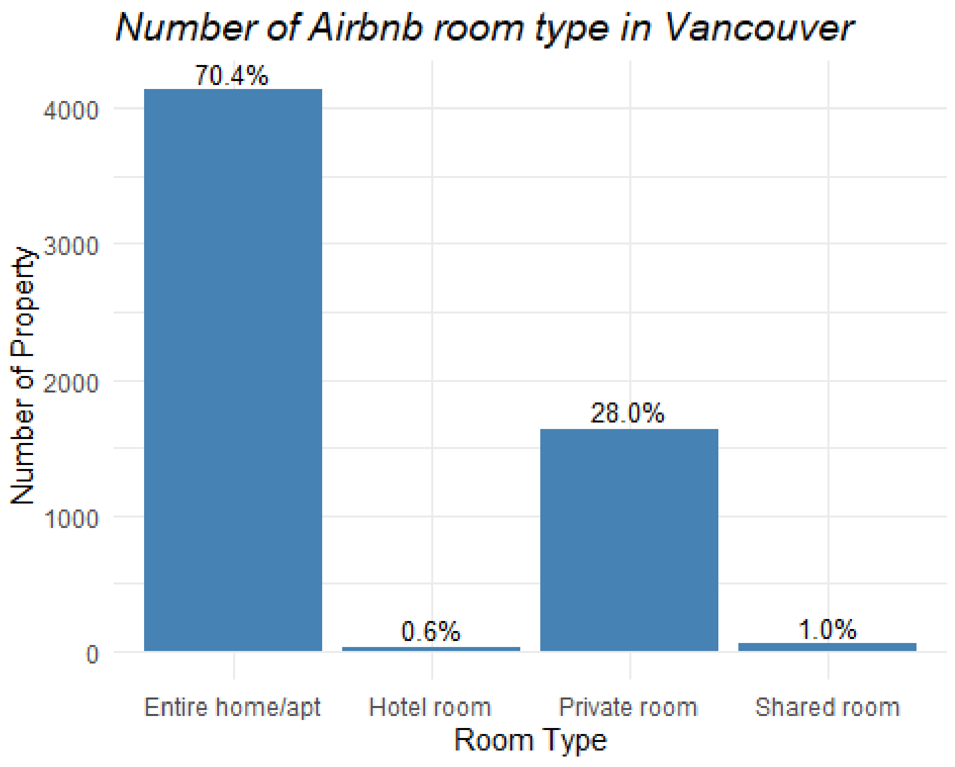
5. The minimum night is the minimum night of booking which means if a property has a minimum night of seven days one can not book it for six days or less. Airbnb is a short stay booking platform so having a minimum night booking of two months or more is quite rare and we can remove these values from the data set.

```
# Remove unnecessary dimension from the data set

selection_Array <-
c("id","name","host_name","host_response_rate","host_neighbourhood","host_lis
tings_count","street", "neighbourhood" ,"zipcode",
"latitude","longitude","property_type","room_type","price","minimum_nights","
availability_30","availability_60","availability_90","availability_365","numb
er_of_reviews","review_scores_value","requires_license","instant_bookable","c
ancellation_policy","reviews_per_month" )

airbnb_data <-  subset(airbnb_data, select = selection_Array )

airbnb_data['price'] <- gsub( ",", "", as.character(airbnb_data$price) )
airbnb_data['price'] <- as.numeric(gsub( "\\$", "",
as.character(airbnb_data$price) ,ignore.case=T))

#Remove extreme outlier

airbnb_data <- filter( airbnb_data, price <= 500)
airbnb_data <- filter( airbnb_data, minimum_nights <= 60)
```

Preliminary Analysis

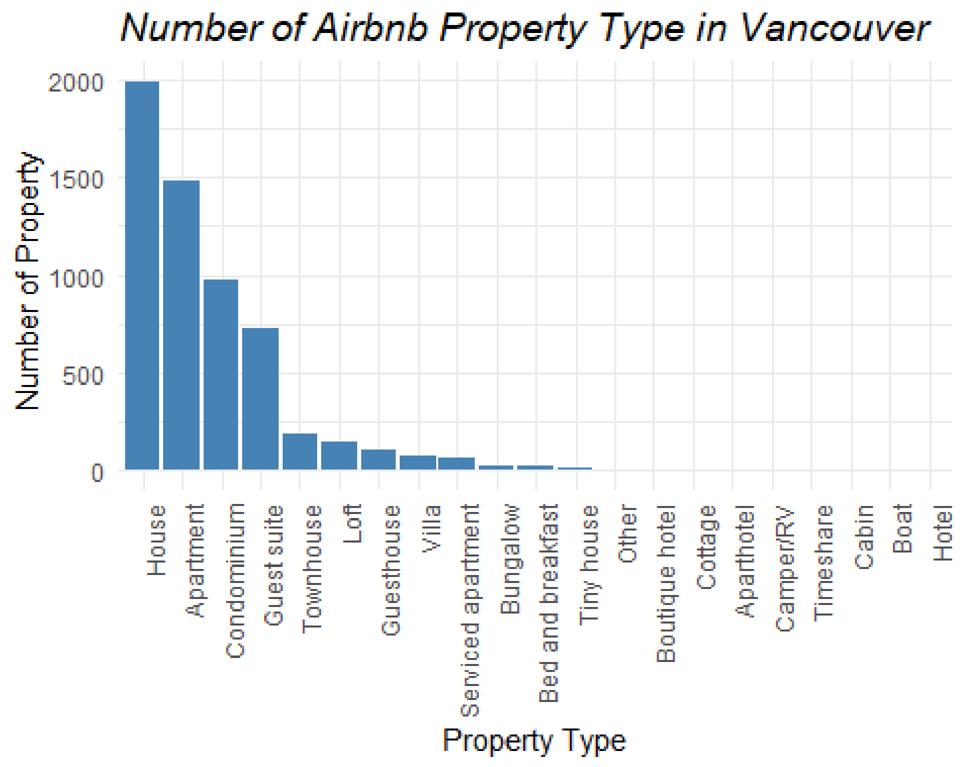
1. There are mainly four types of rooms available at Airbnb Vancouver that is entire home or apartment, hotel room, private room, and shared room. We can observe from the figure below that the entire home or apartment is the most famous room type in Vancouver with almost 70% of the population and all other room type is in rest of the 30%. The private room is 28% of the population while the hotel room and shared room is less than 1%. We can say that the entire home apartment and private room are the two most available room type in Vancouver.

```
data_gr_count_type <-  airbnb_data  %>%
  group_by(room_type) %>%
  count() %>%
  ungroup() %>%
 mutate(percent=n/sum(n)) %>%
  data.frame()

data_gr_count_type['label'] <- scales::percent(data_gr_count_type$percent)

ggplot(data = data_gr_count_type,  aes(x=room_type, y=n)) +
  geom_bar( stat="identity", fill="steelblue") +
  labs(title="Number of Airbnb room type in Vancouver", x="Room Type", y =
"Number of Property ",color="steelblue")+
  geom_text(aes( label =label), vjust=-0.3, size=3.5)+
  theme_minimal() +
  theme(plot.title = element_text( size=14, face="italic"))
```



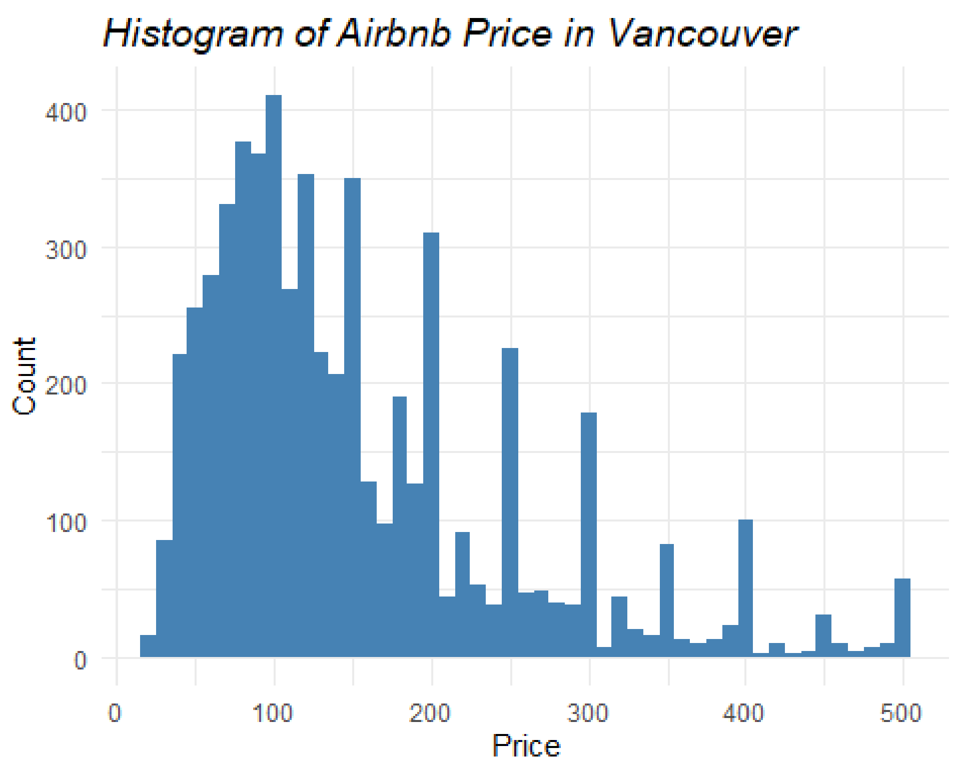
2. There is a total of twenty-one types of property available in Vancouver. House, apartment, condominium, and guest suite contains most of the population which means it is easy to find these types of property, while townhouse, loft, guest house, villa and service apartment are also available but in less quantity. There are twelve more types of property available but in very less number and it will be hard to find such property at the desired location.

```r
data_gr_count_proptype <-  airbnb_data  %>%
  group_by(property_type) %>%
  count() %>%
  ungroup() %>%
  mutate(percent=n/sum(n)) %>%
  data.frame()

data_gr_count_proptype['label'] <-
scales::percent(data_gr_count_proptype$percent)
data_gr_count_proptype <- data_gr_count_proptype[order(-
data_gr_count_proptype$n),]

ggplot(data = data_gr_count_proptype,  aes(x=reorder(property_type,-n), y=n))
+
  geom_bar( stat="identity", fill="steelblue") +
  labs(title="Number of Airbnb Property Type in Vancouver", x="Property
Type", y = "Number of Property ",
  color="steelblue")+
  theme_minimal() +
  theme(plot.title = element_text( size=14, face="italic"),
        axis.text.x = element_text(angle = 90, hjust = 1))
```



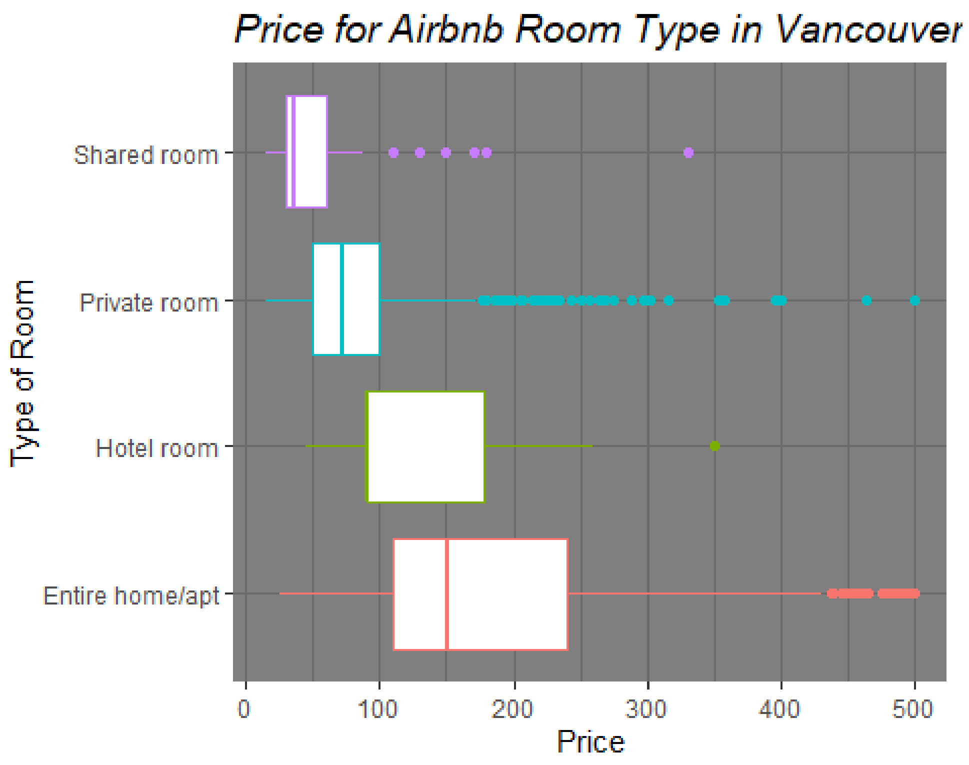
3. Looking at the histogram of the price we can observe that one can get an Airbnb accommodation for as low as $20 per night to $500 but most of the price is between $45 to $150. We can also observe an important trend from this histogram that most properties have a price in a round figure with a $50 increase like $100, $150, $200, $250, etc.

```
ggplot(data = airbnb_data, aes(x =price)) +
 geom_histogram( binwidth=10, na.rm=TRUE, fill="steelblue") +
  theme_minimal() +
  xlab('Price')+
  ylab('Count')+
 ggtitle("Histogram of Airbnb Price in Vancouver")+
  theme(plot.title = element_text(color="black", size=14, face="italic"))
```



