PRÁCTICA2

Autor: Juan Camilo Rivera Palacio & Martin Loizate Sarrionandia

Diciembre de 2020

Índice general

1. Descripción del dataset.

¿Por qué es importante y qué pregunta/problema pretende Responder?

2. Integración y selección de los datos de interés a analizar.

Nuestro objetivo será comparar las casas en alquiler en la plataforma AirBnB de las tres ciudades mejor valoradas (según el ranking de bestcities.org) del mundo. Para ello, descargaremos el dataset de cada ciudad de la página (http://insideairbnb.com), y los uniremos.

```
library(readr)
set.seed(2)
London_detailed<- read_csv("DATOS/London_detailed.csv")
aux <- sample(London_detailed, length(London_detailed)/3)

write.csv(aux, "DATOS/London_detailed_.csv", row.names = F)
```

Añadiremos una columna "City" con el valor "London" para etiquetar los registros pertenecientes a London, para después poder compararlos con otras ciudades.

```
London_detailed$City = "London"
```

Haremos lo mismo con las otras dos ciudades (Paris y New York), cargaremos los dataset y les añadiremos una columna con el nombre de la ciudad:

```
Paris_detailed<- read_csv("DATOS/Paris_detailed.csv")
aux <- sample(Paris_detailed, length(Paris_detailed)/3)
write.csv(aux, "DATOS/Paris_detailed_.csv", row.names = F)
NY_detailed<- read_csv("DATOS/NY_detailed.csv")
aux <- sample(Paris_detailed, length(Paris_detailed)/3)
write.csv(aux, "DATOS/NY_detailed_.csv", row.names = F)
Paris_detailed$City = "Paris"
NY_detailed$City = "NY"
```

Una vez cargados los datos de las tres ciudades y etiquetados correctamente, los uniremos para formar el dataset completo con el que empezaremos a trabajar:

```
cat("London| columns:",ncol(London_detailed), "rows:", nrow(London_detailed))
```

```
## London| columns: 75 rows: 76984
```

```
cat("Paris| columns:",ncol(Paris_detailed), "rows:", nrow(Paris_detailed))
```

```
## Paris| columns: 75 rows: 66336
```

```
cat("NY| columns:",ncol(NY_detailed), "rows:", nrow(NY_detailed))
```

```
## NY| columns: 75 rows: 36923
```

```
data_detailed <- rbind(London_detailed, Paris_detailed, NY_detailed)
cat("Complete dataset| columns:",ncol(data_detailed), "rows:", nrow(data_detailed))
```

```
## Complete dataset| columns: 75 rows: 180243
```

Como vemos, nuestro dataset tiene 74 atributos con diferente información sobre el alojamiento. De todos estos atributos la mayoría no los necesitaremos para este análisis. Escogeremos los atributos que utilizaremos a lo largo del trabajo, y que nos serán más útiles para hacer comparaciones entre distintas ciudades:

```
library (dplyr)
data = select(data_detailed,'id', 'host_id', 'host_since',
'host_response_rate', 'host_acceptance_rate',
'host_is_superhost', 'host_listings_count',
'property_type', 'room_type', 'price', 'last_review',
'number_of_reviews', 'availability_30','review_scores_rating',
'City')
```

Hemos reducido nuestro dataset de 74 atributos a 15. A continuación haremos un análisis inicial sobre los datos, y los describiremos:

id: Es el identificador único de cada alojamiento.

host_id: Es el identificador de cada propietario. Un propietario puede poseer más de un alojamiento.

host_since: Es la fecha en el que se registró el propietario en la plataforma.

host_response_rate: El porcentaje de solicitudes respondidas por el propietario.

host_acceptance_rate: Es el porcentaje de propuestas que son aceptadas por el propietario.

host_is_superhost: Es un distintivo que la plataforma AirBnB dá a algunos propietarios, si cumplen unos requisitos específicos (mas información en: https://www.airbnb.com/help/article/829/how-do-i-become-a-superhost).

host_listings_count: Es la cantidad de alojamientos que tiene en la plataforma el propietario.

property_type: Es la forma de propiedad del alojamiento

room_type: Indica clase de alojamiento.

price: El precio por día del alojamiento.

last_review: Fecha de la última reseña (podremos utilizarlo para detectar alojamientos inactivos).

number_of_reviews: Cantidad de reseñas que tiene el alojamiento.

availability_30: Los días que está disponible en el plazo de un mes.

review_scores_rating: La puntuación de las reseñas.

City: Es la ciudad donde se encuentra el alojamiento (London, Paris o NY).

3. Limpieza de los datos.

¿Los datos contienen ceros o elementos vacíos? ¿Cómo gestionarías cada uno de estos casos?

3.1 Limpeza

Como vemos en los datos, algunos atributos no están en el formato adecuado para poderlos tratar. Por una parte, tenemos varios atributos del tipo *character* que deberemos de categorizar (property_type, room_type y City). Por otra parte, los atributos price, host_response_rate o host_acceptance_rate son del tipo *character* (contienen carácteres como % o $) y los tendremos que modificar para pasarlos a númerico, utilizando la libreria *stringr*. Además, transformaremos el atributo host_is_superhost para que, en vez de tomar valores TRUE o FALSE, sea una columna de 1 y 0.

```
library(stringr)

data$room_type <- as.factor(data$room_type)
data$property_type <- as.factor(data$property_type)
data$City <- as.factor(data$City)

#Dataprice
data$price <- data$price %>% str_extract_all("\\(?[0-9,.]+\\)?")%>%
gsub(",", "", .) %>% as.numeric()

#Host response rate
data$host_response_rate <- data$host_response_rate %>%
str_extract_all("\\(?[0-9,.]+\\)?") %>% gsub(",", "", .) %>% as.numeric()

#host acceptance rate
data$host_acceptance_rate <- data$host_acceptance_rate %>%
str_extract_all("\\(?[0-9,.]+\\)?") %>% gsub(",", "", .) %>% as.numeric()

data$host_is_superhost <- as.numeric(ifelse(data$host_is_superhost == 'TRUE', 1, 0))
```

```
sort(colMeans(is.na(data)), decreasing = TRUE)
```

```
##   host_response_rate host_acceptance_rate review_scores_rating
##          0.530372885          0.384148067          0.278607214
##          last_review           host_since    host_is_superhost
##          0.258079371          0.000166442          0.000166442
##  host_listings_count                   id              host_id
##          0.000166442          0.000000000          0.000000000
```

```
##        property_type            room_type                price
##          0.000000000          0.000000000          0.000000000
##    number_of_reviews      availability_30                 City
##          0.000000000          0.000000000          0.000000000
```

