UPC
Universidad Peruana
de Ciencias Aplicadas

TA1

Administración de la Información

Profesor:

Reyes Silva, Patricia

Integrantes:

Gonzalez De La Cruz, Grober Ericson	**U201920609**
Mauricio Tecco, Cristian Alexander	**U201922705**
Omar Williams Fuentes Ortiz	**U20181a993**

Lima, 2021

Índice

1. Caso de análisis
2. Conjunto de datos (data set)
3. Análisis exploratorio de los datos
4. Conclusiones preliminares
5. Archivar y publicar
6. Bibliografía

1. Caso de análisis

Los datos utilizados para el estudio pertenecen a un hotel resort y a un hotel urbano de los cuales se desconoce el nombre, puesto que los datos relacionados con la identificación de los hoteles fueron eliminados. Se pudo conocer acerca de estos datos gracias a que la empresa editorial moderna Elseiver con sede principal en Amsterdam publicó un artículo en 2018 compartiendo los datos y haciendo un análisis de ellos.

Estos datos podrían ser usados para el desarrollo de modelos de predicción para clasificar la probabilidad de cancelación de una reserva de hotel, además serían de importancia y beneficio para las empresas relacionadas con el turismo y los viajes.

2. Conjunto de datos (data set)

Los datos están organizados en 31 variables que describen 40 060 observaciones del hotel resort y 79 330 del hotel urbano, los datos provienen de las reservas de ambos hoteles entre el 1 de julio de 2015 y el 31 de agosto de 2017.

3. Análisis exploratorio de datos

Cargar datos: Usamos la función read.csv para leer los datos y cargarlos en la tabla original y en la tabla datos, para poder modificar la tabla datos y poder hacer una comparación al final.

```
library(dplyr)
Tabla_Datos <- read.csv("hoteles.csv", header = TRUE, stringsAsFactors = FALSE)
Tabla_Original <- read.csv("hoteles.csv", header = TRUE, stringsAsFactors = FALSE)
View(Tabla_Datos)
```

Inspeccionar datos: Podemos ver la tabla, la cantidad de objetos y variables que contiene.

Data	
Tabla_Datos	119390 obs. of 32 variables
Tabla_Original	119390 obs. of 32 variables

Identificacion_datos_faltantes_NA.R | Tabla_Datos | Tabla_Original

Filter

	ï..hotel	is_canceled	lead_time	arrival_date_year	arrival_date_month	arrival_date_week_number	arrival_date_day_of_month	stays_in_weekend_nights
1	Resort Hotel	0	342	2015	July	27	1	NA
2	Resort Hotel	0	737	2015	July	27	1	0
3	Resort Hotel	0	7	2015	July	27	1	0
4	Resort Hotel	0	13	2015	July	27	1	0
5	Resort Hotel	0	14	2015	July	27	1	0
6	Resort Hotel	0	14	2015	July	27	1	0
7	Resort Hotel	0	0	2015	July	27	1	0
8	Resort Hotel	0	9	2015	July	27	1	0
9	Resort Hotel	1	85	2015	July	27	1	0
10	Resort Hotel	1	75	2015	July	27	1	0
11	Resort Hotel	1	23	2015	July	27	1	0
12	Resort Hotel	0	35	2015	July	27	1	0
13	Resort Hotel	0	68	2015	July	27	1	0
14	Resort Hotel	0	18	2015	July	27	1	0
15	Resort Hotel	0	37	2015	July	27	1	0
16	Resort Hotel	0	68	2015	July	27	1	0
17	Resort Hotel	0	37	2015	July	27	1	0
18	Resort Hotel	0	12	2015	July	27	1	0
19	Resort Hotel	0	0	2015	July	27	1	0
20	Resort Hotel	0	7	2015	July	27	1	0