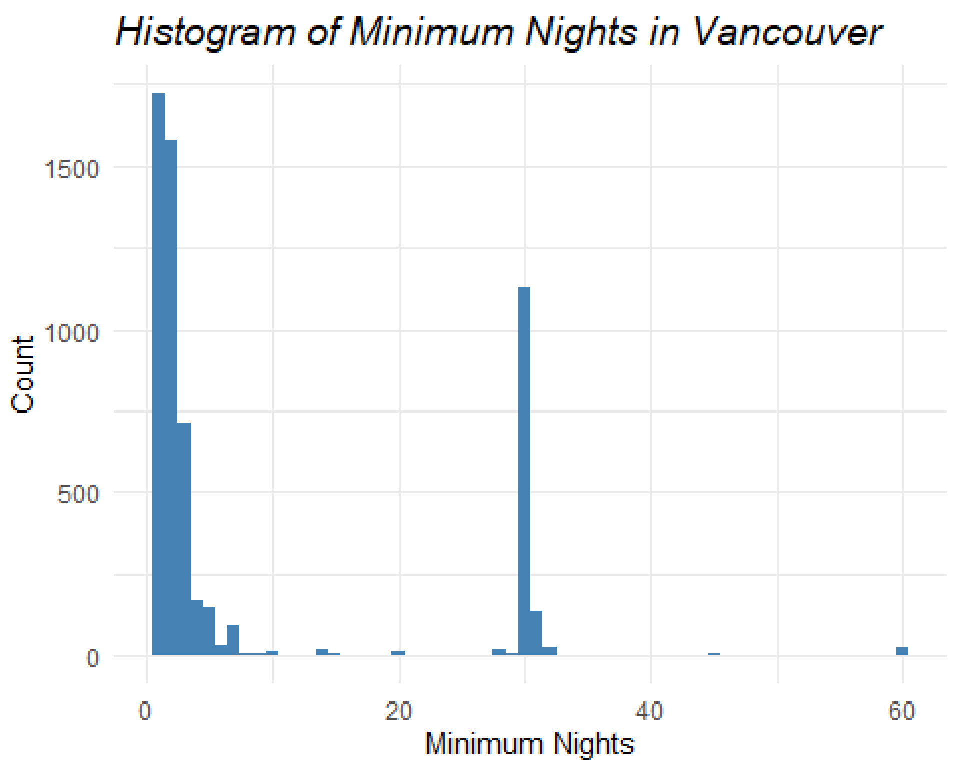
4. From the boxplot of price and room type below we can observe that the overall price is in decreasing order for an entire home apartment, hotel room, private room, and shared room respectively. We can also note a piece of important information that there is always an overlap in the price range for all room types which means one can get an entire home apartment for the same price that is offered for some hotel room and the same holds for private room and shared room.

```
ggplot(data = airbnb_data, aes(x = room_type, y=price, color = room_type)) +
  geom_boxplot( na.rm=TRUE,show.legend = FALSE)+
  xlab("Type of Room") +
  ylab('Price')+
  ggtitle("Price for Airbnb Room Type in Vancouver")+
  coord_flip()+
  theme_dark() +
  theme(plot.title = element_text( size=14, face="italic"),
        legend.title = element_blank())
```



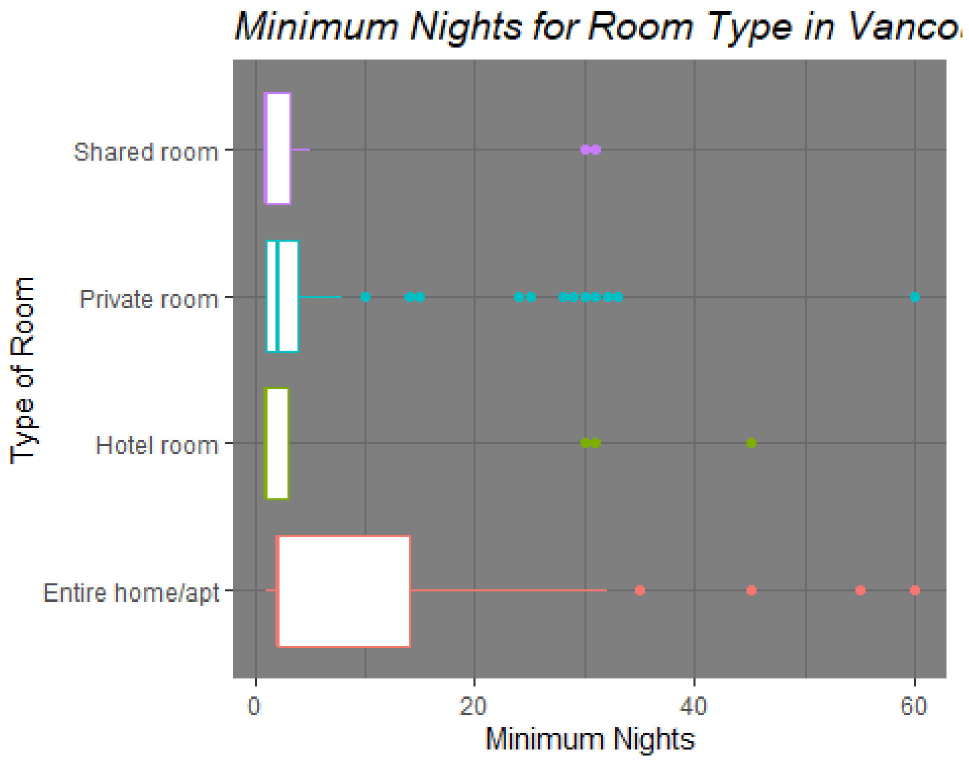
5. The minimum night is the minimum night stay required for booking the property. We can observe from the below histogram of the minimum night that most of the property has a minimum night booking from one day to a week but some property requires minimum booking for a month.

```
ggplot(data = airbnb_data, aes(x =minimum_nights)) +
   geom_histogram( binwidth=1, na.rm=TRUE, fill="steelblue") +
   xlab('Minimum Nights')+
   ylab('Count')+
   ggtitle("Histogram of Minimum Nights in Vancouver")+
  theme_minimal()+
  theme(plot.title = element_text( size=14, face="italic"))
```



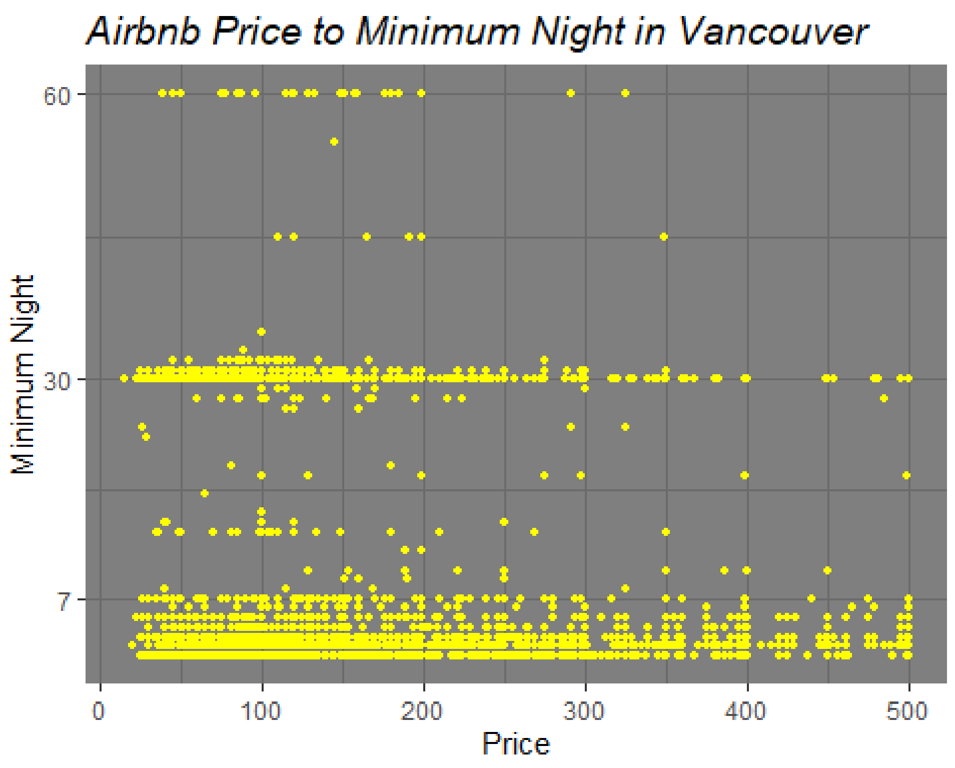
6. We can observe from the boxplot of the minimum night for the different room type that the average minimum night of booking required for a hotel room and the shared room is one night while for an entire home apartment and a private room is two nights.

```
ggplot(data = airbnb_data, aes(x = room_type, y=minimum_nights, color =
room_type)) +
  geom_boxplot( na.rm=TRUE, show.legend = FALSE)+
  xlab("Type of Room") +
  ylab('Minimum Nights')+
  ggtitle("Minimum Nights for Room Type in Vancouver")+
  coord_flip()+
  theme_dark() +
  theme(plot.title = element_text( size=14, face="italic"),
        legend.title = element_blank())
```



7. From the scatterplot of price and minimum night, we can see that as the price increases the minimum night required for the booking is decreasing. We have a lot more property with a minimum night booking of one month for a price of less than $150 as compared to a price between $150 to $250. This trend follows to price range from $250 to $350 and $350 to $450.

```
ggplot(data=airbnb_data, aes(x = price, y = minimum_nights)) +
  geom_point(col="#ffff00", size=1, na.rm = TRUE) +
  xlab("Price") +
  ylab("Minimum Night") +
  ggtitle("Airbnb Price to Minimum Night in Vancouver") +
  scale_y_continuous(breaks=c(7, 30,60))+
  theme_dark()+
  theme(plot.title = element_text( size=14, face="italic"))
```



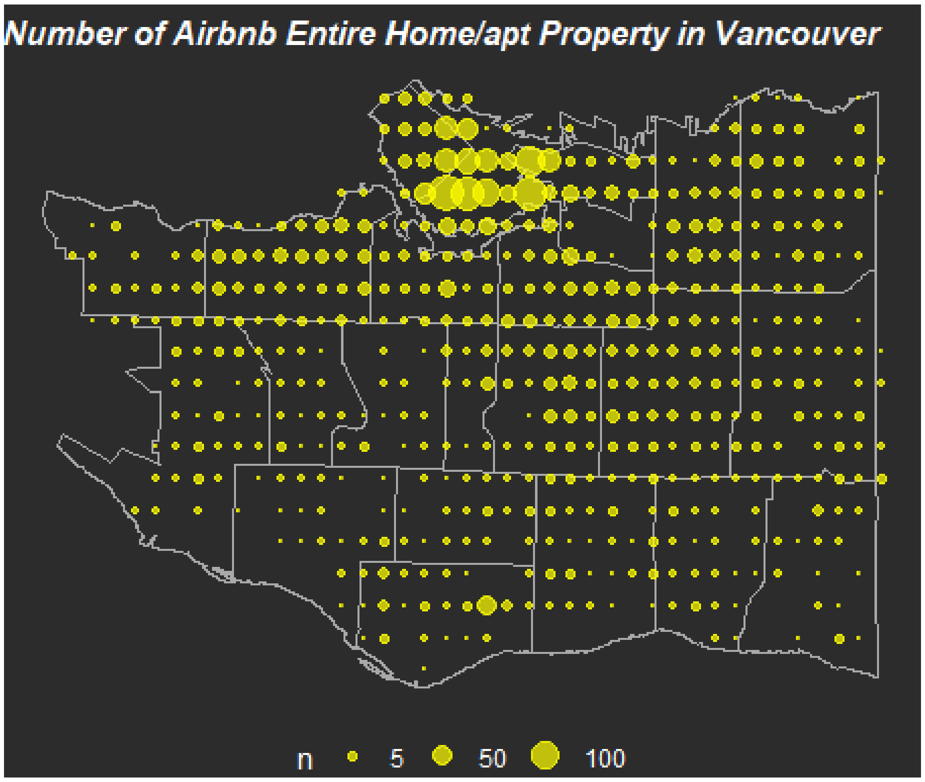
```r
# Create grid with 0.005 latitude and longitude resolution
grid <- function(xy, starting = c(0,0), gridsize = c(0.005,0.005)) {
  t(apply(xy, 1, function(z) gridsize/2+starting+gridsize*(floor((z -
starting)/gridsize))))
}
centroids <- grid(cbind(airbnb_data$latitude, airbnb_data$longitude))
airbnb_data$long <- centroids[, 2]
airbnb_data$lat <- centroids[, 1]
airbnb_data$grid <- paste(airbnb_data$long, airbnb_data$lat)
```

8. We know that the entire home or apartment consists of 70% of the total property in Vancouver so we want to look at its geographic distribution. To observe this distribution without overplotting we created a grid of 0.005 degrees and combined all the available property based on type in that grid area. We can observe from the scatterplot of the entire home or apartment on Vancouver map that most of the property is available in Vancouver downtown and it decreases as we move away from the downtown core.

```r
airbnb_cnt_apt <- airbnb_data %>%
  filter(room_type == "Entire home/apt") %>%
  group_by(long, lat, grid) %>%
  count() %>%
  ungroup()

ggplot() +
  geom_polygon(data = vancouver_geoJson, aes( x = long, y = lat, group =
group), fill="#2c2c2d", color="darkgray") +
  coord_map(projection = "mercator")+
  geom_point(data =airbnb_cnt_apt ,aes(x = long, y = lat, size = n), alpha =
0.7,  colour = '#ffff00')+
  scale_size(range = c(0, 20))+
  scale_size_area(max_size = 4, guide = FALSE)+ scale_radius()+
  scale_size(range=c(0.5,6),breaks=c(5,50,100,200),labels=c(
"5","50","100","200"))+
  ggtitle("Number of Airbnb Entire Home/apt Property in Vancouver")+
   theme_void()+
  theme(
        plot.background = element_rect(fill = "#2c2c2d"),
        legend.position = "bottom",
        legend.direction = "horizontal",
        legend.justification = "center",
        legend.text = element_text(color = "white"),
        legend.title = element_text(color = "white"),
        plot.title = element_text(color="white", size=12,
face="bold.italic"))
```

```
## Regions defined for each Polygons
```

```
## Scale for 'size' is already present. Adding another scale for 'size',
## which will replace the existing scale.
## Scale for 'size' is already present. Adding another scale for 'size',
## which will replace the existing scale.
## Scale for 'size' is already present. Adding another scale for 'size',
## which will replace the existing scale.
```