Como se puede ver en la tabla anterior, tenemos bastantes datos perdidos, sobre todo en las columnas host_response_rate (95596) y host_acceptance_rate (69240). En review_scores_rating (50217) y las_review (46517) también hay bastantes, y en una cantidad mucho menor en host_since, host_is_superhost y host_listings_count (30). El siguiente paso será analizar qué significa cada valor perdido y resolver que hacer con cada uno de ellos.

Filtramos y visualizaremos los 30 registros que tienen valores perdidos en *host_since*:

```
datos_perdidos <- data %>% filter(is.na(host_since))
dim(datos_perdidos)
```

```
## [1] 30 15
```

Como podemos ver, se trata de casos que, por algún error en la recolección de los datos, o por error de la plataforma, algunos datos se han perdido, ya que tienen valores perdidos en mínimo cinco atributos. En este caso, es una pequeña cantidad de registros (el 0.0001 %), por lo que la decisión será eliminarlos.

```
data = filter(data, !is.na(host_since))
```

Por otro lado, una cantidad bastante grande (53 % y 38 %) de los registros toman el valor *NA* en la columna host_response_rate y host_acceptance_rate. Se puede tratar de que, en algunos casos, esta información no es pública. La cantidad de valores perdidos es demasiado grande como para eliminar los registros, por lo que tendremos que prescindir de estos atributos para el análisis.

```
data = data[ -c(4:5) ]
```

Por último veremos que significan los valores perdidos de review_scores_rating. Hechando un vistazo sobre las filas que tienen este campo perdido, nos damos cuenta de que se trata de alojamientos que no tienen ninguna reseña (o tienen una única), por lo que no tienen ninguna puntuación, ni tampoco (lógicamente) ninguna fecha para *last_review*. Que un alojamiento no tenga ninguna reseña indica que no ha sido alquilada en ninguna ocasión (o en muy pocas ocasiones). También puede indicar que el alojamiento no esta "activo", es decir, aunque aparezca en la plataforma, en realidad el propietario no está pendiente de él.

Con todo ello, y tomando en cuenta de que se trata de una variable importante (tanto por lo que expresa como por lo que se puede deducir de él), se ha decidido eliminar los registros que tienen valores perdidos en el atributo *review_scores_rating*. Esto supone eliminar la cuarta parte de los datos, pero de esta forma se obtendrá un dataset más íntegro y más acorde con la realidad, ya que supone eliminar gran parte de los alojamientos "inactivos".

```
data = filter(data, !is.na(review_scores_rating))
dim(data)
```

```
## [1] 130016     13
```

Para finalizar con la preparación y limpieza de los datos, eliminaremos los alojamientos que no hayan tenido una reseña en los últimos 2 años, ya que se trata de alojamientos "inactivos". Si tuvieramos en cuenta estos alojamientos "inactivos", éstos alterarían el resultado real de la oferta o del precio en cada ciudad, por ejemplo.

```
data = data %>% filter(last_review >= as.Date("2019-01-01"))
dim(data)
```

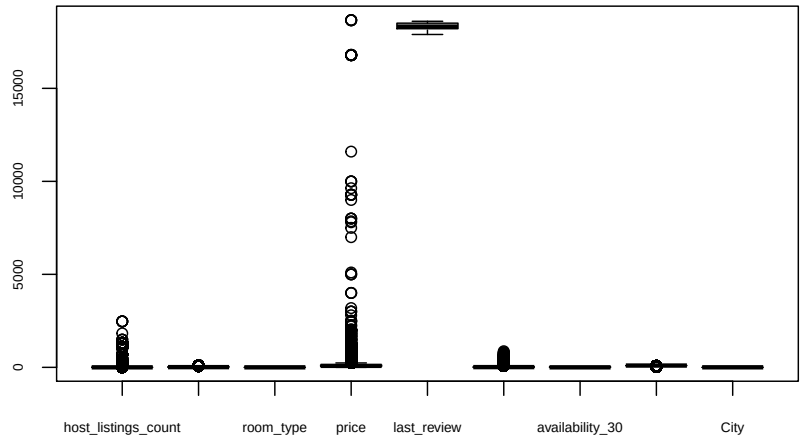
```
## [1] 98084    13
```

Una vez hecha la limpieza, nos queda un dataset de 98,084 registros con 13 atributos.

3.2 Outliers

Graficaremos boxplots por cada variable numérica:

```
par(cex.axis=0.6)
boxplot(data[5:13])
```



Como se puede ver en los boxplot, se han detectado bastantes outliers en las variables host_listings_count, price y number_of_reviews. Tendremos que ver

en cada variable la razón de estos outleirs, si son valores que entran dentro del rango lógico y tienen una explicación no los trataremos.