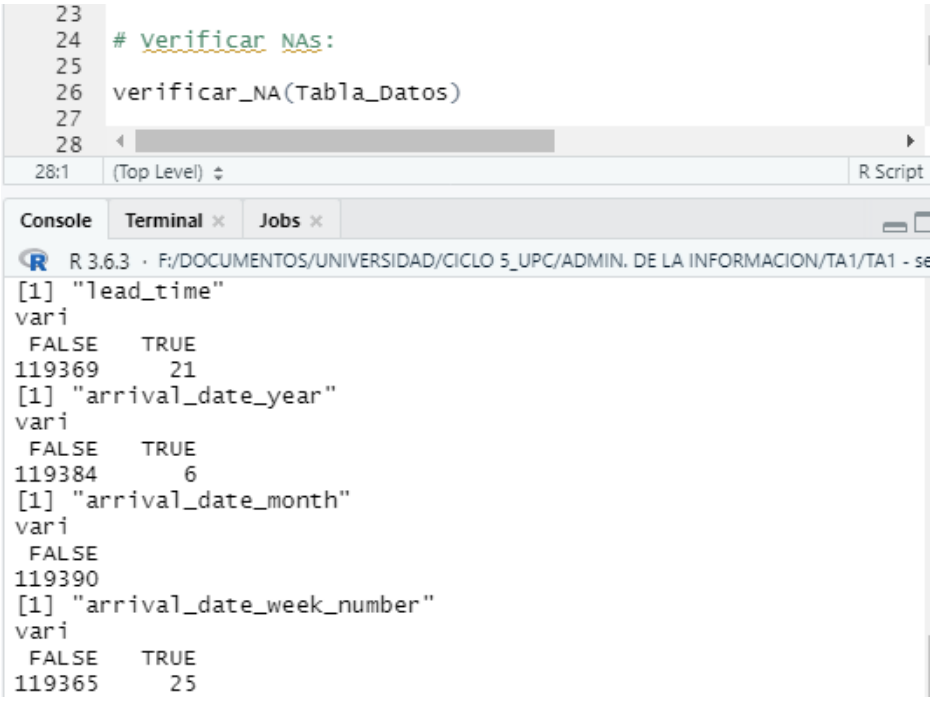
Preprocesar datos: Para preprocesar los datos aplicamos métodos para reemplazar los datos NA y los datos atípicos existentes en el dataset.

Primero aplicamos el método de reemplazo de los valores NA con la media de la población para las columnas que contuviesen datos numéricos.

Creamos una función para verificar los NA y otra para hacer el reemplazo con la media de la población.

```
# Funcion Verificar las columnas con valores NA
verificar_NA <- function(df){
  for(variable in names(df)) {
    print(variable)
    vari <- is.na(df[,c(variable)])
    print(table(vari))
  }
}

# Funcion Arreglar columna con reemplazo de datos NA con la media de la población
Reemplazar_media <- function(columna){
  colum <- Tabla_Datos[,c(columna)]
  Tabla_Datos[,c(columna)][colum == 0] <- NA
  temp.mean <- ifelse(is.na(colum), mean(colum, na.rm = TRUE), colum)
  temp.mean <- round(temp.mean, digits = 0)
  return(temp.mean)
}
```

Una vez que verificamos las columnas que contenían valores NA:



```
23
24  # Verificar NAs:
25
26  verificar_NA(Tabla_Datos)
27
28
28:1  (Top Level)  R Script

Console  Terminal  Jobs
R 3.6.3 · F:/DOCUMENTOS/UNIVERSIDAD/CICLO 5_UPC/ADMIN. DE LA INFORMACION/TA1/TA1 - se
[1] "lead_time"
vari
 FALSE   TRUE
119369     21
[1] "arrival_date_year"
vari
 FALSE   TRUE
119384      6
[1] "arrival_date_month"
vari
 FALSE
119390
[1] "arrival_date_week_number"
vari
 FALSE   TRUE
119365     25
```

