

Universidad Peruana de Ciencias Aplicadas

Escuela de Postgrado

Maestría en Data Science



Trabajo Final

Revision de cuatro casos de uso Grupo 5

Curso: Gestion de Datos

Profesor: Oscar Ramos

Presentado por: Ortega Baca, Aldo Javier

González Victoria, Jorge Wilfredo

Berríos Aguilar, Fabrizio Franco Antonio

Aybar Coronel, Jorge Luis

Lima – 2022

Indices

Introducción.....	3
Primer Caso Estudio de evolución de la Mortalidad de Neonatos y la importancia del lavado de manos antes de atender a los pacientes en el XIX	4
Segundo Caso: Análisis del consumo alcohol en Rusia.....	4
Tercer Caso: Control de Pesos en la Crianza de Aves.....	4
Cuarto Caso Análisis de Outliers de precios en autos usados de la marca BMW	5
Metodología y Resultados.	5
Primer Caso Estudio de evolución de la Mortalidad de Neonatos y la importancia del lavado de manos antes de atender a los pacientes en el XIX	7
Segundo Caso: Análisis del consumo alcohol en Rusia.....	11
Tercer Caso: Control de Pesos en la Crianza de Aves.....	27
Cuarto Caso Análisis de Outliers de precios en autos usados de la marca BMW	35
Conclusiones y Recomendaciones	42
Bibliografía	43
Anexos	43

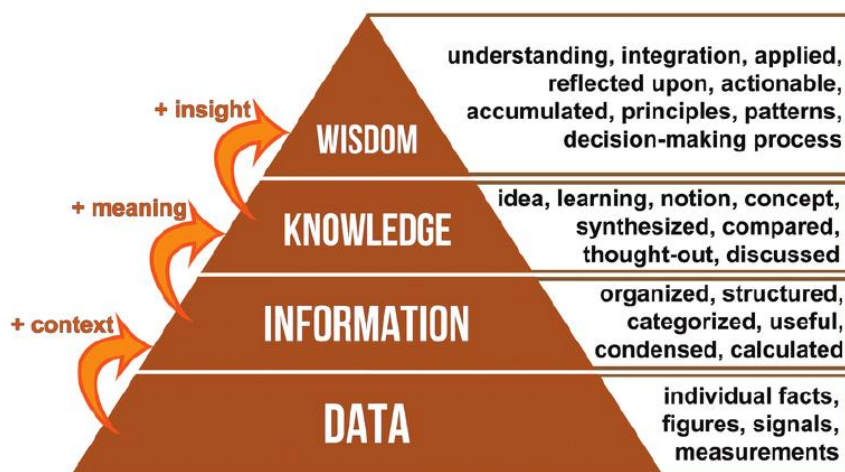
Introducción

La gestión de datos a lo largo del tiempo ha tenido una constante evolución en conformidad a como se han desarrollado los diferentes avances tecnológicos que permiten almacenar, administrar, transmitir, preparar, etiquetar, limpiar la data para así poder preparar modelos que permitan diseñar y entregar modelos estadísticos de apoyo a la decisión basado en hechos facticos soportados por información de valor.

Como bien se sabe la data por sí sola no tiene valor si es que no se aplica un contexto el cual a través de un procesamiento se convierte en información.

Dentro de algunas herramientas y marcos de trabajo muy comúnmente utilizadas podemos mencionar lo que serían utilizados en el presente trabajo de estudio

- DAMA DMBOK
- CRISP-DM
- IBM
- SaS
- Rstudio
- Shiny App
- Python
- Collab
- Atlas Mongo DB
- AWS, GCP, Azure
- AWS MySQL EC2 instance
- AWS RDS
- GITHUB



Pirámide DIKW

En el presente documento se expondrá los pasos para evaluar 4 casos de uso en los cuales se aplicaran las diferentes buenas prácticas y esquemas de análisis y métodos de visualización vistas en el curso que permitirán obtener un nivel de entendimiento de la dinámica de los mismos como parte de las actividades de comprensión de los datos y que podrían ser maximizadas luego con otras técnicas de depuración, preparación y entrenamiento de datos que serán contemplados en cursos posteriores. Claro esta este curso se enfoca en explorar los diferentes esquemas de visualización de los datos.