3.2.1 Outliers de host_listings_count

```
out <- filter(data_detailed, host_listings_count >= 1000)
select(out, host_name, host_listings_count)
```

```
## # A tibble: 1,484 x 2
##    host_name host_listings_count
##    <chr>                   <dbl>
##  1 Veeve                    1105
##  2 Veeve                    1105
##  3 Veeve                    1105
##  4 Veeve                    1105
##  5 Veeve                    1105
##  6 Veeve                    1105
##  7 Veeve                    1105
##  8 Veeve                    1105
##  9 Veeve                    1105
## 10 Veeve                    1105
## # ... with 1,474 more rows
```

Mostrando los registros de los outliers en host_listings_count se observa que los valores muy altos tienen sentido, ya que los nombres de los propietarios que tienen más de 1000 anuncios son de empresas inmoviliarios o hoteleras como Veeve (1105 anuncios), TraveNest (1004 anuncios), etc. Es por ello que no vamos a eliminar o alterar dichos registros.

```
out <- filter(data_detailed, host_listings_count >= 1000)
select(out, host_name, host_listings_count)
```

```
## # A tibble: 1,484 x 2
##    host_name host_listings_count
##    <chr>                   <dbl>
##  1 Veeve                    1105
##  2 Veeve                    1105
##  3 Veeve                    1105
##  4 Veeve                    1105
##  5 Veeve                    1105
##  6 Veeve                    1105
##  7 Veeve                    1105
##  8 Veeve                    1105
##  9 Veeve                    1105
## 10 Veeve                    1105
## # ... with 1,474 more rows
```

Analizando los registros con valores outlier para price, nos damos cuenta de que, aunque algunos anuncios son efectivamente de propiedades lujosas como Villas y tiene sentido que tengan precios tan altos, hay otros anuncios del tipo Break&Breakfast o habitación privada que de ninguna manera pueden valer dicha cantidad. Es decir, hay algunos anuncios "farsa" donde el precio no concuerda con lo que ofrece. Lógicamente, este tipo de anuncios "farsa" no tienen reseñas (o tienen 1 o 2, hechas por el propietario seguramente), y en eso nos basaremos para filtrarlos. Si no filtraramos este tipo de anuncios "farsa" alterarían los valores en el precio de nuestro dataset.

```
data <- filter(data, !(price >= 1000 & number_of_reviews<3))
```

3.2.2 Outliers de number_of_reviews

Por último, en cuanto a número de reseñas, es lógico que algunos valores estén bastante por encima de otros, entra dentro de la lógica por lo que no haremos ninguna transformación al respecto.

4. Análisis de los datos.

4.1. Selección de los grupos de datos.

Nuestro interés es comparar el comportamiento del precio de las viviendas de las tres ciudades, Paris, New York y Londres por tipo de vivienda: Entire home/apt, hotel room, private room y shared room. En total hay doce grupos organizados asi:

- Entire home/apt de Paris
- Entire home/apt de Londres
- Entire home/apt de Nueva York
- hotel room de Paris
- hotel room de Londres
- hotel room de Nueva York
- private room de Paris
- private room de Londres
- private room de Nueva York
- shared room de Paris
- shared room de Londres
- shared room de Nueva York

```
# Agrupación por pais y Entire Home
data_paris_EH<- data[data$City == "Paris" & data$room_type =='Entire home/apt',]
data_londres_EH <- data[data$City == "London" & data$room_type =='Entire home/apt',]
data_ny_EH <- data[data$City == "NY" & data$room_type =='Entire home/apt',]
```

```
# Agrupación por pais y Hotel room
data_paris_HR<- data[data$City == "Paris" & data$room_type =='Hotel room',]
data_londres_HR <- data[data$City == "London" & data$room_type =='Hotel room',]
data_ny_HR <- data[data$City == "NY" & data$room_type =='Hotel room',]


# Agrupación por pais y Private room
data_paris_PR<- data[data$City == "Paris" & data$room_type =='Private room',]
data_londres_PR <- data[data$City == "London" & data$room_type =='Private room',]
data_ny_PR <- data[data$City == "NY" & data$room_type =='Private room',]


# Agrupación por pais y Shared room
data_paris_SR<- data[data$City == "Paris" & data$room_type =='Shared room',]
data_londres_SR <- data[data$City == "London" & data$room_type =='Shared room',]
data_ny_SR <- data[data$City == "NY" & data$room_type =='Shared room',]
```

4.2. Comprobación de la normalidad y homogeneidad de la varianza.

Normalidad

Lo primero es conocer la cantidad de datos que tiene cada grupo para luego decidir que test se debe aplicar. Si el tamaño del grupo es menor a 5000 se aplicará el test de Shapiro-Wilk. Y para mayores a 5000 el test de Kolmogorov-Smirnov.

```
cantidad_datos_paris_EH <- length(data_paris_EH$price)
cantidad_data_londres_EH <- length(data_londres_EH$price)
cantidad_data_ny_EH<- length(data_ny_EH$price)
cantidad_data_paris_HR <- length(data_paris_HR$price)
cantidad_data_londres_HR<- length(data_londres_HR$price)
cantidad_data_ny_HR<- length(data_ny_HR$price)
cantidad_data_paris_PR<- length(data_paris_PR$price)
cantidad_data_londres_PR<- length(data_londres_PR$price)
cantidad_data_ny_PR<- length(data_ny_PR$price)
cantidad_data_paris_SR<- length(data_paris_SR$price)
cantidad_data_londres_SR<- length(data_londres_SR$price)
cantidad_data_ny_SR<- length(data_ny_SR$price)


Resumen <- data.frame( Variables =c("cantidad_datos_paris_EH",
"cantidad_data_londres_EH", "cantidad_data_ny_EH",
"cantidad_data_paris_HR", "cantidad_data_londres_HR",
"cantidad_data_ny_HR","cantidad_data_paris_PR",
"cantidad_data_londres_PR","cantidad_data_ny_PR",
```

```
"cantidad_data_paris_SR", "cantidad_data_londres_SR",
"cantidad_data_ny_SR"), Cantidad_datos = c(cantidad_datos_paris_EH,
cantidad_data_londres_EH, cantidad_data_ny_EH, cantidad_data_paris_HR,
cantidad_data_londres_HR,cantidad_data_ny_HR,cantidad_data_paris_PR,
cantidad_data_londres_PR,cantidad_data_ny_PR, cantidad_data_paris_SR,
cantidad_data_londres_SR, cantidad_data_ny_SR ))


print(Resumen)
```

```
##                    Variables Cantidad_datos
## 1   cantidad_datos_paris_EH          32590
## 2  cantidad_data_londres_EH          23886
## 3       cantidad_data_ny_EH          11000
## 4    cantidad_data_paris_HR            762
## 5  cantidad_data_londres_HR            405
## 6       cantidad_data_ny_HR            200
## 7    cantidad_data_paris_PR           3725
## 8  cantidad_data_londres_PR          16039
## 9       cantidad_data_ny_PR           8609
## 10   cantidad_data_paris_SR            181
## 11 cantidad_data_londres_SR            221
## 12      cantidad_data_ny_SR            392
```