Procedemos a ejecutar la función para modificar los datos NA por columna:

```
# Arreglar las columnas:

# Columna stays_in_weekend_nights
Tabla_Datos$stays_in_weekend_nights.mean <- Reemplazar_media("stays_in_weekend_nights")
Tabla_Datos$stays_in_weekend_nights <- NULL
names(Tabla_Datos)[32] <- "stays_in_weekend_nights"

# Columna lead_time
Tabla_Datos$lead_time.mean <- Reemplazar_media("lead_time")
Tabla_Datos$lead_time <- NULL
names(Tabla_Datos)[32] <- "lead_time"

# Columna arrival_date_year
Tabla_Datos$arrival_date_year.mean <- Reemplazar_media("arrival_date_year")
Tabla_Datos$arrival_date_year <- NULL
names(Tabla_Datos)[32] <- "arrival_date_year"

# Columna arrival_date_week_number
Tabla_Datos$arrival_date_week_number.mean <- Reemplazar_media("arrival_date_week_number")
Tabla_Datos$arrival_date_week_number <- NULL
names(Tabla_Datos)[32] <- "arrival_date_week_number"

# Columna stays_in_week_nights
Tabla_Datos$stays_in_week_nights.mean <- Reemplazar_media("stays_in_week_nights")
Tabla_Datos$stays_in_week_nights <- NULL
names(Tabla_Datos)[32] <- "stays_in_week_nights"

# Columna adults
Tabla_Datos$adults.mean <- Reemplazar_media("adults")
Tabla_Datos$adults <- NULL
names(Tabla_Datos)[32] <- "adults"

# Columna children
Tabla_Datos$children.mean <- Reemplazar_media("children")
Tabla_Datos$children <- NULL
names(Tabla_Datos)[32] <- "children"

# Columna babies
Tabla_Datos$babies.mean <- Reemplazar_media("babies")
Tabla_Datos$babies <- NULL
names(Tabla_Datos)[32] <- "babies"

# Columna babies
Tabla_Datos$days_in_waiting_list.mean <- Reemplazar_media("days_in_waiting_list")
Tabla_Datos$days_in_waiting_list <- NULL
names(Tabla_Datos)[32] <- "days_in_waiting_list"
```

Ahora tocaría modificar los valores NA con un valor simple aleatorio para las columnas con datos de variables categóricas.

Usamos 2 funciones mostradas en clase para reemplazar los valores NA de una columna con valores aleatorios de dicha columna, siempre y cuando no sean NA.

```
rand.valor <- function(x){
  faltantes <- is.na(x)
  tot.faltantes <- sum(faltantes)
  x.obs <- x[!faltantes]
  valorado <- x
  valorado[faltantes] <- sample(x.obs, tot.faltantes, replace = TRUE)
  return (valorado)
}

random.df <- function(df, cols){
  nombres <- names(df)
  for (col in cols) {
    nombre <- paste(nombres[col], "valorado", sep = ".")
    df[nombre] <- rand.valor(df[,col])
  }
  df
}
```

Modificamos la columna:

```
data.limpio <- random.df(Tabla_Datos, c(3))
Tabla_Datos$arrival_date_day_of_month <- data.limpio$arrival_date_month.valorado
rm(data.limpio)

verificar_NA(Tabla_Datos)
```

Ahora toca modificar los valores atípicos

Creamos una función para verificar si la columna contiene valores atipicos

```
# Funcion para verificar los utliers en las columnas
verificar_outliers <- function(df, df_names){
  for(variable in df_names) {
    print(variable)
    outliers.values <- boxplot(df[,c(variable)], main = paste(variable,"con Outliers"))$out
    outliers.values
  }
}

# Verificando las columnas con valores atipicos

verificar_outliers(Tabla_Datos, c("is_canceled"))
verificar_outliers(Tabla_Datos, c("adr"))
verificar_outliers(Tabla_Datos, c("lead_time"))
verificar_outliers(Tabla_Datos, c("stays_in_weekend_nights"))
verificar_outliers(Tabla_Datos, c("stays_in_week_nights"))
verificar_outliers(Tabla_Datos, c("adults"))
verificar_outliers(Tabla_Datos, c("children"))
verificar_outliers(Tabla_Datos, c("babies"))
```