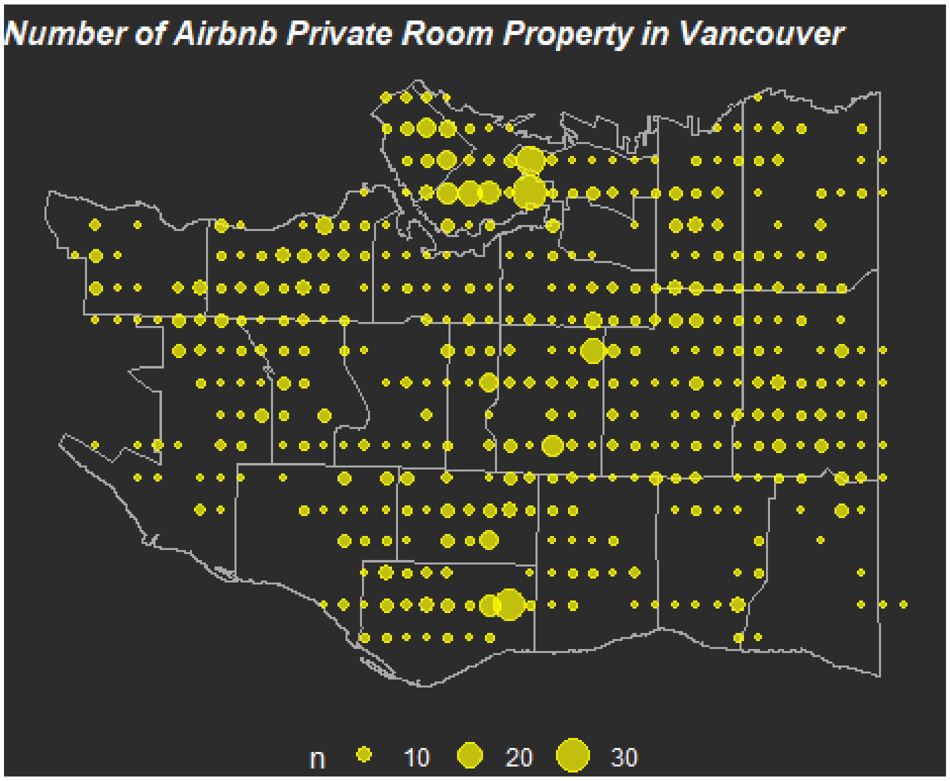


9. We know that the private room consists of 28% of the total property in Vancouver so we also want to look at its geographic distribution. To observe this distribution without overplotting we created a grid of 0.005 degrees and combined all the available property based on type in that grid are similar to what we did for the entire home or apartment room. We can observe a similar trend as seen for the entire home or apartment room additionally we also have a significant private room available in South Vancouver.

```r
airbnb_cnt_proom <- airbnb_data %>%
  filter(room_type == "Private room") %>%
  group_by(long, lat, grid) %>%
  count() %>%
  ungroup()

ggplot() +
  geom_polygon(data = vancouver_geoJson, aes( x = long, y = lat, group =
group), fill="#2c2c2d", color="darkgray") +
  coord_map(projection = "mercator")+
  geom_point(data =airbnb_cnt_proom ,aes(x = long, y = lat, size = n), alpha
= 0.7,  colour = '#ffff00')+
  ggtitle("Number of Airbnb Private Room Property in Vancouver")+
  scale_size_area(max_size = 4, guide = FALSE)+ scale_radius()+
  theme_void()+
  theme(
        plot.background = element_rect(fill = "#2c2c2d"),
        legend.position = "bottom",
        legend.direction = "horizontal",
        legend.justification = "center",
        legend.text = element_text(color = "white"),
        legend.title = element_text(color = "white"),
        plot.title = element_text(color="white", size=12,
face="bold.italic"))
```

```
## Regions defined for each Polygons
```

```
## Scale for 'size' is already present. Adding another scale for 'size',
## which will replace the existing scale.
```



Sampling techniques

```
van_list_reduced_reviewed = airbnb_data

mean (van_list_reduced_reviewed$price)

## [1] 154.5084

sd (van_list_reduced_reviewed$price)

## [1] 98.92144
```

With the cleaned dataset, the population mean is $(153.61 +/- 97.84) per night.

Applying sampling techniques

Three sampling techniques were applied to the dataset:

- **Simple Random Sampling (SRS)**
- **Stratafied Sampling**
- **Cluster Sampling**

For SRS and stratafied sampling, a 30 % sample was taken from the main dataset. For cluster sampling, 10 clusters were sampled from the available 29 in the dataset.

Simple Random Sampling

SRS was applied by sampling a 30 % sample from the population dataset. The population dataset contained 5869 entries after cleaning, so, the sample dataset included 1761 entries.

```
set.seed (1032)

N = round ( length (van_list_reduced_reviewed$price) * 0.30)   #   A sample
of 30% of the dataset

van_sample = sample ( van_list_reduced_reviewed$id, N, replace = FALSE)
ind = which ( van_list_reduced_reviewed$id %in% van_sample)

price_sample = van_list_reduced_reviewed[ind, 14]

#length (price_sample)

mean (price_sample)

## [1] 155.0909

sd (price_sample)

## [1] 95.87882
```

After the application of one SRS, the sample mean was determined to be $(155.09 +/- 95.88). This value is close to the value of the population mean, however, the result must be verified to ensure it is not the result of sampling.

To test whether the result of SRS is reproducible, we repeat the SRS process and determine the mean and standard deviation of the all the sample means. To verify the reproducability of the result from SRS, 50 random samples of 30 % of the population dataset were taken, and the mean of each sample was calculated. The mean and standard deviation of the 50 sample means was then computed.

```
N = length (van_list_reduced_reviewed$price) * 0.30

van_esti_mean = rep (0, 50)

for (i in 1:50){
  van_sample = sample ( van_list_reduced_reviewed$id, N, replace = FALSE)
  ind = which ( van_list_reduced_reviewed$id %in% van_sample)

  price_sample = van_list_reduced_reviewed[ind, 14]

  van_esti_mean[i] = mean (price_sample)
}

mean (van_esti_mean)   #   mean of sample means

## [1] 154.4301

sd (van_esti_mean)

## [1] 1.721769
```

The mean of the 50 sample means was found to be $(154.43 +/- 1.72). This result is approximately 0.08 different from the population mean. The sampling with repetition also produced a small standard deviation of 1.72. This suggests that SRS managed to generate consistence values for the sample mean. The population mean is also within one standard deviation of the mean of SRS means. This suggests that the SRS technique was able to consistently generate a sample mean close to the population mean, therefore, SRS is an appropriate technique to apply to this dataset.

Stratafied Sampling

For stratafied sampling room_type was examined to determine if it would be an appropriate variable to use for the stratification of the dataset. Because room_type is a categorical variable, an individual t-test and an ANOVA test were applied to a linear model including the response variable price and the descriptive variable room_type.

```
room_model = lm ( price ~ room_type, data = van_list_reduced_reviewed)

summary (room_model)
```

```
##
## Call:
## lm(formula = price ~ room_type, data = van_list_reduced_reviewed)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -159.66  -55.66  -17.56   32.34  417.44
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)            184.663      1.356 136.188  < 2e-16 ***
## room_typeHotel room    -47.548     14.797  -3.213  0.00132 **
## room_typePrivate room -102.103      2.543 -40.143  < 2e-16 ***
## room_typeShared room  -129.329     11.335 -11.410  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 87.17 on 5865 degrees of freedom
## Multiple R-squared:  0.2238, Adjusted R-squared:  0.2235
## F-statistic: 563.8 on 3 and 5865 DF,  p-value: < 2.2e-16
```

The variable room_type included four levels, Entire home/apt, Hotel room, Private room, and Shared room. The individual t-tests indicated that all the levels of room_type were significant (all p-values < 0.05). The results of the ANOVA test, F = 563.8, df = 3 and 5865 (p-value < 0.05), also show the linear model between price and room_type was significant. This indicates a correlation between price and room_type and price which justifies the use of room_type for stratifying the population dataset for sampling.

For stratified sampling, a 30 % sample of the population was taken. For room_type, this involved taking a 30 % sample from the four each of the four strata.

```
strat = table (van_list_reduced_reviewed$room_type)
strat

##
## Entire home/apt      Hotel room    Private room     Shared room
##            4133              35            1641              60

N = sort ( round (strat * 0.3), decreasing = TRUE)
N

##
## Entire home/apt    Private room     Shared room      Hotel room
##            1240             492              18              10
```

Sampling from the four strata included 1240 Entire home/apt entries, 492 Private room, 18 Shared room, and 10 Hotel room, for a total sample of 1760 entries. Taking a 30 % sample from each strata maintains the proportion each strata represents in the population data.

```
set.seed (1032)

van_strat_ind = sampling:::strata ( van_list_reduced_reviewed, stratanames =
c('room_type'), N, method = "srswor")

van_strat_data = getdata ( van_list_reduced_reviewed, van_strat_ind)
head (van_strat_data)

##       id                                                  name
## 1  10080                            D1 -  Million Dollar View 2 BR
## 13 19527              Art Deco Apartment in Fairview/South Granville
## 22 33150                        â\200œSuite on Mainâ\200\235  near  Q.E
Park
## 24 42645                                 The Charlie by COMFYSUITES
## 25 68894          Top-of-the-Line House: 5 min walk to Commerical Dr
## 26 70453 Open-concept, Modern Heritage Home with over 20 Skylights
##      host_name host_response_rate host_neighbourhood host_listings_count
## 1         Rami                81%       Coal Harbour                  39
## 13       Bevin               100%          Kitsilano                   1
## 22         Ili               100%         Riley Park                   0
## 24      Alicia               100% Downtown Vancouver                   3
## 25       Trish               100% Grandview-Woodland                   2
## 26 Samantha Jo               100%  Downtown Eastside                   1
##                   street      neighbourhood zipcode latitude longitude
## 1  Vancouver, BC, Canada       Coal Harbour V6E 2P4 49.28772 -123.1211
## 13 Vancouver, BC, Canada           Fairview V6H 1N7 49.26117 -123.1354
## 22 Vancouver, BC, Canada         Riley Park V5V 3L1 49.24686 -123.1047
## 24 Vancouver, BC, Canada Downtown Vancouver V6Z 1P6 49.28001 -123.1248
## 25 Vancouver, BC, Canada Grandview-Woodland V5N 4T4 49.26705 -123.0608
## 26 Vancouver, BC, Canada  Downtown Eastside V6A 2A4 49.27947 -123.0875
##    property_type price minimum_nights availability_30 availability_60
## 1    Condominium   151             60               0               6
## 13     Apartment   168             28              26              56
## 22     Apartment   100              3               0               0
## 24   Condominium   158             30              29              36
## 25         House   112              3               5              30
## 26         House   300              3               4              10
##    availability_90 availability_365 number_of_reviews review_scores_value
## 1               36              311                16                   9
## 13              86               86                35                   9
## 22               7              254                44                  10
## 24              49              229                28                   9
## 25              32              277               104                  10
## 26              10               10                19                  10
##    requires_license instant_bookable         cancellation_policy
## 1                 t                t strict_14_with_grace_period
## 13                t                f                    flexible
## 22                t                f strict_14_with_grace_period
## 24                t                t strict_14_with_grace_period
## 25                t                f strict_14_with_grace_period
## 26                t                t                    moderate
##    reviews_per_month      long     lat              grid       room_type
## 1               0.16 -123.1225 49.2875 -123.1225 49.2875 Entire home/apt
## 13              0.30 -123.1375 49.2625 -123.1375 49.2625 Entire home/apt
## 22              0.43 -123.1025 49.2475 -123.1025 49.2475 Entire home/apt
## 24              0.25 -123.1225 49.2825 -123.1225 49.2825 Entire home/apt
## 25              1.03 -123.0625 49.2675 -123.0625 49.2675 Entire home/apt
## 26              0.36 -123.0875 49.2775 -123.0875 49.2775 Entire home/apt
##    ID_unit      Prob Stratum
## 1        1 0.3000242       1
## 13      13 0.3000242       1
## 22      22 0.3000242       1
## 24      24 0.3000242       1
## 25      25 0.3000242       1
## 26      26 0.3000242       1
```

```
#dim (van_strat_data)

#   mean of one stratified sampling
mean (van_strat_data$price)
```

```
## [1] 154.1148
```

One sample using stratified sampling gave a sample mean of 154.11. This value is close to the value of the population mean, however, the result must be verified to ensure it is not the result of sampling.

Like SRS, to test the reproducability of the result from stratafied sampling, 50 replicates of the stratified sampling process were performed with 30 % of the population dataset, all with the same proportions of the four strata. From the samples the mean and standard deviation of the sample means was computed.

```
van_esti_mean = rep (0, 50)

for (i in 1:50){
  strat = sampling:::strata ( van_list_reduced_reviewed, stratanme =
c('room_type'), N, method = "srswor")
  mydata = getdata (van_list_reduced_reviewed, strat)

  van_esti_mean[i] = mean (mydata$price)
}

mean (van_esti_mean)   #   mean of sample means
```

```
## [1] 153.8726
```

```
sd (van_esti_mean)
```

```
## [1] 1.735651
```

The mean sample mean was determined to be $(153.87 +/- 1.74). This result was close to the population mean, 154.51. Stratafied sampling also resulted in a small standard deviation, 1.74, which indicates the reproducability of the sample mean with stratafied sampling. This suggests that the stratafied sampling technique was able to consistently generate a sample mean close to the population mean, therefore, stratafied sampling is an appropriate technique to apply to this dataset to generate a sample.

To examine the variance between strata (SSB) and within strata (SSW), an ANOVA test was performed.

```
summary ( aov ( price ~ room_type, data = van_list_reduced_reviewed))

##                Df   Sum Sq Mean Sq F value Pr(>F)    
## room_type       3 12853690 4284563   563.8 <2e-16 ***
## Residuals    5865 44567339    7599                   
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The results of the ANOVA were SSB = 12853690 and SSW = 44567339. The results suggest that the variance between the strata is large, which is consistent with the desired results of stratification each strata is as different as possible so that each strata is clearly defined. However, the variance within the strata is also large. This is not the desired result. The desired outcome is a small SSW, meaning each strata is as homogeneous as possible. This brings into question whether stratafied sampling is the most appropriate technique to apply to this dataset.

Cluster Sampling

For cluster sampling, neighbourhood was used to define the primary sampling units (psus) of the dataset. Neighbourhood was used because it appeared to be the most natural variable available to create clusters. In total there were 29 neighbourhoods, psus, in the dataset. To examine cluster sampling, 10 psus, of the available 29, were randomly sampled from the population.