De acuerdo a la tabla de arriba, el test **Shapiro-Wilk** se aplicará para los grupos:

- hotel room de Paris
- hotel room de Londres
- hotel room de Nueva York
- shared room de Paris
- shared room de Londres
- shared room de Nueva York
- private room de Paris

Y el test de **Kolmogorov-Smirnov** para los grupos restantes:

- Entire home/apt de Paris
- Entire home/apt de Londres
- Entire home/apt de Nueva York
- private room de Londres
- private room de Nueva York

A continuación está el **Shapiro-Wilk**

```
testsha_data_paris_SR <- shapiro.test(data_paris_SR$price)
testsha_data_ny_SR <- shapiro.test(data_ny_SR$price)
testsha_data_londres_SR <- shapiro.test(data_londres_SR$price)
```

```
testsha_data_paris_PR<- shapiro.test(data_paris_PR$price)
testsha_data_ny_HR<- shapiro.test(data_ny_HR$price)
testsha_data_londres_HR<- shapiro.test(data_londres_HR$price)
testsha_data_paris_HR<- shapiro.test(data_paris_HR$price)



resultados <- data.frame(Variable = c("data_paris_SR","data_ny_SR", "data_londres_SR","
"data_ny_HR","data_londres_HR","data_paris_HR"),
p_valor =c(testsha_data_paris_SR$p.value,
testsha_data_ny_SR$p.value,testsha_data_londres_SR$p.value,
testsha_data_paris_PR$p.value,testsha_data_ny_HR$p.value,
testsha_data_londres_HR$p.value, testsha_data_paris_HR$p.value))

print(resultados)
```

```
##          Variable      p_valor
## 1   data_paris_SR 1.988071e-21
## 2      data_ny_SR 3.721637e-39
## 3 data_londres_SR 1.671999e-28
## 4   data_paris_PR 4.112057e-83
## 5      data_ny_HR 4.489502e-21
## 6 data_londres_HR 1.371215e-24
## 7   data_paris_HR 4.395242e-38
```

Y el siguiente es el test **Kolmogorov-Smirnov**.

```
testks_data_paris_EH <- ks.test(data_paris_EH$price, "pnorm",
mean=mean(data_paris_EH$price), sd=sd(data_paris_EH$price))

testks_data_londres_EH <- ks.test(data_londres_EH$price, "pnorm",
mean=mean(data_londres_EH$price), sd=sd(data_londres_EH$price))

testks_data_ny_EH <- ks.test(data_ny_EH$price, "pnorm",
mean=mean(data_ny_EH$price), sd=sd(data_ny_EH$price))

testks_data_londres_PR <- ks.test(data_londres_PR$price, "pnorm",
mean=mean(data_londres_PR$price), sd=sd(data_londres_PR$price))

testks_data_ny_PR <- ks.test(data_ny_PR$price, "pnorm",
mean=mean(data_ny_PR$price), sd=sd(data_ny_PR$price))


resultados <- data.frame(Variable = c("data_paris_EH",
"data_londres_EH","data_ny_EH","data_londres_PR",
"data_ny_PR"), p_valor =c(testks_data_paris_EH$p.value,
testks_data_londres_EH$p.value,testks_data_ny_EH$p.value,
```

```
testks_data_londres_PR$p.value, testks_data_ny_PR$p.value))

print(resultados)
```

```
##          Variable p_valor
## 1   data_paris_EH       0
## 2 data_londres_EH       0
## 3      data_ny_EH       0
## 4 data_londres_PR       0
## 5      data_ny_PR       0
```

En ambos tests, la hipótesis nula de estas pruebas es que la población tiene una distribución normal. Por lo tanto, si el valor p es menor que el nivel alfa elegido, 0.05 en este caso, entonces se rechaza la hipótesis nula y hay evidencia de que los datos probados no se distribuyen normalmente. Por otro lado, si el valor p es mayor que el nivel alfa elegido, entonces la hipótesis nula (que los datos provienen de una población distribuida normalmente) no puede rechazarse.[1]

De acuerdo a lo anterior y a que los p valores de todos los grupos son menores a 0.05 entonces la distribucion de sus datos no son normales.