Creamos la función para corregir los valores atípicos y corregimos

```
# Funcion para corregir los valores atipicos

fix_outliers <- function(x, removeNA = TRUE){
    quantiles <- quantile(x, c(0.05, 0.95), na.rm = removeNA)
    x[x<quantiles[1]] <- mean(x, na.rm = removeNA)
    x[x>quantiles[2]] <- median(x, na.rm = removeNA)
    x
}

# Corregir adr
par(mfrow = c(1,2))
verificar_outliers(Tabla_Datos, c("adr"))
boxplot(fix_outliers(Tabla_Datos$adr), main = "adr sin Outliers")

Tabla_Datos$adr <- fix_outliers(Tabla_Datos$adr)

# Corregir lead_time
par(mfrow = c(1,2))
verificar_outliers(Tabla_Datos, c("lead_time"))
boxplot(fix_outliers(Tabla_Datos$lead_time), main = "lead_time sin Outliers")

Tabla_Datos$lead_time <- fix_outliers(Tabla_Datos$lead_time)

# Corregir stays_in_weekend_nights
par(mfrow = c(1,2))
verificar_outliers(Tabla_Datos, c("stays_in_weekend_nights"))
boxplot(fix_outliers(Tabla_Datos$stays_in_weekend_nights), main = "stays_in_weekend_nights sin Outliers")

Tabla_Datos$stays_in_weekend_nights <- fix_outliers(Tabla_Datos$stays_in_weekend_nights)

# Corregir stays_in_week_nights
par(mfrow = c(1,2))
verificar_outliers(Tabla_Datos, c("stays_in_week_nights"))
boxplot(fix_outliers(Tabla_Datos$stays_in_week_nights), main = "stays_in_week_nights sin Outliers")

Tabla_Datos$stays_in_week_nights <- fix_outliers(Tabla_Datos$stays_in_week_nights)
```

Finalmente guardamos la data:

```
# GUARDAR DATA PROCESADA:
write.csv(Tabla_Datos,"hotel_bookings_miss_pre.csv")
```

Visualizar datos:

```
#COMPARACION DE LOS DATOS DESPUES DE MODIFICAR LOS VALORES NA CON LOS VALORES RANDOM

verificar_NA(Tabla_Original, "arrival_date_day_of_month")
verificar_NA(Tabla_preprocesada, "arrival_date_day_of_month")
```

```
# Funcion Verificar las columnas con valores NA
verificar_NA <- function(df, columna){
  print(columna)
  vari <- is.na(df[,c(columna)])
  print(table(vari))
}

verificar_NA(Tabla_Original, "adults")
verificar_NA(Tabla_preprocesada, "adults")


verificar_NA(Tabla_Original, "children")
verificar_NA(Tabla_preprocesada, "children")
```

Aquí modificamos los valores NA con el método de la media poblacional con el metodo is.na, para asi después compararlos

```
> verificar_NA(Tabla_Original, "adults")
[1] "adults"
vari
 FALSE   TRUE
119378     12
> verificar_NA(Tabla_preprocesada, "adults")
[1] "adults"
vari
 FALSE
119390
```