```
table (van_list_reduced_reviewed$neighbourhood)

## 
##                              Arbutus Ridge 
##                                        103 
##                                    Burnaby 
##                                          8 
##                                  Chinatown 
##                                         36 
##                               Coal Harbour 
##                                         22 
##                           Commercial Drive 
##                                        100 
##                          Downtown Eastside 
##                                        111 
##                         Downtown Vancouver 
##                                      1515
##                         Dunbar-Southlands
##                                       176
##                                 Fairview
##                                       148
##                               Fraserview
##                                       206
##                            Frederiksberg
##                                         1
##                                  Gastown
##                                        35
##                        Grandview-Woodland
##                                       197
##                         Hastings-Sunrise
##                                       266
##                 Kensington-Cedar Cottage
##                                       378
##                               Kerrisdale
##                                        95
##                                Killarney
##                                        83
##                                Kitsilano
##                                       466
##                                  Marpole
##                                       261
##                           Mount Pleasant
##                                       321
##                                 Oakridge
##                                       148
## Point Grey/University of British Columbia
##                                       127
##                       Renfrew-Collingwood
##                                       265
##                              Riley Park
##                                       345
##                              Shaughnessy
##                                       102
##                             South Cambie
##                                        91
##                               Strathcona
##                                        27
##                                 West End
##                                       177
##                                 Yaletown
##                                        59
```

```r
set.seed (1032)
van_clusters = sampling:::cluster ( van_list_reduced_reviewed, clustername =
c("neighbourhood"), size = 10, method = "srswor")
van_clstd_data = getdata ( van_list_reduced_reviewed, van_clusters)
```

```r
table (van_clstd_data$neighbourhood)
```

```
## 
##                                 Arbutus Ridge 
##                                           103 
##                                       Burnaby 
##                                             0 
##                                     Chinatown 
##                                             0 
##                                  Coal Harbour 
##                                            22 
##                              Commercial Drive 
##                                             0 
##                            Downtown Eastside 
##                                           111 
##                           Downtown Vancouver 
##                                             0 
##                             Dunbar-Southlands 
##                                             0 
##                                      Fairview 
##                                             0 
##                                    Fraserview 
##                                           206 
##                                 Frederiksberg 
##                                             0 
##                                       Gastown 
##                                             0 
##                            Grandview-Woodland 
##                                             0 
##                              Hastings-Sunrise 
##                                             0 
##                     Kensington-Cedar Cottage 
##                                             0 
##                                    Kerrisdale 
##                                             0 
##                                     Killarney 
##                                            83 
##                                     Kitsilano 
##                                             0 
##                                       Marpole 
##                                           261 
##                                Mount Pleasant 
##                                           321 
##                                      Oakridge 
##                                           148 
## Point Grey/University of British Columbia 
##                                             0 
##                           Renfrew-Collingwood 
##                                             0 
##                                    Riley Park 
##                                           0
##                                 Shaughnessy
##                                           0
##                                South Cambie
##                                           0
##                                  Strathcona
##                                          27
##                                    West End
##                                         177
##                                    Yaletown
##                                           0
```

```
#   mean of one cluster sampling
mean (van_clstd_data$price)

## [1] 141.9342
```

The sample mean for one cluster sampling is 141.93.

To test the reproducability of the result from cluster sampling, 100 replicates of the cluster sampling process were performed with 10 psus randomly sampled from the population dataset. From the samples the mean and standard deviation of the sample means were computed.

```
van_esti_clstr_mean = rep (0, 100)

for (i in 1:100){
  van_clusters = sampling:::cluster ( van_list_reduced_reviewed, clustername
= c("neighbourhood"), size = 10, method = "srswor")
  mydata = getdata ( van_list_reduced_reviewed, van_clusters)

  van_esti_clstr_mean[i] = mean (mydata$price)
}

mean (van_esti_clstr_mean)   #   mean of sample means

## [1] 149.4225

sd (van_esti_clstr_mean)

## [1] 14.73208
```

The mean sample mean was determined to be $(149.42 +/- 14.73). This result is lower than the population mean, 154.51. Because the mean value after 100 replicates is lower, we can infer that the cluster sampling consistently produced a sample mean that was less than the population mean, meaning there is a bias towards smaller price values. The standard deviation for the repeated cluster sampling was also large, 14.73. The large standard deviation indicates that the computed sample means for each sample varies greatly. This would suggest cluster sampling may not be an appropriate sampling method for this dataset, as there is a large amount of variability in the results.

Another issue this dataset presents with cluster sampling, is the result of the varying number of properties, secondary sampling units, ssus, for each neighbourhood, psu, in the dataset. From an examination of the dataset, the number of ssus for each neighbourhood can varying greatly, for example: 1102 ssus are listed in the neighbourhood ‘Downtown’, but only 22 are listed in ‘Coal Harbour.’ Other neighbourhoods vary from containing several hundred ssus to having less than a hundred ssus. Because of the varying number of ssus per cluster, the final sample size can vary greatly depending on the psus sampled.

The varying number of ssus also make it difficult to perform a two-stage cluster sampling. Because of the difficulty, two-stage cluster sampling was not performed.

To examine the variation between strata (SSB) and within strata (SSW), an ANOVA test was performed.

```
summary ( aov ( price ~ neighbourhood, data = van_list_reduced_reviewed))

##                  Df   Sum Sq Mean Sq F value Pr(>F)
## neighbourhood   28  5089200  181757   20.28 <2e-16 ***
## Residuals     5840 52331829    8961
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

For cluster analysis, you want the variance within the the clusters to be large, so that each cluster is as heterogeneous as possible, to be representative of the population. You also want to the variance between clusters to be small, so that the clusters are as similar as possible. The results of the ANOVA were SSB = 5089200 and SSW = 52331829. This indicates a small variance between the clusters and a large variance in within the clusters. These results agree with the desired outcomes, which would indicate cluster sampling is an appropriate technique to apply to the dataset.

From the results of three sampling techniques, SRS and stratafied sampling were found to have the most consistent results and the results closest to the population mean. Cluster sampling was found to have a bias towards values less than the population mean. Examining the variances between, SSB, and within, SSW, strata and clusters, it was found that for cluster sampling, SSB was small and SSW was large, indicating the clusters were heterogeneous and they did not change much between clusters. For stratafied sampling, both SSB and SSW were found to be large, indicating there were differences between strata, but the strata were not homogeneous. This would suggest SRS may be the better sampling method to apply to this dataset. However, even with the question regarding the homogeneity of the strata, because stratafied sampling was shown to consistently provide a sample mean close to the population mean, it may also be an acceptable sampling technique for the dataset.

Categorical Analysis

For categorical analysis, the dependency of price on five descriptive variables was examined. Those five variables were:

- minimum_nights
- room_type
- property_type
- instant_bookable
- neighbourhood

The goal is to create a generalized linear model that will predict the price of a nightly stay, using the chosen variables. The independency of these variables with price were examined to determine which variables should be included in the model.

To perform categorical analysis, price and minimum_nights needed to be converted to categorical variables. Price was divided into five levels of a $100 range , so, all properties with a price less than 100 are one level, properties priced 100 to 199 are the second, and so on. The fifth level is all properties with prices equal to and above 400. For minimum_nights, the variable was divided into four levels. These levels were assigned arbitrary lengths of time. The first level was for properties where the required minimum number of nights was one night. The second level included all properties where the minimum number of nights was greater than one night and up to seven nights (one week). The third level was all properties where the minimum number of nights was between a week and a month (30 nights), and the fourth level was all properties where the minimum number of nights was greater than one month.

```
van_strat_data_dup = van_strat_data

#   Converting the variable price into a categorical variable with 5 levels

van_strat_data_dup$price_cat = 50   #   price [0,100)
van_strat_data_dup$price_cat[ van_strat_data_dup$price > 99 &
van_strat_data_dup$price < 200] = 150   #   price [100, 200)
van_strat_data_dup$price_cat[ van_strat_data_dup$price > 199 &
van_strat_data_dup$price < 300] = 250   #   price [200, 300)
van_strat_data_dup$price_cat[ van_strat_data_dup$price > 299 &
van_strat_data_dup$price < 400] = 350   #   price [300, 400)
van_strat_data_dup$price_cat[ van_strat_data_dup$price > 399] = 450   #
price [400, 500]

#   Converting the variable minimum_nights into a categorical variable with 4
levels

van_strat_data_dup$stay = 1   #   one night
van_strat_data_dup$stay[ van_strat_data_dup$minimum_nights > 1 &
van_strat_data_dup$minimum_nights < 8] = 2   #   one week stay
van_strat_data_dup$stay[ van_strat_data_dup$minimum_nights > 7 &
van_strat_data_dup$minimum_nights < 31] = 3   #   one month stay
van_strat_data_dup$stay[ van_strat_data_dup$minimum_nights > 30] = 4   #
long stay

head (van_strat_data_dup)

##       id                                                    name
## 1  10080                            D1 -  Million Dollar View 2 BR
## 13 19527            Art Deco Apartment in Fairview/South Granville
## 22 33150                     â\200œSuite on Mainâ\200\235  near  Q.E
Park
## 24 42645                                The Charlie by COMFYSUITES
## 25 68894         Top-of-the-Line House: 5 min walk to Commerical Dr
## 26 70453 Open-concept, Modern Heritage Home with over 20 Skylights
##      host_name host_response_rate host_neighbourhood host_listings_count
## 1         Rami                81%       Coal Harbour                  39
## 13       Bevin               100%          Kitsilano                   1
## 22        Ili              100%         Riley Park                    0
## 24     Alicia              100% Downtown Vancouver                    3
## 25      Trish              100% Grandview-Woodland                    2
## 26 Samantha Jo              100%  Downtown Eastside                    1
##                   street       neighbourhood zipcode latitude longitude
## 1  Vancouver, BC, Canada        Coal Harbour V6E 2P4 49.28772 -123.1211
## 13 Vancouver, BC, Canada            Fairview V6H 1N7 49.26117 -123.1354
## 22 Vancouver, BC, Canada          Riley Park V5V 3L1 49.24686 -123.1047
## 24 Vancouver, BC, Canada Downtown Vancouver V6Z 1P6 49.28001 -123.1248
## 25 Vancouver, BC, Canada Grandview-Woodland V5N 4T4 49.26705 -123.0608
## 26 Vancouver, BC, Canada  Downtown Eastside V6A 2A4 49.27947 -123.0875
##    property_type price minimum_nights availability_30 availability_60
## 1    Condominium   151             60               0               6
## 13     Apartment   168             28              26              56
## 22     Apartment   100              3               0               0
## 24   Condominium   158             30              29              36
## 25         House   112              3               5              30
## 26         House   300              3               4              10
##    availability_90 availability_365 number_of_reviews review_scores_value
## 1               36              311                16                   9
## 13              86               86                35                   9
## 22               7              254                44                  10
## 24              49              229                28                   9
## 25              32              277               104                  10
## 26              10               10                19                  10
##    requires_license instant_bookable         cancellation_policy
## 1                 t                t strict_14_with_grace_period
## 13                t                f                    flexible
## 22                t                f strict_14_with_grace_period
## 24                t                t strict_14_with_grace_period
## 25                t                f strict_14_with_grace_period
## 26                t                t                    moderate
##    reviews_per_month      long     lat              grid       room_type
## 1               0.16 -123.1225 49.2875 -123.1225 49.2875 Entire home/apt
## 13              0.30 -123.1375 49.2625 -123.1375 49.2625 Entire home/apt
## 22              0.43 -123.1025 49.2475 -123.1025 49.2475 Entire home/apt
## 24              0.25 -123.1225 49.2825 -123.1225 49.2825 Entire home/apt
## 25              1.03 -123.0625 49.2675 -123.0625 49.2675 Entire home/apt
## 26              0.36 -123.0875 49.2775 -123.0875 49.2775 Entire home/apt
##    ID_unit      Prob Stratum price_cat stay
## 1        1 0.3000242       1       150    4
## 13      13 0.3000242       1       150    3
## 22      22 0.3000242       1       150    2
## 24      24 0.3000242       1       150    3
## 25      25 0.3000242       1       150    2
## 26      26 0.3000242       1       350    2
```

Tests

```
# A self-defined function to calculate the Mantel-Haenszel statistic, as well
as the p-value
```

```r
pears.cor=function(table, rscore, cscore)
{
    dim=dim(table)
    rbar=sum(margin.table(table,1)*rscore)/sum(table)
    rdif=rscore-rbar
    cbar=sum(margin.table(table,2)*cscore)/sum(table)
    cdif=cscore-cbar
    ssr=sum(margin.table(table,1)*(rdif^2))
    ssc=sum(margin.table(table,2)*(cdif^2))
    ssrc=sum(t(table*rdif)*cdif)
    pcor=ssrc/(sqrt(ssr*ssc))
    pcor
    M2=(sum(table)-1)*pcor^2
    M2
    result=c(pcor, M2, (1-pchisq(M2,1)))
    result=as.table(result)
    names(result)=c('Pearson correlation','MH statistic', 'P-Value')
    result
}

#   ANOVA test to determine the independency between a nominal variable and
an ordinal variable.
#   The function converts a contingency table to a dataframe, then performs
an ANOVA test.

cat_anova = function (cat_tab) {

  sorted_table = as.data.frame (cat_tab)

  Var_x = vector ()
  Var_y = vector ()

  for (i in 1:length ( sorted_table[,3])) {
    Var_x = c( rep ( as.character (sorted_table[ i, 1]), sorted_table[ i,
3]), Var_x)
    Var_y = c( rep ( as.numeric ( levels (sorted_table[ i, 2]))[sorted_table[
i, 2]], sorted_table[ i, 3]), Var_y)
  }

  tab_van = data.frame ( Var_x, Var_y)

#  print ( paste ("The x variable is", names (sorted_table)[1]))
#  print ( paste ("The y variable is", names (sorted_table)[2]))

  summary ( aov ( Var_y ~ Var_x, data = tab_van))
}
```

The categorical analysis was performed using contingency tables. Five contingency tables were produced where price was compared with one of the five descriptive variables listed above. To examine the independence of price and the other variables, the Mantel-Haenszel test and the Pearson Chi-square test were used to examine the cases where both variables of the contingency table were ordinal. When a table contained a combination of ordinal and nominal variables, an ANOVA test was performed.

For the generalized linear model

Price and minimum nights

The contingency table comparing price and minimum nights is

```
van_table_minnights = table (van_strat_data_dup$stay,
van_strat_data_dup$price_cat)
names ( dimnames (van_table_minnights)) = c( 'Nights', 'Price')

van_table_minnights

##        Price
## Nights  50 150 250 350 450
##      1 228 188  61  27  14
##      2 244 363 127  65  34
##      3 158 154  26  14  10
##      4  21  22   3   1   0
```

Both variables are ordinal, so a Pearson's correlation test was performed,

```
pears.cor ( van_table_minnights, c(1,2,3,4), c(50,150,250,350,450))

## Pearson correlation        MH statistic             P-Value
##         -0.02631657          1.21821663          0.26971092
```

From the results of the test, r = -0.0263, MH = 1.22 (p-value = 0.27 > 0.05), we would infer that price and minimum_nights are independent, and that the minimum number of nights for a stay do not influence the final nightly price of staying at a certain property. However, a problem with the Mantel-Haenszel test was found. It was found that the result of the test could change depending on the score applied to the MH test.

```
pears.cor ( van_table_minnights, c(1,7,30,55), c(50,150,250,350,450))

## Pearson correlation        MH statistic             P-Value
##       -0.0810014818       11.5412212687        0.0006806991
```

For example, applying the Mantel-Haenszel test again with a different score for the required minimum nights resulted in a different result than from the previous test. From these new results, r = -0.0810, MH = 11.54 (p-value = 0.00068 < 0.05), we would infer that price and minimum_nights are dependent, and that the variable should be kept in the model.