Homogeneidad de la varianza

En este caso utilizamos el test **Levene** [3]. La hipótesis nula es que las varianzas de la población son iguales (lo que se denomina homogeneidad de varianza u homocedasticidad). Si el valor p resultante es menor al nivel de significancia, en este caso 0.05, se rechaza la hipótesis nula de varianzas iguales y se concluye que existe una diferencia entre las varianzas en la población.

```
library (car)
```

```
## Loading required package: carData
```

```
##
## Attaching package: 'car'
```

```
## The following object is masked from 'package:dplyr':
##
##     recode
```

```
#Agrupar los datos en un data frame.

data_paris_EH_ <- rep("data_paris_EH", length(data_paris_EH$price))
data_londres_EH_ <- rep("data_londres_EH", length(data_londres_EH$price))
data_ny_EH_ <- rep("data_ny_EH", length(data_ny_EH$price))

data_paris_HR_ <- rep("data_paris_HR", length(data_paris_HR$price))
data_londres_HR_ <- rep("data_londres_HR", length(data_londres_HR$price))
data_ny_HR_ <- rep("data_ny_HR", length(data_ny_HR$price))

data_paris_PR_ <- rep("data_paris_PR", length(data_paris_PR$price))
```

```
data_londres_PR_ <- rep("data_londres_PR", length(data_londres_PR$price))
data_ny_PR_ <- rep("data_ny_PR", length(data_ny_PR$price))

data_paris_SR_ <- rep("data_paris_SR", length(data_paris_SR$price))
data_londres_SR_ <- rep("data_londres_SR", length(data_londres_SR$price))
data_ny_SR_ <- rep("data_ny_SR", length(data_ny_SR$price))

grupos <- c(data_paris_EH_,data_londres_EH_, data_ny_EH_, data_paris_HR_,
data_londres_HR_,data_ny_HR_,data_paris_PR_, data_londres_PR_, data_ny_PR_,
data_paris_SR_, data_londres_SR_, data_ny_SR_)

datos <- c(data_paris_EH$price,data_londres_EH$price, data_ny_EH$price,
data_paris_HR$price, data_londres_HR$price, data_ny_HR$price,
data_paris_PR$price, data_londres_PR$price,data_ny_PR$price,
data_paris_SR$price, data_londres_SR$price,data_ny_SR$price )

datos <- data.frame(grupos = grupos, datos = datos)
datos$grupos <- as.factor(datos$grupos)

test <- with(datos, leveneTest(datos,grupos))
test
```

```
## Levene's Test for Homogeneity of Variance (center = median)
##          Df F value    Pr(>F)
## group    11  48.304 < 2.2e-16 ***
##       97998
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Como el p valor es cercano a 0 y menor a 0.05 entonces se rechaza la hipotesis nula, es decir existe diferencias entre las varianzas de la población.

En conclusión, tenemos que todos los grupos no tiene una distribución normal y que tampoco cumplen con la propiedad de homogeneidad de la varianza.

4.3. Aplicación de pruebas estadísticas para comparar los grupos de datos.

4.3.1 Método 1. Pruebas de hipótesis

¿En Londres el tipo de alojamiento "Entire home/apt" es más costoso que en Paris?

Como se mencionó en el punto anterior, las distribuciones de los datos no son normales por lo tanto se utiliza el caso de contraste de hipotesis que se utiliza es **muestras grandes no normales**. Todos los grupos tiene tamaño mayor a 30

y por el teorema de limite central tenemos que el estadistico de contraste:

$z = \frac{\bar{x_1} - \bar{x_2}}{\sqrt{S_{\bar{x_1} - \bar{x_2}}}}$

Se distribuye N(0,1)

Para este caso, se define a μ_{paris} u $\mu_{londres}$ como la media de los precios de "Entire home/apt" para Paris y Londres respectivamente. Por lo tanto las hipotesis son:

H0: $\mu_{londres} = \mu_{paris}$

H1: $\mu_{londres} > \mu_{paris}$

```
#Desviación
desvi_paris <- sd(data_paris_EH$price)^2
desvi_londres <- sd(data_londres_EH$price)^2

#Estadistico contraste
denominador  <- sqrt(desvi_paris/(length(data_paris_EH$price))+
desvi_londres/(length(data_londres_EH$price)))
estadistico <- (mean(data_londres_EH$price) -
mean(data_paris_EH$price))/denominador
estadistico
```

```
## [1] 10.09513
```

```
#Calcular el p valor
pvalor = pnorm(10.11803, lower.tail = FALSE)
pvalor
```

```
## [1] 2.297861e-24
```

Como el p valor, cercano a 0, es menor que el nivel de significancia 0.05 entonces podemos rechazar la hipotesis nula es decir que en Londres es más costos que Paris "Entire home/apt".

¿Es más costoso compartir una habitación, share room,en Nueva York que en Paris?

Se define a μ_{ny} y μ_{paris} como la media de los precios de "share room" para New York y Pari, respectivamente. Por lo tanto las hipotesis son:

H0: $\mu_{nuevayork} = \mu_{paris}$

H1: $\mu_{nuevayork} > \mu_{paris}$

```
#Desviación
desvi_paris <- sd(data_paris_SR$price)^2
desvi_ny <- sd(data_ny_SR$price)^2

#Estadistico contraste
denominador  <- sqrt(desvi_paris/(length(data_paris_SR$price))+
desvi_londres/(length(data_ny_SR$pric)))
```

```
estadistico <- (mean(data_ny_SR$price) -
mean(data_paris_SR$price))/denominador
estadistico
```

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

```
#Calcular el p valor
pvalor = pnorm(0, lower.tail = FALSE)
pvalor
```

```
## [1] 0.5
```

Como el p-valor 0.5 es mayor a 0.05 entonces no podemos rechazar la hipotesis nula, es decir no es más costoso compartir una habitación en Nueva York que en Paris.

¿Es más costoso compartir una habitación, share room, en Nueva York que alquilar un private room en Londres?

Se define a μ_{ny} y μ_{paris} como la media de los precios de "share room" para New York y Paris respectivamente. Por lo tanto las hipótesis son:

H0: $\mu_{nuevayork} = \mu_{londres}$

H1: $\mu_{nuevayork} > \mu_{londres}$

```
#Desviación
desvi_londres <- sd(data_londres_PR$price)^2
desvi_ny <- sd(data_ny_SR$price)^2

#Estadistico contraste
denominador  <- sqrt(desvi_londres/(length(data_londres_PR$price))+
desvi_ny/(length(data_ny_SR$price)))
estadistico <- (mean(data_ny_SR$price) -
mean(data_londres_PR$price))/denominador
estadistico
```