Primer Caso Estudio de evolución de la Mortalidad de Neonatos y la importancia del lavado de manos antes de atender a los pacientes en el XIX

El doctor Semmelweis a mediados de 1840 observo que un 10% de los recién nacidos dos clínicas 1 y 2 fallecían por fiebre relacionada a infecciones y él recomendó que los doctores debían lavarse las manos antes de atender a los pacientes pues pensaba que por esas razones las infecciones proliferaban durante el parto. Claro está eso es un hecho factico en el XXI la importancia de las condiciones asépticas, pero en ese contexto del XIX no había forma de validar esta afirmación del doctor tenga influencia positiva en la reducción de muertes en neonatos durante el parto.

Por estas razones utilizando la data que registraba las muertes de neonatos anual y mensual registradas desde 1841 a 1846.

Segundo Caso: Análisis del consumo alcohol en Rusia

Este caso de uso se utilizan la data de venta histórica de una cadena de tiendas en Rusia para diferentes productos descritos en ingles debido al contenido de la data: wine, beer, vodka, champagne and brandy

La compañía ejecuto una campana exitosa en la región de Saint Petersburg y la idea es poder replicarla en las 10 regiones que tengan comportamiento o hábitos de compra similares porque el área de marketing no tiene recursos para implementar campanas en todas las regiones.

En este caso se utilizaran diversos esquemas de visualización, gestión de datos perdidos, análisis de valores atípicos que permitan soportar la decisión de donde convendría aplicar la campaña de marketing

Tercer Caso: Control de Pesos en la Crianza de Aves

El contexto de este caso está en la industria avícola de la empresa peruana Grupo Santa Elena S.A. que tiene como principal actividad la crianza de aves de corral, específicamente

pollos engorde. El proceso de crianza de pollos busca lograr el máximo de kilogramos de carne de pollo y para ello debe cuidar sus parámetros productivos como la mortalidad, la nutrición y el peso corporal del ave; todos ellos obedecen a estándares dados por la genética del ave, sistema de crianza y sexo.

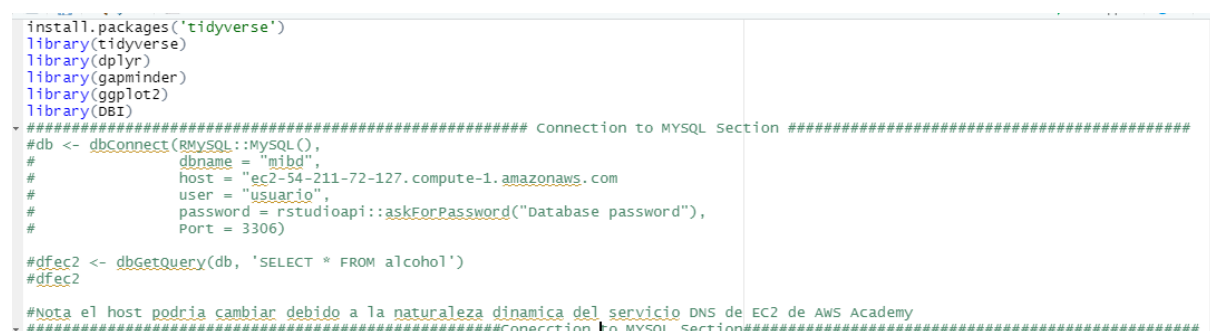
Cuarto Caso Análisis de Outliers de precios en autos usados de la marca BMW

Metodología y Resultados.

Para todos los casos expuestos en el siguiente documento se utilizará un servicio en MySQL instalado en una instancia de AWS EC2 estableciendo la conexión pertinente al entorno de desarrollo y visualización de resultados RStudio.