To resolve this issue a Chi-square test was applied to the contingency table.

```
chisq.test (van_table_minnights)
```

```
## Warning in chisq.test(van_table_minnights): Chi-squared approximation may
## be incorrect

##
##  Pearson's Chi-squared test
##
## data:  van_table_minnights
## X-squared = 57.826, df = 12, p-value = 5.604e-08
```

From the result, Chi-square = 57.826, df = 12 (p-value = 0.000000056 < 0.05), we would infer that price and minimum_nights are dependent and should remain in the model.

```
chisq.residuals (van_table_minnights)

## Warning in stats::chisq.test(tab): Chi-squared approximation may be
## incorrect

##       Price
## Nights    50   150   250   350   450
##      1  2.63 -1.78 -0.36 -0.80 -0.74
##      2 -3.65  1.02  2.40  2.02  1.25
##      3  2.08  0.37 -2.79 -1.71 -0.56
##      4  0.87  0.59 -1.16 -1.10 -1.24
```

Examining the residuals, indicates that properties which cost less than $100 a night and require a minimum stay of one week deviated the most from the independence assumption.

Because of the issue with the Mantel-Haenszel test, we deferred to the result of the Chi-square test and concluded there was a dependency between price and minimum_nights and that minimum_nights should be included in the model.

The other four contingency tables were a mix of ordinal and nominal data. Therefore, an ANOVA test was used to determine whether there was independence between the variables.

A summary of the results is provided in the following table,

Variable	Test statistic (F, df)	p-value
room_type	164.9, df = 3, 1756	< 0.05
property_type	6.433, df = 16, 1743	0.000000000000029 < 0.05
instant_bookable	2.5, df = 1, 1758	0.114 > 0.05
neighbourhood	7.724, df = 27, 1732	< 0.05

Price and room type

```
van_table_room_type = table (van_strat_data_dup$room_type,
van_strat_data_dup$price_cat)
names ( dimnames (van_table_room_type)) = c( 'Room', 'Price')

van_table_room_type
```

```
##                     Price
## Room                  50 150 250 350 450
##   Entire home/apt 245 624 207 106  58
##   Hotel room          5   1   4   0   0
##   Private room      385 101   6   0   0
##   Shared room        16   1   0   1   0
```

```
cat_anova (van_table_room_type)
```

```
##                 Df   Sum Sq Mean Sq F value Pr(>F)    
## Var_x            3  3994044 1331348   164.9 <2e-16 ***
## Residuals     1756 14173933    8072                   
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Price and property type

```
van_table_prop_type = table (van_strat_data_dup$property_type,
van_strat_data_dup$price_cat)
names ( dimnames (van_table_prop_type)) = c( 'Property', 'Price')

van_table_prop_type
```

```
##                        Price
## Property                 50 150 250 350 450
##   Aparthotel              0   0   1   0   1
##   Apartment             131 209  53  25  18
##   Bed and breakfast       1   2   2   0   0
##   Boat                    0   0   0   0   0
##   Boutique hotel          1   1   0   0   0
##   Bungalow                9   1   1   0   0
##   Cabin                   0   0   0   0   0
##   Camper/RV               1   1   0   0   0
##   Condominium            65 140  59  28  12
##   Cottage                 1   1   0   0   0
##   Guest suite            90 108  18   5   0
##   Guesthouse             10  22   4   1   0
##   Hotel                   0   0   0   0   0
##   House                 302 178  57  35  21
##   Loft                    6  29   6   4   3
##   Other                   0   0   0   0   0
##   Serviced apartment      6   7   3   1   1
##   Timeshare               0   0   3   0   0
##   Tiny house              0   4   1   0   0
##   Townhouse              12  20   6   8   2
##   Villa                  16   4   3   0   0
```

```
cat_anova (van_table_prop_type)
```

```
##                 Df   Sum Sq Mean Sq F value   Pr(>F)    
## Var_x           16  1012991   63312   6.433 2.92e-14 ***
## Residuals     1743 17154986    9842                     
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Price and instant bookable

```
van_table_inst_book = table (van_strat_data_dup$instant_bookable,
van_strat_data_dup$price_cat)
names ( dimnames (van_table_inst_book)) = c( 'InstBook', 'Price')

van_table_inst_book

##          Price
## InstBook  50 150 250 350 450
##        f 335 379 105  46  27
##        t 316 348 112  61  31

cat_anova (van_table_inst_book)

##               Df   Sum Sq Mean Sq F value Pr(>F)
## Var_x          1    25795   25795     2.5  0.114
## Residuals   1758 18142182   10320
```

Price and neighbourhood

```
van_table_neighbour = table (van_strat_data_dup$neighbourhood,
van_strat_data_dup$price_cat)
names ( dimnames (van_table_neighbour)) = c( 'Neighbourhood', 'Price')

van_table_neighbour

##                                            Price
## Neighbourhood                                50 150 250 350 450
##   Arbutus Ridge                               9  11   7   1   4
##   Burnaby                                     1   0   0   0   0
##   Chinatown                                   3   6   3   1   0
##   Coal Harbour                                2   4   1   0   0
##   Commercial Drive                            9  10   3   3   2
##   Downtown Eastside                          13  21   3   4   1
##   Downtown Vancouver                         87 200  84  42  32
##   Dunbar-Southlands                          26  22   5   4   3
##   Fairview                                   21  19   3   0   0
##   Fraserview                                 34  22   8   1   2
##   Frederiksberg                               0   0   0   0   0
##   Gastown                                     1   4   2   2   0
##   Grandview-Woodland                         25  22   4   0   0
##   Hastings-Sunrise                           55  29   5   3   0
##   Kensington-Cedar Cottage                   57  46  14   4   3
##   Kerrisdale                                 18  10   2   1   0
##   Killarney                                  15  10   1   1   0
##   Kitsilano                                  38  54  17  10   2
##   Marpole                                    44  32   5   3   1
##   Mount Pleasant                             32  42  10   8   0
##   Oakridge                                     27  12   2   2   0
##   Point Grey/University of British Columbia   16  15   1   3   0
##   Renfrew-Collingwood                          45  25   5   1   0
##   Riley Park                                   36  45  10   3   3
##   Shaughnessy                                  12  12   3   1   0
##   South Cambie                                  6  16   3   2   1
##   Strathcona                                    4   8   1   0   0
##   West End                                     11  19  12   4   3
##   Yaletown                                      4  11   3   3   1
```

```
cat_anova (van_table_neighbour)
```

```
##                 Df   Sum Sq Mean Sq F value Pr(>F)    
## Var_x           27  1952596   72318   7.724 <2e-16 ***
## Residuals     1732 16215381    9362                   
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

From the results of the Chi-square and ANOVA tests, only instant_bookable was found to not be significant. So, the variables room_type, property_type, minimum_nights and neighbourhood were decided to be included in the generalized linear model.

For the multinomial logistic regression

Along with the generalized linear model, we also decided to create a multinomial logistic regression model. This model will predict the type of room you should rent depending on the minimum number of nights you plan to stay and the nightly price you plan to pay. For categorical analysis, it has already been determined that room_type and price are dependent. Therefore, the dependency of the minimum number of nights and room_type must be assessed.

room_type and minimum nights

Because the minimum_nights is ordinal and room_type is nominal, an ANOVA test was performed to determine the independence between the two variables.

```
van_table_rt_stay = table (van_strat_data_dup$room_type,
van_strat_data_dup$stay)
names ( dimnames (van_table_rt_stay)) = c( 'Room type', 'Night')

van_table_rt_stay
```

```
##                    Night
## Room type             1   2   3   4
##   Entire home/apt 301 631 267  41
##   Hotel room        7   3   0   0
##   Private room    199 196  91   6
##   Shared room      11   3   4   0
```

```
cat_anova (van_table_rt_stay)
```

```
##                Df Sum Sq Mean Sq F value Pr(>F)
## Var_x           3     26   8.676   14.65  2e-09 ***
## Residuals    1756   1040   0.592
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The result of the ANOVA test, F = 14.65, df = 3, 1756 (p-value = 0.000000002 < 0.05), indicates that room_type and minimum_nights are dependent. Therefore, the type of room you rent is affected by the minimum number of nights the room is available, so, the minimum-nights should be kept in the model.

Generalized Linear Regression Model

1. Split the data into train & test using SRS using 75% and 25% ratios respectively

```
set.seed(1000)
sample = sample.split(airbnb_data, SplitRatio = 0.75)
train_srs = subset(airbnb_data, sample == TRUE)
test_srs  = subset(airbnb_data, sample == FALSE)

head(test_srs)
```

```
##       id                                 name      host_name
## 2  13188     Garden level studio in ideal loc.     Family Guns
## 7  17158   Vancouver 4br 3ba house 20min to DT          Elaine
## 12 18795    *Best choice in downtown-5 Stars *             Kai
## 14 20085 Bright & Cheerful Uptown Garden Suite Tony And Rachel
## 15 20566      Downtown Vancouver Perfect Suite           Shona
## 22 33150   â\200œSuite on Mainâ\200\235  near  Q.E Park              Ili
##    host_response_rate  host_neighbourhood host_listings_count
## 2                100%          Riley Park                   2
## 7                100% Renfrew-Collingwood                   1
## 12                94%            West End                   3
## 14               100%      Mount Pleasant                   1
## 15                N/A  Downtown Vancouver                   1
## 22               100%          Riley Park                   0
##                 street       neighbourhood zipcode latitude longitude
## 2  Vancouver, BC, Canada          Riley Park     V5Y 49.24577 -123.1052
## 7  Vancouver, BC, Canada Renfrew-Collingwood V5M 2V4 49.25195 -123.0355
## 12 Vancouver, BC, Canada            West End     V6E 49.28251 -123.1283
## 14 Vancouver, BC, Canada      Mount Pleasant V5T 2V6 49.25732 -123.0854
## 15 Vancouver, BC, Canada  Downtown Vancouver V6Z 2S1 49.28219 -123.1250
## 22 Vancouver, BC, Canada          Riley Park V5V 3L1 49.24686 -123.1047
##    property_type       room_type price minimum_nights availability_30
## 2      Apartment Entire home/apt   120              2              13
## 7          House Entire home/apt   135              4               0
## 12         House    Private room    95             30              21
## 14   Guest suite Entire home/apt   110             30               0
## 15     Apartment Entire home/apt   231              3               0
## 22     Apartment Entire home/apt   100              3               0
##    availability_60 availability_90 availability_365 number_of_reviews
## 2               28              36              183               225
## 7                0               0               14                 5
## 12              35              65              340               119
## 14               0               6              228                72
## 15               0               0                7                53
## 22               0               7              254                44
##    review_scores_value requires_license instant_bookable
## 2                   10                t                t
## 7                   10                t                f
## 12                   9                t                f
## 14                   9                t                f
## 15                   9                t                f
## 22                  10                t                f
##            cancellation_policy reviews_per_month      long     lat
## 2                     moderate              1.90 -123.1075 49.2475
## 7  strict_14_with_grace_period              0.05 -123.0375 49.2525
## 12 strict_14_with_grace_period              1.15 -123.1275 49.2825
## 14                    moderate              0.61 -123.0875 49.2575
## 15                    flexible              0.45 -123.1275 49.2825
## 22 strict_14_with_grace_period              0.43 -123.1025 49.2475
##                  grid
## 2  -123.1075 49.2475
## 7  -123.0375 49.2525
## 12 -123.1275 49.2825
## 14 -123.0875 49.2575
## 15 -123.1275 49.2825
## 22 -123.1025 49.2475
```

2. k-fold cross validation

Based on the independence tests performed earlier, we know the most relevant variables are "neighbourhood", "property_type", "room_type", "minimum_nights". However, to reaffirm our result we are going to use K-fold cross validation to see if any combination of three of these variables can create an error close to 4-variable model so we can build a less complicated model with almost the same accuracy. K=10 has been used for the cross validation. It can be seen that having all 4 variables in the model returns a significantly smaller error. Surprisingly, eliminating the "room_type" from the model resulted in a huge increase in the MSE (8562.144) while MSEs for the other three combinations remain fairly in the same range. This indicates that "room_type" has a significant impact on the model and should be included. The result of k-fold cross validation reaffirms that the model with four variables is the best choice. So, using three variables will create non-negligible error difference

```
n_iter = 5
cv_error_a = cv_error_b = cv_error_c = cv_error_d =numeric(n_iter)

for(i in 1:n_iter){
  model_fit_a<-train(price
~neighbourhood+property_type+room_type+minimum_nights, data=airbnb_data,
trControl = trainControl(method = "cv", number = 10), method='glm',
family="gaussian")
  model_fit_b<-train(price ~property_type+room_type+minimum_nights,
data=airbnb_data, trControl = trainControl(method = "cv", number = 10),
method='glm', family="gaussian")
  model_fit_c<-train(price ~neighbourhood+room_type+minimum_nights,
data=airbnb_data, trControl = trainControl(method = "cv", number = 10),
method='glm', family="gaussian")
  model_fit_d<-train(price ~neighbourhood+property_type+minimum_nights,
data=airbnb_data, trControl = trainControl(method = "cv", number = 10),
method='glm', family="gaussian")


  cv_error_a[i]=  as.numeric(model_fit_a$results[2]) # returns the RMSE

  cv_error_b[i]=  as.numeric(model_fit_b$results[2]) # returns the RMSE

  cv_error_c[i]=  as.numeric(model_fit_c$results[2]) # returns the RMSE

  cv_error_d[i]=  as.numeric(model_fit_d$results[2]) # returns the RMSE


}

# to get MSE
data.frame(All_var_MSE = cv_error_a^2,property_room_minimum_nights_MSE
=cv_error_b^2,neighbour_room_minimum_nights_MSE
=cv_error_c^2,neighbour_property_minimum_nights_MSE =cv_error_d^2)

##   All_var_MSE property_room_minimum_nights_MSE
## 1    6408.680                         6788.375
## 2    6411.870                         6792.005
## 3    6417.667                         6790.021
## 4    6395.129                         6778.348
## 5    6402.499                         6838.059
##   neighbour_room_minimum_nights_MSE neighbour_property_minimum_nights_MSE
## 1                          6890.288                              8561.431
## 2                          6882.961                              8556.825
## 3                          6880.475                              8554.176
## 4                          6891.280                              8579.241
## 5                          6871.007                              8559.061

# Average all the MSEs to compare
data.frame(all_four = mean(cv_error_a)^2,property_room_nights
=mean(cv_error_b)^2,neighbour_room_nights=
mean(cv_error_c)^2,neighbour_property_nights= mean(cv_error_d)^2)

##   all_four property_room_nights neighbour_room_nights
## 1 6407.167             6797.345                6883.2
##   neighbour_property_nights
## 1                  8562.144
```