```
## [1] 2.222418
```

```
#Calcular el p valor
pvalor = pnorm(2.35, lower.tail = FALSE)
pvalor
```

```
## [1] 0.009386706
```

Como el p-valor, 0.009 es menor a 0.05 entonces podemos rechazar la hipotesis nula, es decir es más costoso compartir una habitación, share room, en Nueva York que alquilar un private room en Paris.

4.2.1 Método 2. Correlación

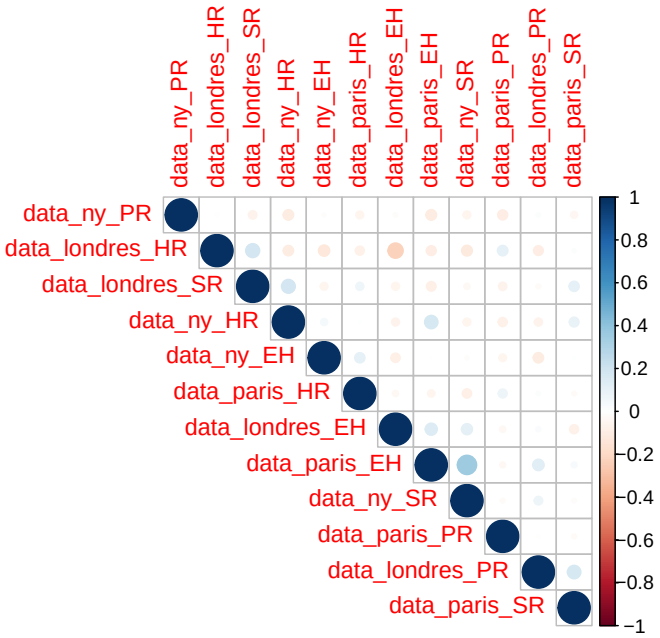
En esta sección se explorará si existe o no relación entre los precios de los alojamientos. Para esto se tomá aleatoriamente 100 datos por cada grupo de datos.

```
library(tidyr)
library(dplyr)
library(corrplot)
```

```
## corrplot 0.84 loaded
```

```
set.seed(1)

#data frame
datos_corr <- data.frame(data_paris_EH = sample(data_paris_EH$price,100),
data_londres_EH = sample(data_londres_EH$price,100),
data_ny_EH = sample(data_ny_EH$price,100),
data_paris_HR= sample(data_paris_HR$price,100),
data_londres_HR=sample(data_londres_HR$price,100),
data_ny_HR=sample(data_ny_HR$price,100),
data_paris_PR=sample(data_paris_PR$price,100),
data_londres_PR= sample(data_londres_PR$price,100),
data_ny_PR = sample(data_ny_PR$price,100),
data_paris_SR=sample(data_paris_SR$price,100),
data_londres_SR = sample(data_londres_SR$price, 100),
data_ny_SR = sample(data_ny_SR$price, 100))
```

```
#chart.Correlation(datos_corr, histogram=TRUE, pch=19)
M <-cor(datos_corr)
corrplot(M, type="upper", order="hclust")
```



De acuerdo al grafico, existe una relación directa entre los precios de Paris para "Entire home" y "private room" de Londres. De la misma manera que, "Hotel Room" de Nueva York y "Private Room" de Londres. En cambio, hay una relación inversa entre "Private room" de Nueva York y "Hotel Room" de Paris. Nótese que los datos fueron escogidos aleatoriamente y no se tuvo en cuenta si pertenecen a la misma fecha. El objetivo de esta correlación es solamente explorar esas posibles relaciones. Para futuros trabajos se pueden explicar estas relaciones de acuerdo a la situación economica de cada país y del sector hotelero de acuerdo a la fecha de recolección de los datos.

4.2.3 Método 3. Regresión

En esta sección se analizará el precio de los alojamientos para share room por ciudad de acuerdo a los otro tipos de alojamiento. Por ejemplo, si la ciudad es Paris entonces se pretende modelar el precio de share room de acuerdo al precio de los otros tipos de alojamiento: hotel room, private room, Entire home/apt.

```
#Paris
SR_Paris = lm(data_paris_SR ~data_paris_PR+
data_paris_HR + data_paris_EH,data=datos_corr)


#Londres
SR_Londres = lm(data_londres_SR ~data_londres_PR+
data_londres_HR + data_londres_EH,data=datos_corr)
```

```
#Nueva York
SR_NY = lm(data_ny_SR ~data_ny_PR+
data_ny_HR + data_ny_EH,data=datos_corr)


resultados <- data.frame(Model = c("Paris", "Londres", "Nueva York"),
R_2 =c(summary(SR_Paris)$r.squared, summary(SR_Londres)$r.squared,
summary(SR_NY)$r.squared))

resultados
```

```
##        Model         R_2
## 1      Paris 0.001791771
## 2    Londres 0.035612510
## 3 Nueva York 0.007783050
```

Como se muestra en la tabla anterior, los R cuadrados de los modelos son muy bajos para las tres ciudades, se puede concluir que el precio de los share room no depende del precio de los otros tipos de alojamientos solamente. Para modelar el precio se necesitan de más variables, que puedan describir mejor al share room, por ejemplo la cantidad de baños, si está cerca al centro, si tiene agua caliente y también tener consideraciones a nivel macroeconomico, por ejemplo el precio del dolar o euro, esto posiblemente influye en el precio.

5. Representación de los resultados a partir de tablas y gráficas.

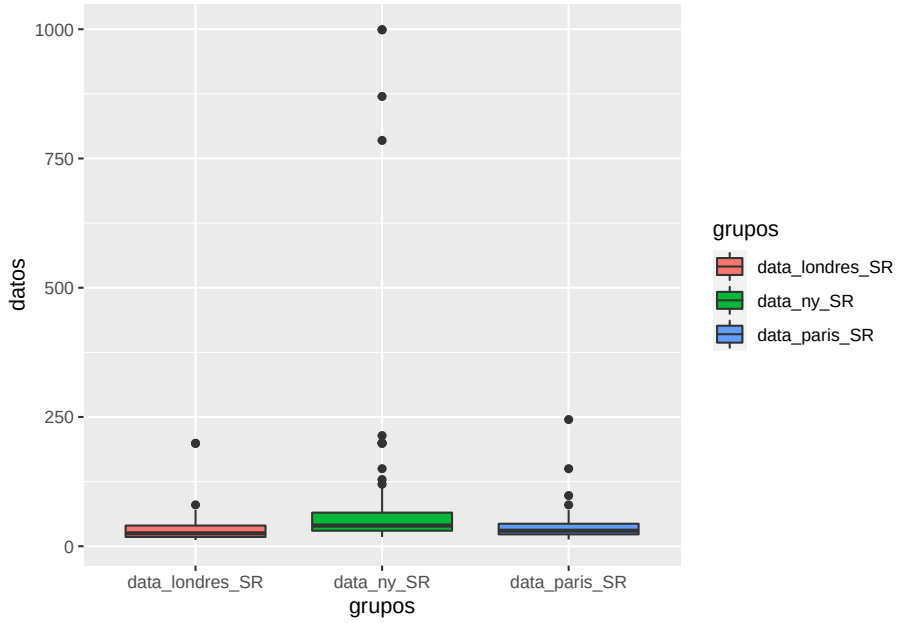
A continuación esta la tabla que muestra el precio promedio de cada uno de las ciudades.