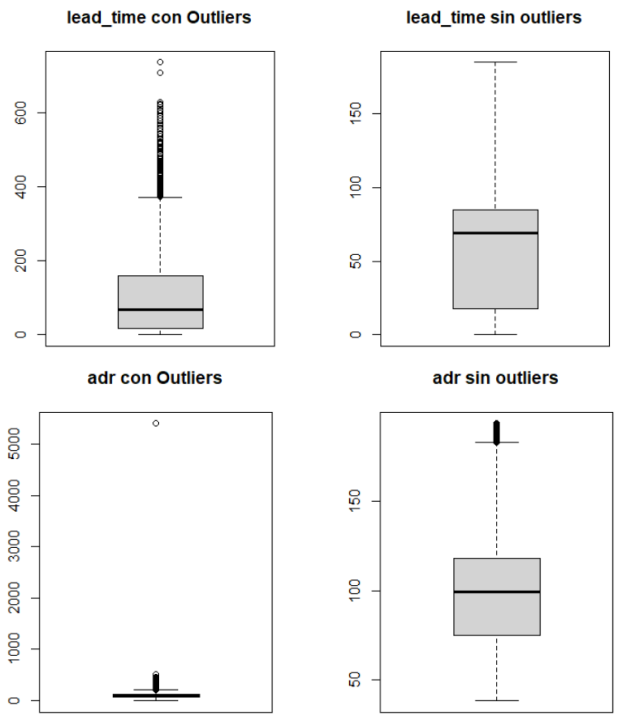
```
> verificar_NA(Tabla_Original, "children")
[1] "children"
vari
 FALSE   TRUE
119386      4
> verificar_NA(Tabla_preprocesada, "children")
[1] "children"
vari
 FALSE
119390
```

```r
library(dplyr)
#LEYENDO LA TABLA PREPROCESADA Y LA ORIGINAL PARA SU COMPARACION
Tabla_preprocesada <- read.csv("hotel_bookings_miss_pre.csv", header = TRUE, stringsAsFactors = FALSE)
Tabla_Original <- read.csv("hotel_bookings_miss.csv", header = TRUE, stringsAsFactors = FALSE)
View(Tabla_preprocesada)
```

```
> verificar_NA(Tabla_Original, "arrival_date_day_of_month")
[1] "arrival_date_day_of_month"
vari
 FALSE   TRUE
119383      7
> verificar_NA(Tabla_preprocesada, "arrival_date_day_of_month")
[1] "arrival_date_day_of_month"
vari
 FALSE
119390
```

Aqui se modificaron los valores NA con unos aleatorios de la tabla, teniendo en cuenta que estos no pueden ser random otra vez

```r
#COMPARACION DE LOS DATOS DESPUES DE ELIMINAR LOS DATOS ATIPICOS DE LA TABLA ADR Y LEAD TIME

verificar_outliers <- function(df, df_names){
  for(variable in df_names) {
    print(variable)
    outliers.values <- boxplot(df[,c(variable)], main = paste(variable,"con Outliers"))$out
    outliers.values
  }
}
par(mfrow = c(1,2))

verificar_outliers(Tabla_Original, c("adr"))
boxplot((Tabla_preprocesada$adr), main = "adr sin outliers")



verificar_outliers(Tabla_Original, c("lead_time"))
boxplot((Tabla_preprocesada$lead_time), main = "lead_time sin outliers")
```



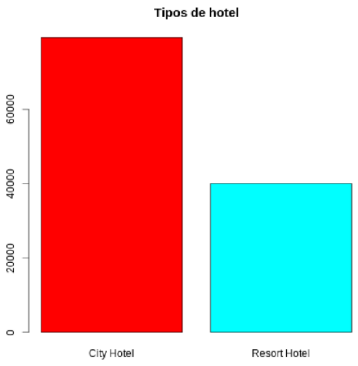
Aquí se comparan los datos tras de eliminar los datos atípicos de las tablas mostradas, para ello verificamos los datos outliers en las tablas y los eliminamos, esta vendría a ser la tabla preprocesada con la cual compararemos los datos anteriores