3. str() function checks if categorical variables are already converted to a factor or not. If they are, no need to use factor() function in the model. As shown below, "neighbourhood","property_type" and "room_type" are all factor.

```
str(train_srs)
```

```
## 'data.frame':    4401 obs. of  28 variables:
##  $ id                 : int  10080 13357 13490 14267 16254 17765 18270 
18589 18773 19527 ...
##  $ name               : Factor w/ 6120 levels "","- Luxury New Condo With 
City Views -",..: 2293 7 5745 2557 1746 1516 4621 1825 1766 772 ...
##  $ host_name          : Factor w/ 2266 levels "","A","Ã–nder",..: 1660 
1213 851 1607 905 883 1662 1980 483 248 ...
##  $ host_response_rate : Factor w/ 40 levels "","0%","100%",..: 22 3 30 40 
40 3 30 3 3 3 ...
##  $ host_neighbourhood : Factor w/ 67 levels "","Arbutus Ridge",..: 9 8 28 
28 25 40 40 10 32 32 ...
##  $ host_listings_count: int  39 4 4 1 1 1 2 1 1 1 ...
##  $ street             : Factor w/ 16 levels " Vancouver, BC, Canada",..: 
12 12 12 12 12 12 12 12 12 12 ...
##  $ neighbourhood      : Factor w/ 29 levels "Arbutus Ridge",..: 4 7 15 15 
14 20 20 13 18 9 ...
##  $ zipcode            : Factor w/ 2721 levels "",".","112345",..: 1714 
1497 342 1053 10 1221 753 193 1972 1844 ...
##  $ latitude           : num  49.3 49.3 49.3 49.2 49.3 ...
##  $ longitude          : num  -123 -123 -123 -123 -123 ...
##  $ property_type      : Factor w/ 21 levels "Aparthotel","Apartment",..: 9 
2 2 14 11 2 9 14 2 2 ...
##  $ room_type          : Factor w/ 4 levels "Entire home/apt",..: 1 1 1 1 1 
1 3 3 1 1 ...
##  $ price              : num  151 152 145 140 195 120 55 85 169 168 ...
##  $ minimum_nights     : int  60 30 30 2 3 1 30 1 3 28 ...
##  $ availability_30    : int  0 0 23 0 0 0 0 0 0 26 ...
##  $ availability_60    : int  6 11 29 0 0 0 0 0 10 56 ...
##  $ availability_90    : int  36 41 51 0 0 0 0 0 10 86 ...
##  $ availability_365   : int  311 316 296 0 73 0 0 7 10 86 ...
##  $ number_of_reviews  : int  16 57 84 31 5 193 117 381 14 35 ...
##  $ review_scores_value: int  9 8 10 9 9 9 9 10 8 9 ...
##  $ requires_license   : Factor w/ 2 levels "f","t": 2 2 2 2 2 2 2 2 2 2 
...
##  $ instant_bookable   : Factor w/ 2 levels "f","t": 2 1 1 2 2 1 1 1 1 1 
...
##  $ cancellation_policy: Factor w/ 4 levels "flexible","moderate",..: 4 4 4 
4 2 4 4 4 4 1 ...
##  $ reviews_per_month  : num  0.16 0.48 0.82 0.28 0.32 2.23 1.11 3.71 0.12 
0.3 ...
##  $ long               : num  -123 -123 -123 -123 -123 ...
##  $ lat                : num  49.3 49.3 49.3 49.2 49.3 ...
##  $ grid                : chr  "-123.1225 49.2875" "-123.1075 49.2775" "-
123.0675 49.2575" "-123.0825 49.2475" ...
```

4. Build a GLM Linear Regression Model using the SRS sampled training set (75% of the whole dataset)

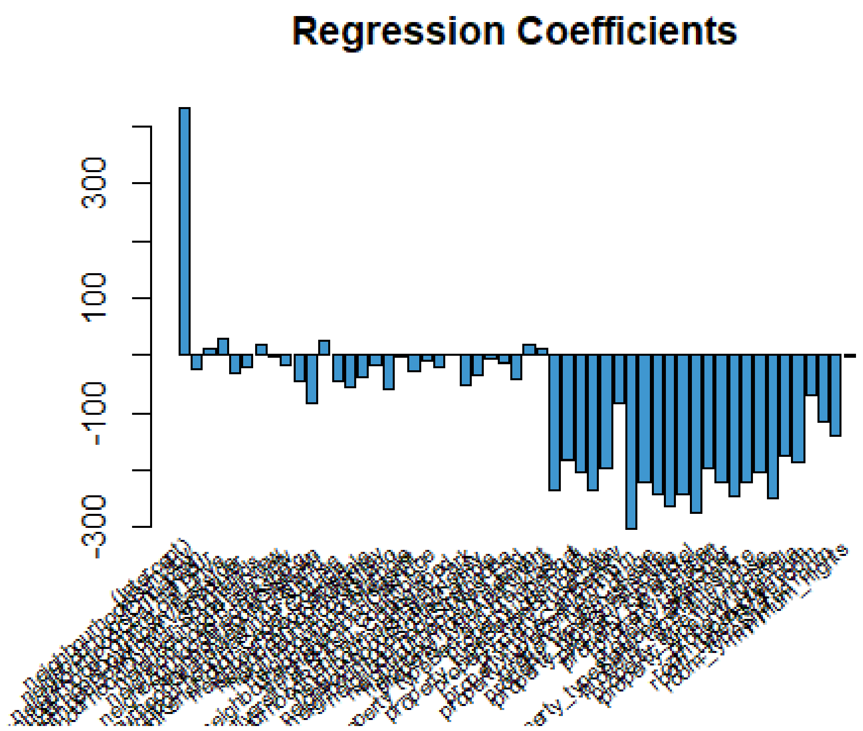
The reason we chose these four variables as the most relevant ones is if someone suddenly decided to stay in Vancouver knowing the neighbourhood, minimum length of stay, type of room and type of property, what would be the expected price?

The result shows that only some levels of the categorical variable " neighbourhood " are statistically significant. However, in this situation all levels should be allowed to be in the model as selecting only some will change the coefficient interpretation. The other three variables are almost significant and should be included in the model. In terms of the deviance, it seems that null deviance is dramatically larger than the residual deviance which signals that having these variables decreases the fit error.

```
LR_model = glm(price
~neighbourhood+property_type+room_type+minimum_nights,data = train_srs,
family = gaussian(link="identity"))
summary(LR_model)
```

5. A bar chart is plotted in order to better represent the model coefficients for LR_model trained with SRS sampled training set. As shown, the coefficients for “neighbourhood” are not as large as the ones for “property_type” and “room_type” which aligns with the t-test results.

```
coeffs <- coefficients(LR_model)
      mp <- barplot(coeffs, col="#3F97D0", xaxt='n', main="Regression
Coefficients")
      lablist <- names(coeffs)
      text(mp, par("usr")[3], labels = lablist, srt = 40, adj = c(1.1,1.1),
xpd = TRUE, cex=0.6)
```



6. In order to see how good the model is fitted to our training set, we now test the accuracy of the model with the SRS sampled training set and calculate the error.

as expected the mean of the observed data and fitted data are the same and the MSE is even lower than what we calculated in the cross validation stage but fairly in the same range.

```
options(scipen = 999)
fitted_lmSRS_train <- predict(LR_model,newdata=train_srs,type='response')

comp_srs_train = data.frame(Observed_value = train_srs$price,Fitted_value =
round(fitted_lmSRS_train,0), Squared_Error = round((train_srs$price-
fitted_lmSRS_train)^2,2) )
head(comp_srs_train)

##   Observed_value Fitted_value Squared_Error
## 1            151          153          2.34
## 3            152          172        395.68
## 4            145          115        903.78
## 5            140          197       3213.59
## 6            195          111       7020.89
## 8            120          185       4272.83

c(Ave_observed = mean(train_srs$price),Ave_fitted = mean(fitted_lmSRS_train),
MSE =mean(comp_srs_train$Squared_Error))

## Ave_observed   Ave_fitted          MSE
##      153.958      153.958     6305.290
```

7. In order to see how good the model can predict, we now test the prediction accuracy of the model with the SRS sampled test set and calculate the error.

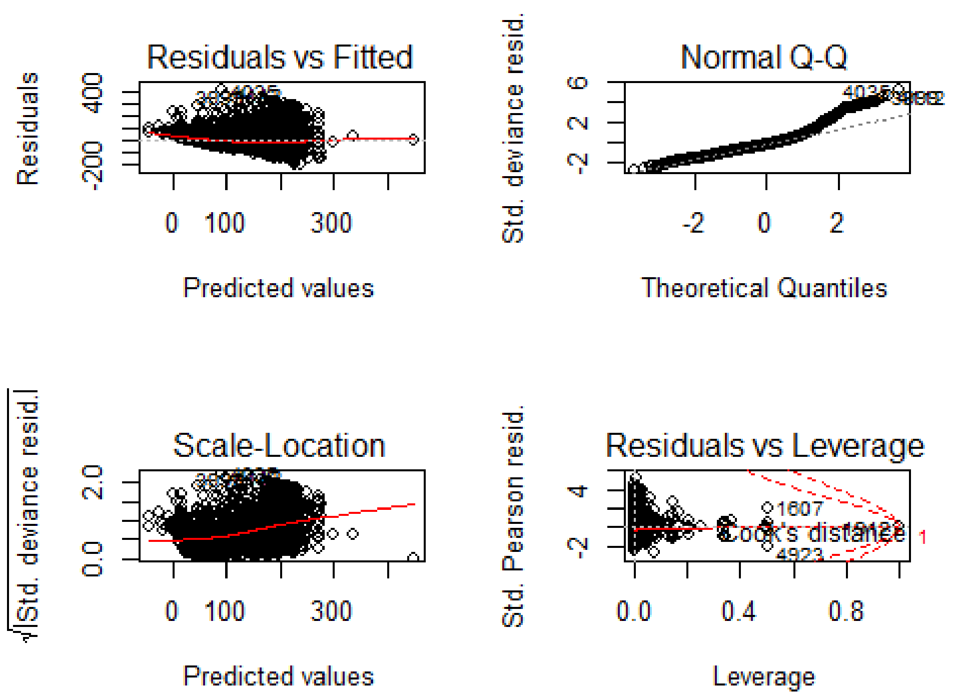
the result shows a fairly good estimate for the mean and also MSE in the same range as the accuracy with the training set. In general the MSE is too high even though the model is done a pretty good job at estimating the mean.

```
fitted_lmSRS <- predict(LR_model,newdata=test_srs,type='response')

comp_srs = data.frame(Observed_value = test_srs$price,Fitted_value =
round(fitted_lmSRS,0), Squared_Error = round((test_srs$price-
fitted_lmSRS)^2,2) )
head(comp_srs)

##    Observed_value Fitted_value Squared_Error
## 2             120          159       1530.73
## 7             135          178       1866.87
## 12             95           99         13.19
## 14            110          116         35.79
## 15            231          212        370.68
## 22            100          158       3323.34

c(Ave_observed = mean(test_srs$price),Ave_fitted = mean(fitted_lmSRS), MSE
=mean(comp_srs$Squared_Error))

## Ave_observed   Ave_fitted          MSE
##     156.1587     154.8143    6356.4254
```

8. Assumption Test is not required when GLM is used rather than LM. However, all the assumption have been checked to gain better understanding about the data.

The following plot shows that there is a funnel shape pattern in the residual plot . So we can say that these data has none of the linearity, normality and homoscedasticity. Although, these are not required one can relate the high error of the model to the heterogeneous nature of these data.

```
par(mfrow = c(2, 2))
plot(LR_model)

## Warning in sqrt(crit * p * (1 - hh)/hh): NaNs produced

## Warning in sqrt(crit * p * (1 - hh)/hh): NaNs produced
```



8.1 Testing Equal Variance using Breusch-Pagan test. The test hypothesis

Ho: homoscedasticity

Ha: heteroscedasticity

From the above output of Breusch-Pagan test p-value is less than alpha=0.05 so we reject the null hypothesis and conclude that heteroscedasticity is present.

```
bptest(LR_model)

##
##  studentized Breusch-Pagan test
##
## data:  LR_model
## BP = 375.59, df = 52, p-value < 0.00000000000000022
```

8.2 Testing Normality using Shapiro-Wilk test

Ho: the sample data are significantly normally distributed

Ha: the sample data are not significantly normally distributed

P_value is smaller than 0.05 and we can reject our H0 meaning that the data is not normally distributed

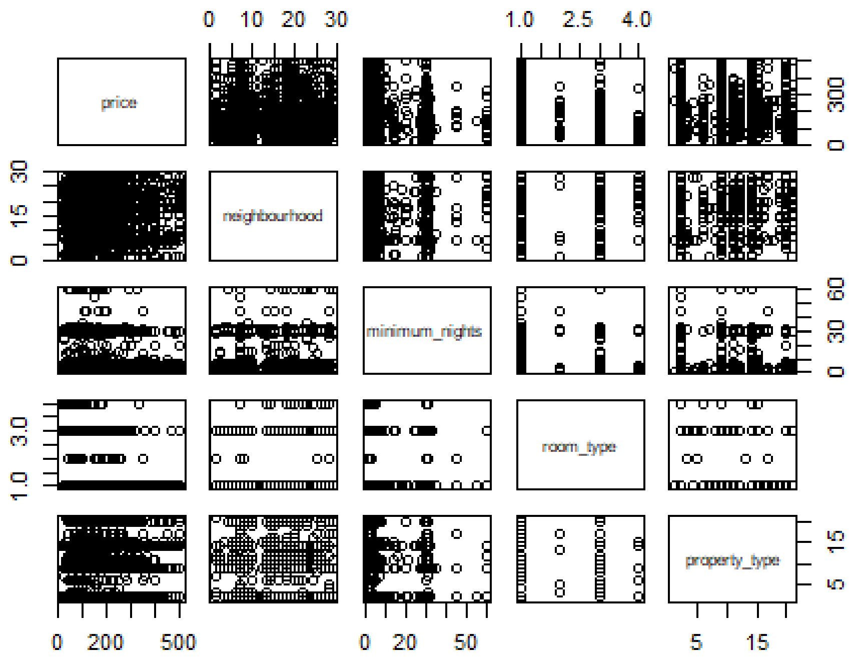
```
shapiro.test(residuals(LR_model))
```

```
##
##  Shapiro-Wilk normality test
##
## data:  residuals(LR_model)
## W = 0.91759, p-value < 0.00000000000000022
```

8.3 Testing for Multicollinearity using correlation plot and VIF test

From the below plot and VIF test we can state that there is no multicollinearity in our variables.

```
pairs(~price +
neighbourhood+minimum_nights+room_type+property_type,data=airbnb_data)
```



```
X1<-cbind( airbnb_data$neighbourhood, airbnb_data$minimum_nights,
airbnb_data$room_type,airbnb_data$property_type)
imcdiag(X1,airbnb_data$price, method="VIF")

##
## Call:
## imcdiag(x = X1, y = airbnb_data$price, method = "VIF")
##
##
##  VIF Multicollinearity Diagnostics
##
##        VIF detection
## V1 1.0396         0
## V2 1.0245         0
## V3 1.0704         0
## V4 1.1014         0
##
## NOTE:  VIF Method Failed to detect multicollinearity
##
##
## 0 --> COLLINEARITY is not detected by the test
##
## ===================================
```

9. Now we try to build a GLM Linear Regression Model with “interactions” using the SRS training set (75% of the whole dataset) with the hope of improving the fit.