```
resultados <- data.frame(Ciudad = c("Paris", "Londres", "Nueva York"),
PrecioPromedioPR = c(mean(datos_corr$data_paris_PR),
mean(datos_corr$data_londres_PR), mean(datos_corr$data_ny_PR) ),
PrecioPromedioHR = c(mean(datos_corr$data_paris_HR),
mean(datos_corr$data_londres_HR), mean(datos_corr$data_ny_HR) ),
PrecioPromedioEH = c(mean(datos_corr$data_paris_EH),
mean(datos_corr$data_londres_EH), mean(datos_corr$data_ny_EH)),
PrecioPromedioSR = c(mean(datos_corr$data_paris_SR),
mean(datos_corr$data_londres_SR), mean(datos_corr$data_ny_SR)))

resultados
```

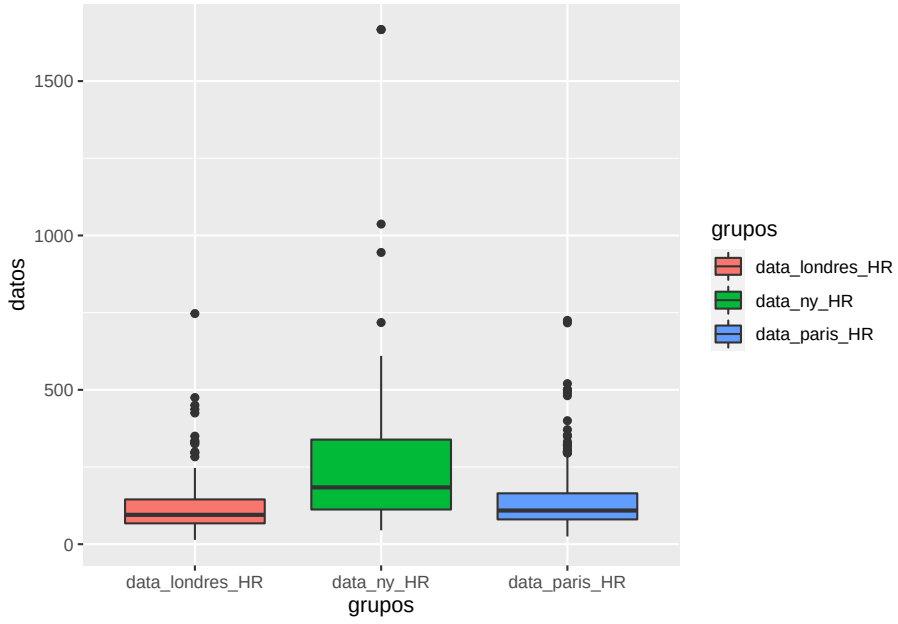
```
##       Ciudad PrecioPromedioPR PrecioPromedioHR PrecioPromedioEH
## 1      Paris            63.64           155.83           104.44
## 2    Londres            44.67           118.76           139.62
```