Específicamente, en el tercer caso de estudio se utilizará el servicio Atlas de Mongo DB con conexión al servicio de Google Collab en donde se realiza el análisis y visualización de los esquemas correspondientes

El esquema utilizado para conectar Rstudio con la instancia MySQL EC2 de AWS se muestra en la siguiente imagen.



```
install.packages('tidyverse')
library(tidyverse)
library(dplyr)
library(gapminder)
library(ggplot2)
library(DBI)

##### Connection to MYSQL Section #####
#db <- dbConnect(RMySQL::MySQL(),
#               dbname = "midb",
#               host = "ec2-54-211-72-127.compute-1.amazonaws.com",
#               user = "usuario",
#               password = rstudioapi::askForPassword("Database password"),
#               Port = 3306)

#dfec2 <- dbGetQuery(db, 'SELECT * FROM alcohol')
#dfec2

#Nota el host podría cambiar debido a la naturaleza dinámica del servicio DNS de EC2 de AWS Academy
#####Conexión to MYSQL Section#####
```

Ilustración 1 : Código de Conexión hacia MySQL/EC2 desde Rstudio

Asimismo, en la siguiente imagen se demuestra la creación de las tablas en la base de datos midb en MySQL que serán utilizados en todos los casos excepto el tercero.

Las tablas creadas y ser utilizadas son las siguientes:

- Monthly.
- Yearly
- Alcohol

- BMW

```

45 updates can be applied immediately.
To see these additional updates run: apt list --upgradable

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

Last login: Fri Apr 29 03:51:07 2022 from 179.6.24.192
ubuntu@ip-172-31-50-100:~$ sudo mysql
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 8
Server version: 8.0.28-0ubuntu0.20.04.3 (Ubuntu)

Copyright (c) 2000, 2022, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> use mibd;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> show tables;
+-----+
| Tables_in_mibd |
+-----+
| Boxoffice      |
| Monthly        |
| Movies         |
| Yearly         |
| alcohol        |
| bmw            |
| north_american_cities |
+-----+
7 rows in set (0.01 sec)

mysql>

```

Ilustración 2 : Validación MYSQL Database on AWS EC2 Service

En el tercer caso de uso se utilizara la Bases de Datos BD_Granjas y la colección pesos de mongo DB como se observa en la siguiente ilustración

Base de datos en Atlas Mongo DB

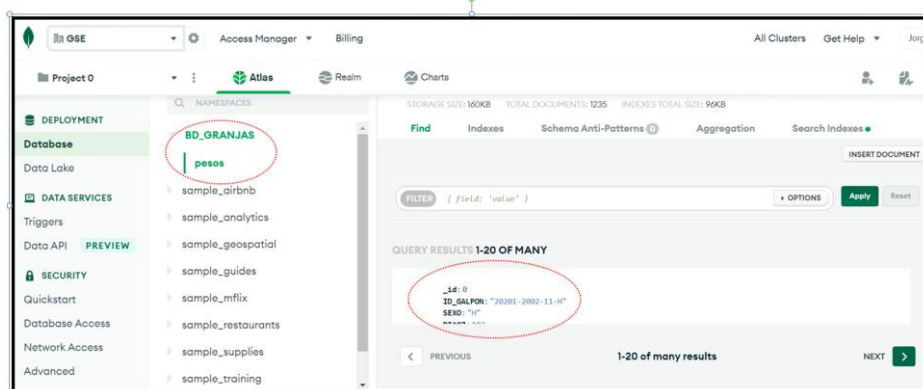
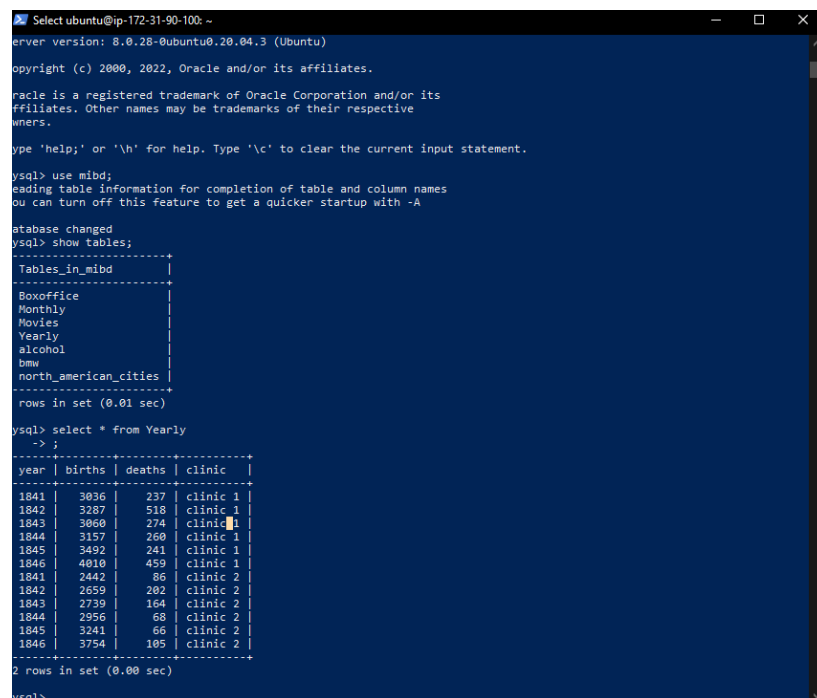