Comparing the interaction model with the original model shows, both the residual deviance and AIC of the interaction model are larger than the original model. It seems that adding interactions just make the model more complex than accurate. So, we stick to the original model

```
LR_model_int = glm(price
~(neighbourhood+property_type+room_type+minimum_nights)^2,data = train_srs,
family = gaussian(link="identity"))
summary(LR_model_int)
```

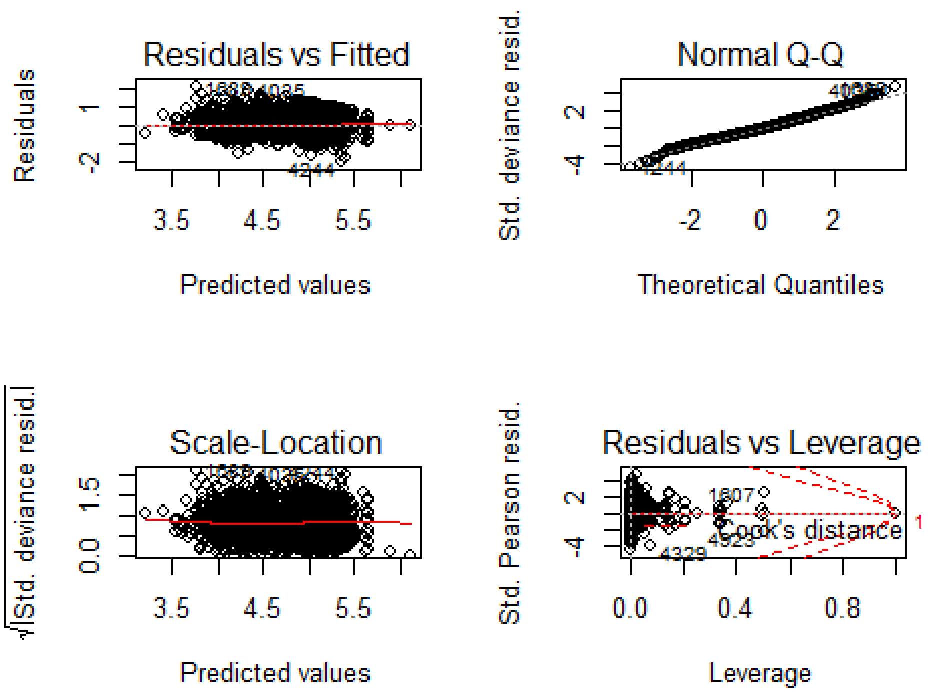
10. In order to decrease the heterogeneity of the data, a LOG Transform has been applied to the response variable in the GLM Linear Regression Model using the SRS training set

```
LR_model_log = glm(log(price)
~neighbourhood+property_type+room_type+minimum_nights,data = train_srs,
family = gaussian(link="identity"))
summary(LR_model_log)
```

10.1 checking if Log transform reduced the variability in our residuals.

The plot shows that log transform has visually improved the spread of residuals and also the normality

```
par(mfrow = c(2, 2))
plot(LR_model_log)

## Warning in sqrt(crit * p * (1 - hh)/hh): NaNs produced

## Warning in sqrt(crit * p * (1 - hh)/hh): NaNs produced
```



10.2 Equal Variance Assumption for the log-transformed model is being tested with Breusch-Pagan test

Ho: homoscedasticity

Ha: heteroscedasticity

The test returns a p-value which is less than alpha=0.05. So we reject the null hypothesis and conclude that heteroscedasticity is still present.

```
bptest(LR_model_log)

##
##  studentized Breusch-Pagan test
##
## data:  LR_model_log
## BP = 126.26, df = 52, p-value = 0.00000004075
```

11. Test the accuracy of the log-transformed Linear Regression Model using the " SRS sampled " test data. It seems that the log transform shows less accuracy and higher error than the original LR model. So, we can conclude that log transformed model is not a suitable model for these data

```
options(scipen = 999)
fitted_lmSRS_log <- predict(LR_model_log,newdata=test_srs,type='response')

comp_srs_log = data.frame(Observed_value = test_srs$price,Fitted_value =
round(exp(fitted_lmSRS_log),0), Squared_Error = round((test_srs$price-
exp(fitted_lmSRS_log))^2,2) )
head(comp_srs_log)

##    Observed_value Fitted_value Squared_Error
## 2             120          142        502.63
## 7             135          133          2.38
## 12             95           76        351.77
## 14            110          109          0.33
## 15            231          197       1161.40
## 22            100          141       1681.89

c(Ave_observed = mean(test_srs$price),Ave_fitted =
mean(exp(fitted_lmSRS_log)), MSE =mean(comp_srs_log$Squared_Error))

## Ave_observed   Ave_fitted          MSE
##     156.1587     139.3462    6581.2433
```

12. Split the data into training & test sets using proportional stratified sampling with "room_type" as the strata. This is mainly to compare the result with the model built with the SRS sampled data

```
# N should be sorted descendingly otherwise there will be an error
N=sort(round(table(airbnb_data$room_type)*0.25),decreasing = TRUE)
set.seed(835)  # Ryan's seed is  1032
cl=sampling:::strata(airbnb_data, stratanames=c("room_type"), N,
method="srswor")
test_strat=getdata(airbnb_data,cl)
train_strat=airbnb_data[-cl$ID_unit,]
```

13. Build a GLM Linear Regression Model using the Stratified training set (75% of the whole dataset).

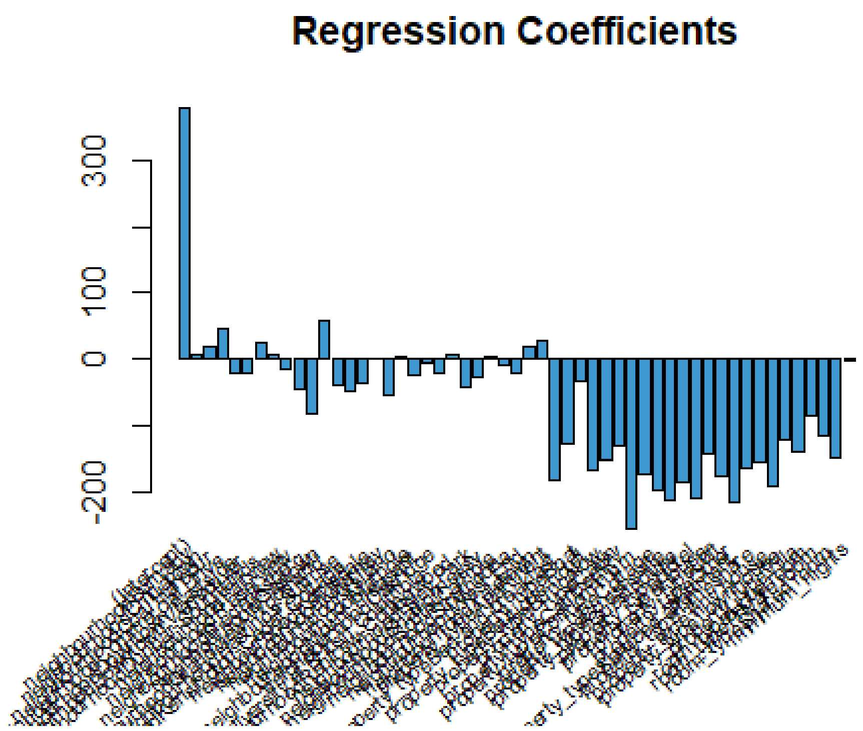
The t-test shows some levels of "neighbourhood" and also "property_type" are not significant but we should keep them in the model. The other two variables are both significant

```
LR_model2 = glm(price
~neighbourhood+property_type+room_type+minimum_nights,data = train_strat,
family = gaussian(link="identity"))
summary(LR_model2)
```

14. plot the model coefficients for Linear Regression Model with stratified sampled training set.

The following plot shows that the coefficients for ""property_type" and "room_type" are larger than the ones for "neighbourhood"

```
coeffs_strat <- coefficients(LR_model2)
      mp_strat <- barplot(coeffs_strat, col="#3F97D0", xaxt='n',
main="Regression Coefficients")
      lablist_strat <- names(coeffs_strat)
      text(mp_strat, par("usr")[3], labels = lablist_strat, srt = 40, adj =
c(1.1,1.1), xpd = TRUE, cex=0.6)
```



15. Test the accuracy of the Linear Regression Model using the " stratified sampled" test set.

It appears that test data created with stratified sampling method led to a slightly smaller MSE than the one created with SRS. This might be due to setting the room_type which showed to be a critical variable as strata and sample based on that.

```
fitted_lmstrat <- predict(LR_model2,newdata=test_strat,type='response')

comp_strat = data.frame(Observed_value =test_strat$price,Fitted_value
=round(fitted_lmstrat,0), Squared_Error = round((test_strat$price-
fitted_lmstrat)^2,2))
head(comp_strat)

##    Observed_value Fitted_value Squared_Error
## 11            169          193        564.69
## 15            231          216        234.43
## 24            158          183        628.65
## 25            112          192       6457.20
## 28            129          174       1985.50
## 31            110          174       4039.74

c(Ave_observed = mean(test_strat$price),Ave_fitted = mean(fitted_lmstrat),
MSE =mean(comp_strat$Squared_Error))

## Ave_observed   Ave_fitted          MSE
##     151.6605     154.2782    6294.8132
```

16. Test the accuracy of the Linear Regression Model using the " stratified sampled" training set

```
fitted_lmstrat_train <-
predict(LR_model2,newdata=train_strat,type='response')

comp_strat_train = data.frame(Observed_value =train_strat$price,Fitted_value
=round(fitted_lmstrat_train,0), Squared_Error = round((train_strat$price-
fitted_lmstrat_train)^2,2))
head(comp_strat_train)

##   Observed_value Fitted_value Squared_Error
## 1            151          158         43.94
## 2            120          163       1891.76
## 3            152          174        464.79
## 4            145          113       1040.90
## 5            140          197       3236.20
## 6            195          111       6984.21

c(Ave_observed = mean(train_strat$price),Ave_fitted =
mean(fitted_lmstrat_train), MSE =mean(comp_strat_train$Squared_Error))

## Ave_observed   Ave_fitted          MSE
##     155.4575     155.4575    6315.7330
```

17. Build a GLM multinomial Regression Model using the SRS training set (75% of the whole dataset) and VGLM function

A multinomial logistic regression model with all four variables used in the linear model was built initially but the result shows that none of the levels of property_type and neighbourhood are significant. So, they have been removed and only price and minimum nights were kept as the exploratory variables against room_type (response variable).

The motivation to build this models is if someone knew how much he/she can spend on a airbnb rental property and at least how many nights he/she wanted to stay, would he/she be able to predict which type of room they are going to end up with?