```
## 3 Nueva York           72.43           292.88           182.94
##   PrecioPromedioSR
## 1            41.01
## 2            44.80
## 3            72.70
```

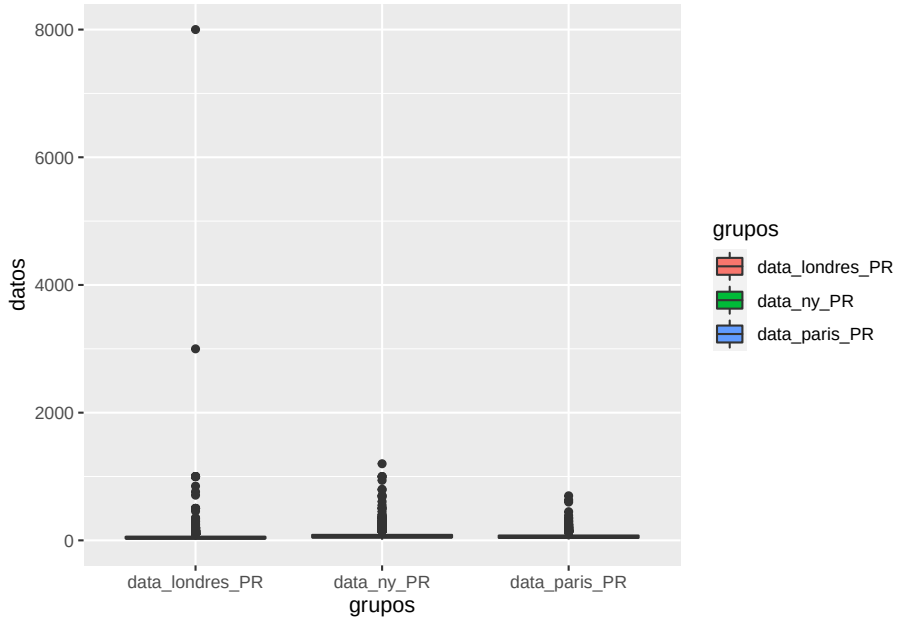
```
library(ggplot2)
 #SR
SR <- datos %>% filter(grupos == c("data_paris_SR", "data_londres_SR", "data_ny_SR" ) )
SR$grupos <- factor(SR$grupos )
SR %>% ggplot( aes(x=grupos, y=datos, fill=grupos)) + geom_boxplot()
```



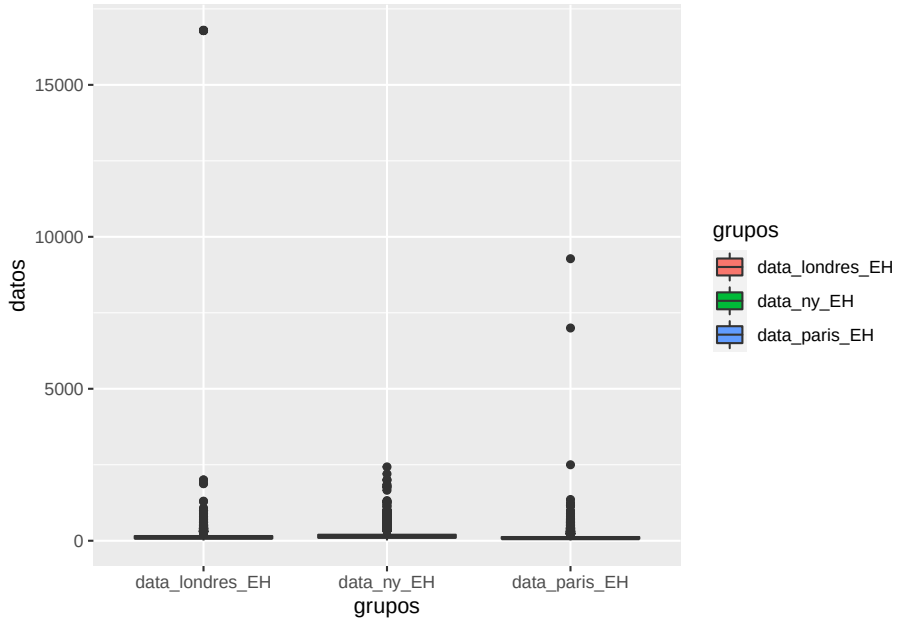

```
library(ggplot2)

HR <- datos %>% filter(grupos == c("data_paris_HR", "data_londres_HR", "data_ny_HR" ) )
HR$grupos <- factor(HR$grupos )
HR %>% ggplot( aes(x=grupos, y=datos, fill=grupos)) +geom_boxplot()
```



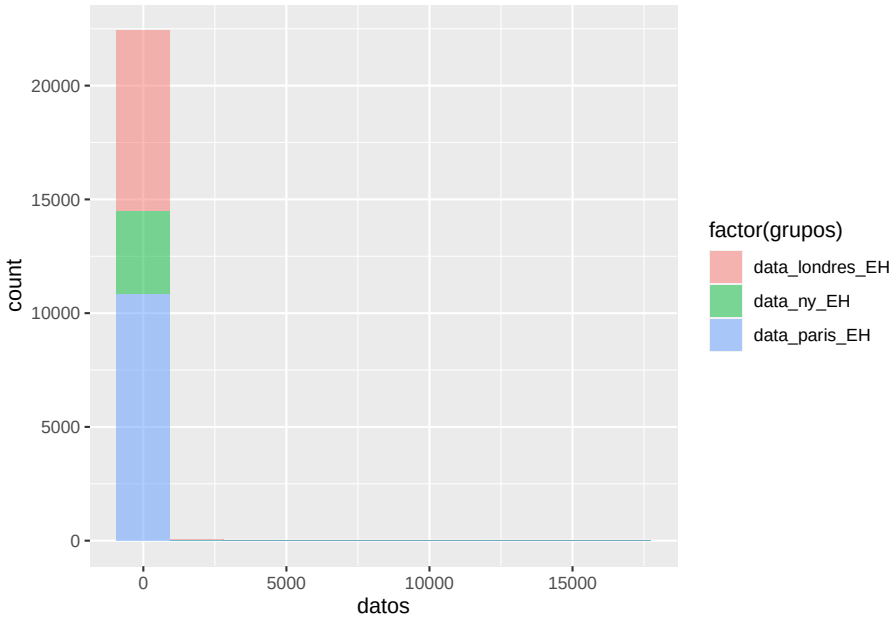

```
library(ggplot2)


PR <- datos %>% filter(grupos == c("data_paris_PR", "data_londres_PR", "data_ny_PR" ) )
PR$grupos <- factor(PR$grupos )
PR %>% ggplot( aes(x=grupos, y=datos, fill=grupos)) + geom_boxplot()
```




```
library(ggplot2)
EH <- datos %>% filter(grupos == c("data_paris_EH", "data_londres_EH", "data_ny_EH" ) )
EH$grupos <- factor(EH$grupos )
EH %>% ggplot( aes(x=grupos, y=datos, fill=grupos)) + geom_boxplot()
```



```
library(ggplot2)
EH <- datos %>% filter(grupos == c("data_paris_EH", "data_londres_EH", "data_ny_EH" ) )
ggplot(data = EH) + geom_histogram(aes(x=datos,fill=factor(grupos)),bins=10, position = "stack",a
```



ggplot(data = mtcars) + geom_histogram(aes(x=qsec,fill=factor(am)),bins=10, position = "stack",alpha = 0.5)

##6. Resolución del problema. A partir de los resultados obtenidos, ¿cuáles son las conclusiones? ¿Los resultados permiten responder al problema

Bibliografía

[1] https://en.wikipedia.org/wiki/Shapiro%E2%80%93Wilk_test [2] https://en.wikipedia.org/wiki/Levene%27s_test [3] https://en.wikipedia.org/wiki/Levene%27s_test