Ilustración 3: Servicio Atlas Mongo DB para el Tercer Caso

Primer Caso Estudio de evolución de la Mortalidad de Neonatos y la importancia del lavado de manos antes de atender a los pacientes en el XIX

Para este caso de estudio los datos que serán utilizados en el modelo de evaluación visual pertenecen a las tablas Yearly and Monthly del entorno MySQL/EC2 como se muestra en la siguiente figura



```
Selectubuntu@ip-172-31-90-100: ~
mysql version: 8.0.28-0ubuntu0.20.04.3 (Ubuntu)

Copyright (c) 2000, 2022, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

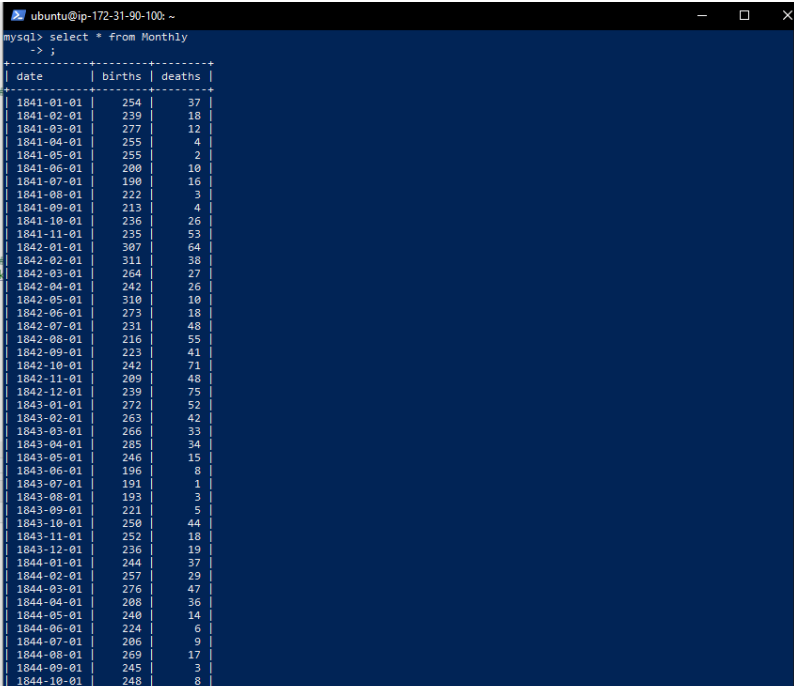
mysql> use mibd;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> show tables;
+-----+
Tables_in_mibd |
+-----+
Boxoffice       |
Monthly         |
Movies          |
Yearly          |
alcohol         |
bmw             |
north_american_cities |
+-----+
rows in set (0.01 sec)

mysql> select * from Yearly
-> ;
+-----+
year | births | deaths | clinic |
+-----+
1841 | 3036   | 237    | clinic 1 |
1842 | 3287   | 518    | clinic 1 |
1843 | 3060   | 274    | clinic 1 |
1844 | 3157   | 260    | clinic 1 |
1845 | 3492   | 241    | clinic 1 |
1846 | 4010   | 459    | clinic 1 |
1841 | 2442   | 86     | clinic 2 |
1842 | 2659   | 202    | clinic 2 |
1843 | 2739   | 164    | clinic 2 |
1844 | 2956   | 68     | clinic 2 |
1845 | 3241   | 66     | clinic 2 |
1846 | 3754   | 105    | clinic 2 |
+-----+
2 rows in set (0.00 sec)

mysql>
```