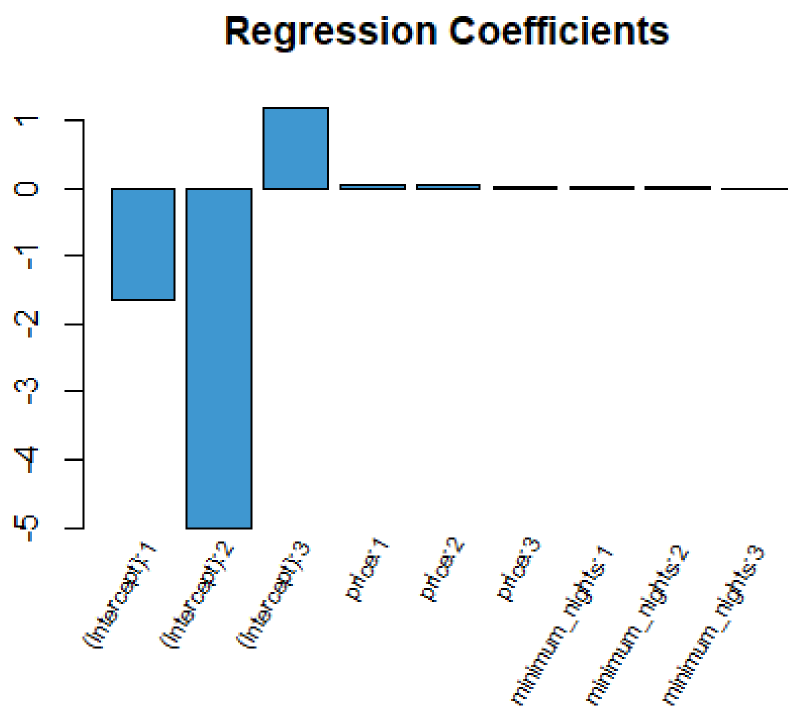
The Z-test shows that “price” is significant in all three logit equations. However, “minimum_nights” was only significant in one but we still include it in the model.

```
options(scipen = FALSE)


logit_roomtype_vglm=vglm(room_type~price+minimum_nights,family=multinomial,da
ta=train_srs)
summary(logit_roomtype_vglm)

##
## Call:
## vglm(formula = room_type ~ price + minimum_nights, family = multinomial,
##     data = train_srs)
##
```

```
## Pearson residuals:
##                       Min       1Q   Median       3Q    Max
## log(mu[,1]/mu[,4]) -9427 -0.29494  0.12544  0.38622   2.55
## log(mu[,2]/mu[,4]) -9406 -0.04996 -0.04272 -0.02276  27.84
## log(mu[,3]/mu[,4]) -9397 -0.34750 -0.10189  0.51532 364.86
##
## Coefficients:
##                    Estimate Std. Error z value Pr(>|z|)
## (Intercept):1    -1.656732   0.435549  -3.804 0.000143 ***
## (Intercept):2    -5.000425   0.619460  -8.072 6.90e-16 ***
## (Intercept):3     1.182001   0.424168   2.787 0.005326 **
## price:1           0.060653   0.006463   9.385  < 2e-16 ***
## price:2           0.054124   0.006973   7.762 8.38e-15 ***
## price:3           0.030693   0.006409   4.789 1.68e-06 ***
## minimum_nights:1  0.034926   0.013070   2.672 0.007536 **
## minimum_nights:2 -0.015281   0.026552  -0.576 0.564934
## minimum_nights:3  0.003897   0.012960   0.301 0.763623
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Names of linear predictors: log(mu[,1]/mu[,4]), log(mu[,2]/mu[,4]),
## log(mu[,3]/mu[,4])
##
## Residual deviance: 4212.592 on 13194 degrees of freedom
##
## Log-likelihood: -2106.296 on 13194 degrees of freedom
##
## Number of Fisher scoring iterations: 9
##
## Warning: Hauck-Donner effect detected in the following estimate(s):
## '(Intercept):2', 'price:1'
##
##
## Reference group is level  4  of the response
```

18. plot the model coefficients for MR model with SRS training set

```
coeffs_MR <- coefficients(logit_roomtype_vglm)
      mp_MR <- barplot(coeffs_MR, col="#3F97D0", xaxt='n', main="Regression
Coefficients")
      lablist_MR <- names(coeffs_MR)
      text(mp_MR, par("usr")[3], labels = lablist_MR, srt = 60, adj =
c(1.1,1.1), xpd = TRUE, cex=0.7)
```



19. goodness-of-fit test. The hypotheses for this test are:

H0: there is no significant difference between the observed and the expected value.

Ha: there is a significant difference between the observed and the expected value.

It seems that the p_value is very small and close to zero. Then we can reject the null hypothesis in favor of the alternative meaning there is a difference between the observed and predicted values.

```
1-pchisq(sum(resid(logit_roomtype_vglm,
type="pearson")^2),df.residual(logit_roomtype_vglm))

## [1] 0
```

20. Check the accuracy of our GLM multinomial Regression Model on the SRS sampled training set. The accuracy shows the model predicted the correct room_type 82.59% of the time on average when using the training set

```
prob.fit_vglm2<-predict(logit_roomtype_vglm, newdata=train_srs,
type="response") # return the probability of being in one of four categories
of the room type
fitted.result_vglm2<-colnames(prob.fit_vglm2)[rowMaxs(prob.fit_vglm2)] # turn
probabilities into actual categories, pick the max of each row and return the
column name associated with it
head(data.frame(Observed_value = train_srs$room_type, Fitted_value =
fitted.result_vglm2))
```

```
##    Observed_value    Fitted_value
## 1 Entire home/apt Entire home/apt
## 2 Entire home/apt Entire home/apt
## 3 Entire home/apt Entire home/apt
## 4 Entire home/apt Entire home/apt
## 5 Entire home/apt Entire home/apt
## 6 Entire home/apt Entire home/apt

misClasificError_vglm2 <- mean(fitted.result_vglm2 != train_srs$room_type)
print(paste('Accuracy %',round((1-misClasificError_vglm2)*100 ,2)))

## [1] "Accuracy % 82.59"
```

21. Check the accuracy of our GLM multinomial Regression Model on the SRS sampled test set. The accuracy shows the model predicted the correct room_type 83.31% of the time on average when using the test set

```
prob.fit_vglm<-predict(logit_roomtype_vglm, newdata=test_srs,
type="response") # return the probability of being in one of four categories
of the room type
fitted.result_vglm<-colnames(prob.fit_vglm)[rowMaxs(prob.fit_vglm)] # turn
probabilities into actual categories, pick the max of each row and return the
column name associated with it
head(data.frame(Observed_value = test_srs$room_type, Fitted_value =
fitted.result_vglm))

##    Observed_value    Fitted_value
## 1 Entire home/apt Entire home/apt
## 2 Entire home/apt Entire home/apt
## 3    Private room Entire home/apt
## 4 Entire home/apt Entire home/apt
## 5 Entire home/apt Entire home/apt
## 6 Entire home/apt Entire home/apt

misClasificError_vglm <- mean(fitted.result_vglm != test_srs$room_type)
print(paste('Accuracy %',round((1-misClasificError_vglm)*100 ,2)))

## [1] "Accuracy % 83.31"
```

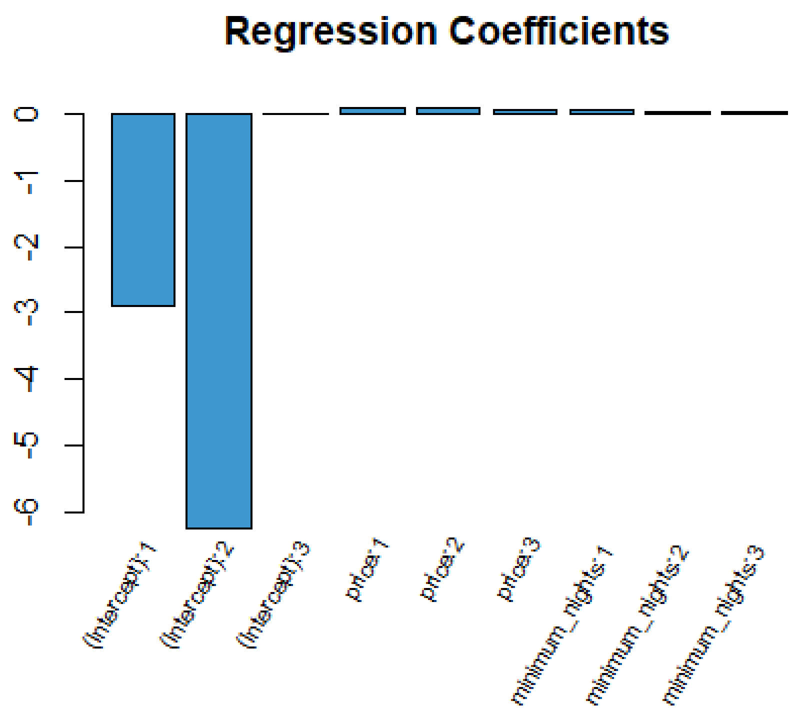
22. Build a GLM multinomial Regression Model using the stratified training set (75% of the whole dataset) and VGLM function. Again Z-test shows that "price" is significant in all three logit equations. However, "minimum_nights" was only significant in one but we still keep it in the model. This model shows a smaller residual deviance compare to the logistic model with SRS training set which can be deemed as a better model thanks to stratified sampled test set

```
options(scipen = FALSE)

logit_roomtype_vglm_strat=vglm(room_type~price+minimum_nights,family=multinom
ial,data=train_strat)
summary(logit_roomtype_vglm_strat)
```

```
##
## Call:
## vglm(formula = room_type ~ price + minimum_nights, family = multinomial,
##     data = train_strat)
##
## Pearson residuals:
##                       Min       1Q   Median       3Q     Max
## log(mu[,1]/mu[,4]) -299.7 -0.27948  0.11691  0.37654   2.795
## log(mu[,2]/mu[,4]) -292.0 -0.05124 -0.03952 -0.03067  24.410
## log(mu[,3]/mu[,4]) -296.6 -0.34088 -0.09174  0.53157 401.969
##
## Coefficients:
##                   Estimate Std. Error z value Pr(>|z|)
## (Intercept):1    -2.906343   0.481060  -6.042 1.53e-09 ***
## (Intercept):2    -6.233774   0.665251  -9.371  < 2e-16 ***
## (Intercept):3    -0.012083   0.468646  -0.026  0.97943
## price:1           0.082280   0.008330   9.878  < 2e-16 ***
## price:2           0.074551   0.008794   8.477  < 2e-16 ***
## price:3           0.051824   0.008270   6.267 3.69e-10 ***
## minimum_nights:1  0.045264   0.013776   3.286  0.00102 **
## minimum_nights:2  0.015999   0.023151   0.691  0.48953
## minimum_nights:3  0.013172   0.013620   0.967  0.33348
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Names of linear predictors: log(mu[,1]/mu[,4]), log(mu[,2]/mu[,4]),
## log(mu[,3]/mu[,4])
##
## Residual deviance: 4103.249 on 13197 degrees of freedom
##
## Log-likelihood: -2051.624 on 13197 degrees of freedom
##
## Number of Fisher scoring iterations: 9
##
## Warning: Hauck-Donner effect detected in the following estimate(s):
## '(Intercept):2', 'price:1'
##
##
## Reference group is level  4  of the response
```

23. plot the model coefficients for MR model with STRATIFIED training set

```
coeffs_MR2 <- coefficients(logit_roomtype_vglm_strat)
      mp_MR2 <- barplot(coeffs_MR2, col="#3F97D0", xaxt='n', main="Regression
Coefficients")
      lablist_MR2 <- names(coeffs_MR2)
      text(mp_MR2, par("usr")[3], labels = lablist_MR2, srt = 60, adj =
c(1.1,1.1), xpd = TRUE, cex=0.7)
```



24. Check the accuracy of our GLM multinomial Regression Model on the stratified training set. The accuracy shows the model predicted the correct room_type 82.69% of the time on average when using the training set. It is almost the same as the accuracy for the model with SRS training set.

```
prob.fit_vglm3<-predict(logit_roomtype_vglm_strat, newdata=train_strat,
type="response") # return the probability of being in one of four categories
of the room type
fitted.result_vglm3<-colnames(prob.fit_vglm3)[rowMaxs(prob.fit_vglm3)] # turn
probabilities into actual categories, pick the max of each row and return the
column name associated with it
head(data.frame(Observed_value = train_strat$room_type, Fitted_value =
fitted.result_vglm3))

##    Observed_value    Fitted_value
## 1 Entire home/apt Entire home/apt
## 2 Entire home/apt Entire home/apt
## 3 Entire home/apt Entire home/apt
## 4 Entire home/apt Entire home/apt
## 5 Entire home/apt Entire home/apt
## 6 Entire home/apt Entire home/apt

misClasificError_vglm3 <- mean(fitted.result_vglm3 != train_strat$room_type)
print(paste('Accuracy %',round((1-misClasificError_vglm3)*100 ,2)))

## [1] "Accuracy % 82.69"
```

25. Check the accuracy of our GLM multinomial Regression Model on the stratified test set. The accuracy shows the model predicted the correct room_type 83.03% of the time on average when using the TEST dataset. It is almost the same as the accuracy for the model with SRS training set.

```
prob.fit_vglm4<-predict(logit_roomtype_vglm_strat, newdata=test_strat,
type="response") # return the probability of being in one of four categories
of the room type
fitted.result_vglm4<-colnames(prob.fit_vglm4)[rowMaxs(prob.fit_vglm4)] # turn
probabilities into actual categories, pick the max of each row and return the
column name associated with it
head(data.frame(Observed_value = test_strat$room_type, Fitted_value =
fitted.result_vglm4))

##    Observed_value    Fitted_value
## 1 Entire home/apt Entire home/apt
## 2 Entire home/apt Entire home/apt
## 3 Entire home/apt Entire home/apt
## 4 Entire home/apt Entire home/apt
## 5 Entire home/apt Entire home/apt
## 6 Entire home/apt Entire home/apt

misClasificError_vglm4 <- mean(fitted.result_vglm4 != test_strat$room_type)
print(paste('Accuracy %',round((1-misClasificError_vglm4)*100 ,2)))

## [1] "Accuracy % 83.03"
```

Conclusion

This project has been done in four main sections, Data Wrangling, Sampling, Contingency Tables and Modeling using Vancouver's Airbnb dataset. All the findings and results achieved through the statistical analysis are as follows:

a. We conclude from our preliminary analysis that most of the Airbnb property in Vancouver are either Entire homes/apartments with an average required minimum nights booking of 2 days. Most of the Airbnb properties in Vancouver are available in a price range of $45 to $150 and located in downtown or nearby location.

b. Stratified sampling produces a better representation of the mean compare to SRS sampling

c. Cluster sampling is not an appropriate technique for this dataset. One of the reasons might be number of data point that we have in each cluster (neighbourhood) is widely different and sometimes the quantity of data points in each cluster it is too small.

d. It seems that using Stratified sampled train and test set to build a model with this dataset leads to a slightly less error (MSE) in the model but the difference is not significant enough to deterministically rule out one method over another

e. An interaction linear regression model was built but its AIC was higher than the original model and was too complex to be used. Therefore, it was initially ruled out as as an alternative

f. Log-transform was applied to the response variable (price) in the original linear regression model which Visually improved the homogeneity of the data but BP test still rejected the homoscedasticity of the data. Therefore, It was ruled our as it didn't provide any added-value to the current model.

g. After using all the sampling and modeling techniques, we can also conclude that it is not possible to build a robust linear model to predict the price based on the available information.

Division of labour

Gaurav was responsible for the data wrangling and performing the preliminary data analysis with all the various bar charts, box plots, and scatter plots. He was also responsible for demography analysis based on various room types by combining the map of Vancouver with a scatter plot.

Ryan was responsible for examining the three sampling methods applied in the project. He also performed the categorical analysis, creating the various contingency tables and performing the required statistical tests to determine the independence of the five tested variables with the variable price, and between minimum_nights and room_type.

Shora was responsible for producing the generalized linear model and multinomial regression model produced for the project. This included determining the primary descriptive variables, examining for interaction terms, testing for higher order terms, and testing the assumption of the linear model.

We all played an equal part in the creation of the powerpoint presentation.

Ryan was responsible for merging the final report together.

Gaurav was responsible for knitting the file and submitting the document.