4. Conclusiones preliminares

A. ¿Qué tipo de hotel prefiere la gente?

El tipo de hotel que la gente prefiere es City Hotel

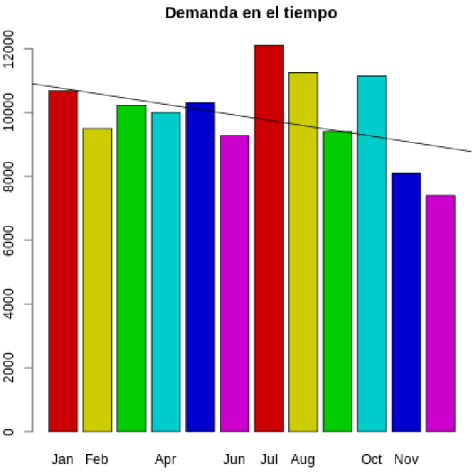
```r
# a. ¿Cuántas reservas se realizan por tipo de hotel?
# o ¿Qué tipo de hotel prefiere la gente?

library(dplyr)

hotel_data <- read.csv("hotel_bookings_miss_pre.csv")

hotel_table <- table(hotel_data$hotel);

print(hotel_table)

hotel <- barplot(hotel_table,
    main="Tipos de hotel", col=rainbow(2))

```



B. ¿Está aumentando la demanda con el tiempo?

La demanda de reservas se reduce según los meses.

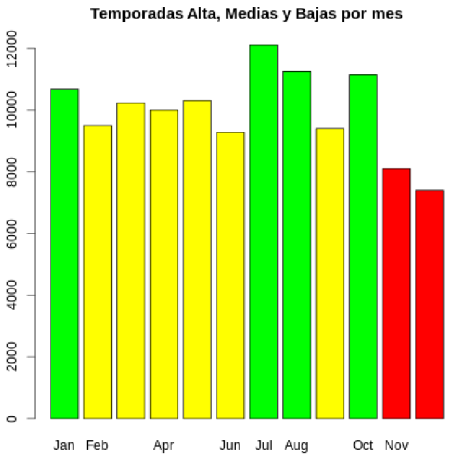
```r
# b. ¿Está aumentando la demanda con el tiempo?

library("dplyr")
library("lubridate")

# Cargar Datos
hotel_data <- read.csv("hotel_bookings_miss_pre.csv")

# Meses de cancelacion de Reservas
m_x <- month(as.POSIXlt(
    hotel_data$reservation_status_date,
    format="%d/%m/%Y"))
hotel_data$mon <- m_x

hotel_data.grp <- hotel_data %>%
    group_by(mon) %>%
    summarise(n = n())

regresion <- lm(hotel_data.grp$n ~ hotel_data.grp$mon,
    col="red")

barplot(hotel_data.grp$n, names.arg=month.abb,
    col=rainbow(6, v=.8))
abline(regresion)

```



C. ¿Cuándo se producen las temporadas de reservas: Alta, media y baja?

Se producen reservas bajas en noviembre y diciembre, reservas medias en febrero, abril, marzo, junio, y septiembre, y reservas altas en enero, julio, agosto y octubre.

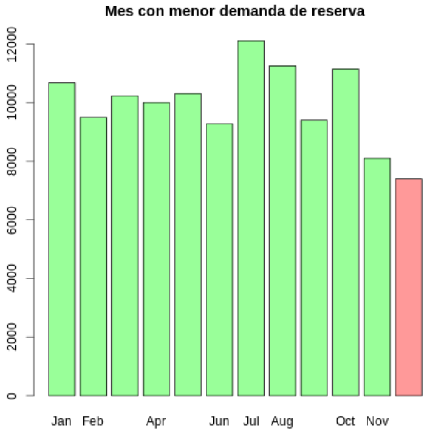
```r
# c. ¿Cuándo se producen las temporadas de
#  reservas: alta, media y baja?

library("dplyr")
library("lubridate")

# Cargar Datos
hotel_data <- read.csv("hotel_bookings_miss_pre.csv")

# Cuando se producen las temporadas de reserva
m_x <- month(as.POSIXlt(hotel_data$reservation_status_date,
    format="%d/%m/%Y"))
hotel_data$mon <- m_x

hotel_data.grp <- hotel_data %>%
  group_by(mon) %>%
  summarise(n = n())
hotel_menor_d <- min(hotel_data.grp$n)
hotel_mayor_d <- max(hotel_data.grp$n)
d = (hotel_mayor_d - hotel_menor_d)/3

colors_menor <- (hotel_data.grp$n >= hotel_menor_d &
    hotel_data.grp$n < hotel_menor_d + d)
colors_mayor <- (hotel_data.grp$n <= hotel_mayor_d &
    hotel_data.grp$n > hotel_mayor_d - d)
colors_medio <- ( hotel_data.grp$n < hotel_mayor_d - d &
    hotel_data.grp$n > hotel_menor_d + d)
colores <- ifelse(colors_mayor , "green" ,
    ifelse (colors_menor, "red" ,
    ifelse(colors_medio,"yellow", "gray")))

barplot(hotel_data.grp$n, names.arg=month.abb,
    col = colores )
```