Ilustración 4: Bases de datos Yearly



```
mysql> select * from Monthly
> ;
```

date	births	deaths
1841-01-01	254	37
1841-02-01	239	18
1841-03-01	277	12
1841-04-01	255	4
1841-05-01	255	2
1841-06-01	200	10
1841-07-01	190	16
1841-08-01	222	3
1841-09-01	213	4
1841-10-01	236	26
1841-11-01	235	53
1842-01-01	307	64
1842-02-01	311	38
1842-03-01	264	27
1842-04-01	242	26
1842-05-01	310	10
1842-06-01	273	18
1842-07-01	231	48
1842-08-01	216	55
1842-09-01	223	41
1842-10-01	242	71
1842-11-01	209	48
1842-12-01	239	75
1843-01-01	272	52
1843-02-01	263	42
1843-03-01	266	33
1843-04-01	285	34
1843-05-01	246	15
1843-06-01	196	8
1843-07-01	191	1
1843-08-01	193	3
1843-09-01	221	5
1843-10-01	250	44
1843-11-01	252	18
1843-12-01	236	19
1844-01-01	244	37
1844-02-01	257	29
1844-03-01	276	47
1844-04-01	208	36
1844-05-01	240	14
1844-06-01	224	6
1844-07-01	206	9
1844-08-01	269	17
1844-09-01	245	3
1844-10-01	248	8

Ilustración 5: Bases de Datos Monthly

Utilizando la base datos Yearly bajo la utilizacion de las librias Tydeverse y ggplot2 se puede desarrollar la visualizacion de la evolucion de muertes por anioo en cada una de las dos clinicas 1 y 2

```
# Plot yearly proportion of deaths at the two clinics
# .... YOUR CODE FOR TASK 3 ....

ggplot(yearly,aes(x=year,y=proportion_deaths,color=clinic)) + geom_line()
```