D. ¿Cuándo es menor la demanda de reservas?

La demanda es menor en el mes de diciembre.

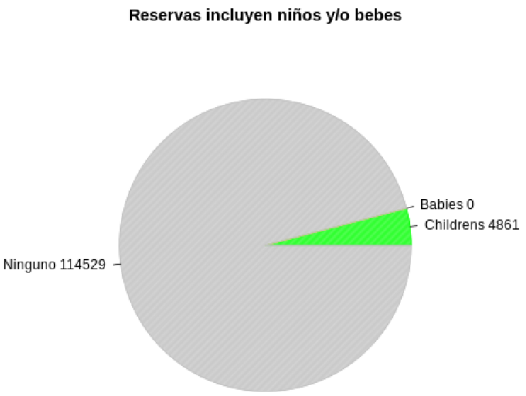
```r
# d. ¿Cuándo es menor la demanda de reservas?

library("dplyr")
library("lubridate")
library("purrr")

# Cargar Datos
hotel_data <- read.csv("hotel_bookings_miss_pre.csv")

# Meses de cancelacion de Reservas
m_x <- month(as.POSIXlt(hotel_data$reservation_status_date,
    format="%d/%m/%Y"))
hotel_data$mon <- m_x

hotel_data.grp <- hotel_data %>%
    group_by(mon) %>%
    summarise(n = n())
hotel_menor_d <- min(hotel_data.grp$n)

colors <- hotel_data.grp$n == hotel_menor_d
colors <- ifelse(colors, "#ff9999", "#99ff99")

barplot(hotel_data.grp$n, names.arg=month.abb, col=colors)
```



E. ¿Cuántas reservas incluyen niños y/o bebés?

El número de reservas que incluyen niños y bebés son 4061.

```r
# e. ¿Cuántas reservas incluyen niños y/o bebes?

hotel_data <- read.csv("hotel_bookings_miss_pre.csv")

hotel_data.babies <- hotel_data[hotel_data$babies>0,]
hotel_data.babies <- hotel_data.babies[
        is.na(hotel_data.babies$children) == 0,]

hotel_data.children <- hotel_data[hotel_data$children>0,]
hotel_data.children <- hotel_data.children[
        is.na(hotel_data.children$children) == 0,]

hotel_data.all <- hotel_data[
        (hotel_data$children==0)&(hotel_data$babies==0),]
hotel_data.all <- hotel_data.all[
        (is.na(hotel_data.all$children)|
        is.na(hotel_data.all$babies))==0,]

n_children <- nrow(hotel_data.children)
n_babies <- nrow(hotel_data.babies)
n_all <- nrow(hotel_data.all)

colors <- c("green", "yellow", "gray")
labels <- c("Childrens", "Babies", "Ninguno")
values <- c(n_children, n_babies, n_all)
etiquetas <- paste0(labels, " ", values)

pie(values, labels = etiquetas, density = 50, col = colors)
```



F. ¿Es importante contar con espacios de estacionamiento?

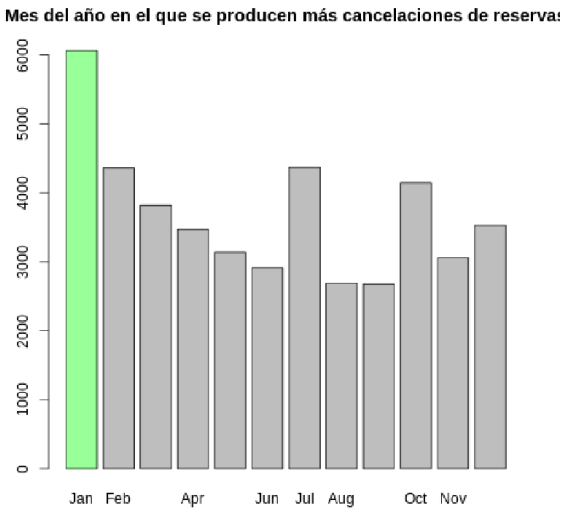
Solo en 7 '383 reservas de 111' 974 reservas se consideró importante los espacios de estacionamiento.

```r
# f. ¿Es importante contar con espacios de estacionamiento?

hotel_data <- read.csv("hotel_bookings_miss_pre.csv")

hotel_data.tr <- hotel_data[hotel_data
    $required_car_parking_spaces==1,]
hotel_data.tr <- hotel_data.tr[
    is.na(hotel_data.tr$required_car_parking_spaces) == 0,]

hotel_data.fl <- hotel_data[
    hotel_data$required_car_parking_spaces==0,]
hotel_data.fl <- hotel_data.fl[
    is.na(hotel_data.fl$required_car_parking_spaces) == 0,]

n_tr <- nrow(hotel_data.tr)
n_fl <- nrow(hotel_data.fl)

colors <- c("green", "gray")
labels <- c("Important", "No Important")
values <- c(n_tr, n_fl)
etiquetas <- paste0(labels, " ", values)

pie(values, labels = etiquetas, col = colors)
```

G. ¿En qué meses del año se producen más cancelaciones de reservas?

El mes en el que se produjeron más cancelaciones de reservas se dió en enero.

```r
# g. ¿En qué meses del año se producen
# más cancelaciones de reservas?

library("dplyr")
library("lubridate")

# Cargar Datos
hotel_data <- read.csv("hotel_bookings_miss_pre.csv")

# Meses de cancelacion de Reservas
hotel_data.sts <- hotel_data[hotel_data$is_canceled == 1,]
m_x <- month(as.POSIXlt(
    hotel_data.sts$reservation_status_date,
    format="%d/%m/%Y"))
hotel_data.sts$mon <- m_x

hotel_data.grp <- hotel_data.sts %>%
    group_by(mon) %>%
    summarise(n = n())
hotel_data.grp$monName <- month.abb[hotel_data.grp$mon]

# Obtener Mes con mayores cancelamientos
hotel_data.maximo <- max(hotel_data.grp$n)
hotel_data.r <- hotel_data.grp[
    hotel_data.grp$n == hotel_data.maximo, ]

colors <- hotel_data.grp$n == hotel_data.maximo
colors <- ifelse(colors, "#99ff99", "gray")

# Graficar Reservas canceladas por Meses
barplot(hotel_data.grp$n,
    names.arg=hotel_data.grp$monName, col=colors)
```



6. Bibliografía

Nuno, A. & De Almeida, A. & Nunes, L (2018). Hotel booking demand datasets. *Data in Brief 22*, 41-49. doi: 10.1016/j.dib.2018.11.126

ELSEVIER (s.f.). About Elsevier. Ámsterdam:Elsevier. Recuperado de: https://www.elsevier.com/about [Consulta: 1 de octubre de 2021]