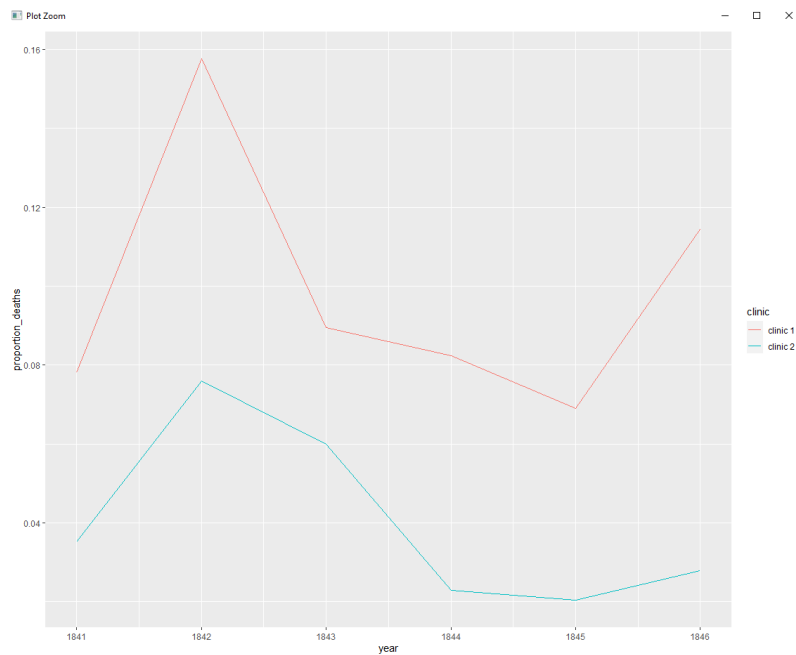



Ilustración 6 Muertes Neonatos por clinica

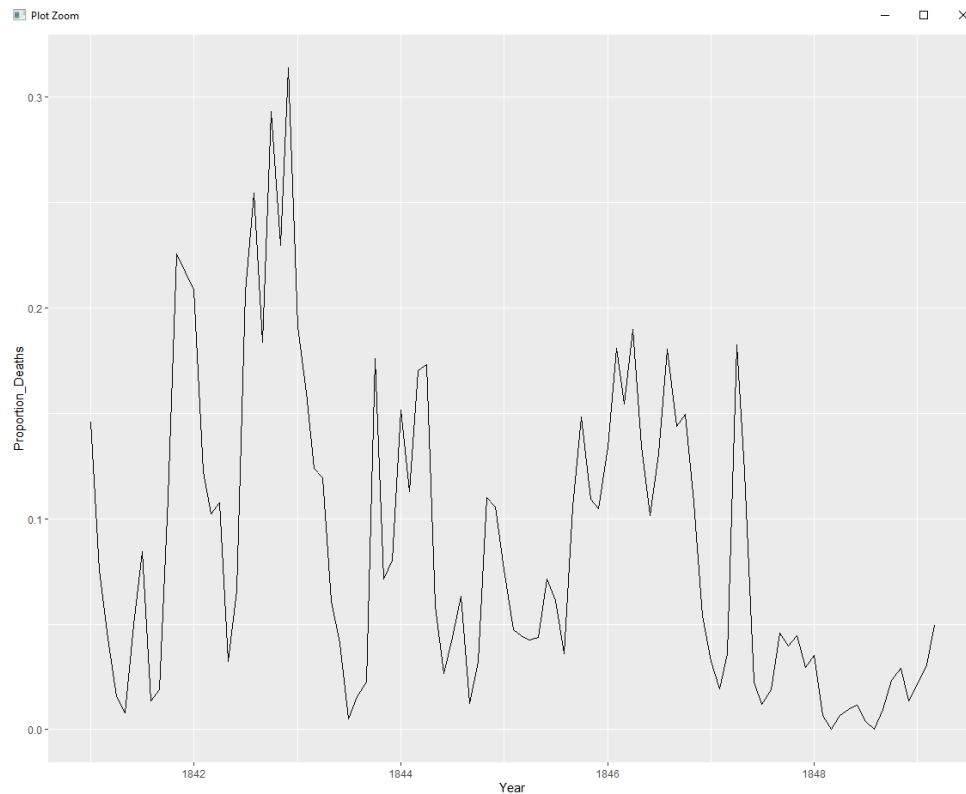
Luego utilizando la fuente de datos Monthly de MySQL EC2 podemos calcular la proporción de muertes por mes durante los años 1841 a 1846 utilizando el método mutate de tidyverse.

```
# A tibble: 98 x 4
  date       births deaths proportion_deaths
  <date>     <dbl>  <dbl>         <dbl>
1 1841-01-01    254     37         0.146
2 1841-02-01    239     18         0.0753
3 1841-03-01    277     12         0.0433
4 1841-04-01    255      4         0.0157
5 1841-05-01    255      2         0.00784
6 1841-06-01    200     10         0.05
7 1841-07-01    190     16         0.0842
8 1841-08-01    222      3         0.0135
9 1841-09-01    213      4         0.0188
10 1841-10-01    236     26         0.110
# ... with 88 more rows
```

```
monthly <- monthly %>% mutate(proportion_deaths = deaths/births)
print(monthly)

# Plot monthly proportion of deaths
# ... YOUR CODE FOR TASK 5 ...
ggplot(monthly, aes(x=date, y=proportion_deaths)) + geom_line() + labs(x="Year", y="Proportion_Deaths")
```

Por consiguiente se puede obtener la evolución de la proporción de muertes por mes desde 1841 a 1846



El Doctor Semmelweis hizo su recomendación y aseveración el **01-06-1847**, por consiguiente luego de implementar esta política del lavado de manos en las dos clínicas de estudio antes de atender a las parteras, se puede visualizar una importante reducción de la proporción de las muertes de neonatos.

```
# From this date handwashing was made mandatory
handwashing_start = as.Date('1847-06-01')

# Add a TRUE/FALSE column to monthly called handwashing_started
# ... YOUR CODE FOR TASK 6 ...
monthly <- monthly %>% mutate(handwashing_started = (date >= handwashing_start))

# Plot monthly proportion of deaths before and after handwashing
# ... YOUR CODE FOR TASK 6 ...
ggplot(monthly, aes(x=date, y = proportion_deaths, color=handwashing_started)) + geom_line() + labs(x = "Year", y = "Proportion of Deaths")

# calculate the mean proportion of deaths
```

```
# A tibble: 2 x 2
  handwashing_started mean_proportion_deaths
  <lgl>                <dbl>
1 FALSE                0.105
2 TRUE                 0.0211
```

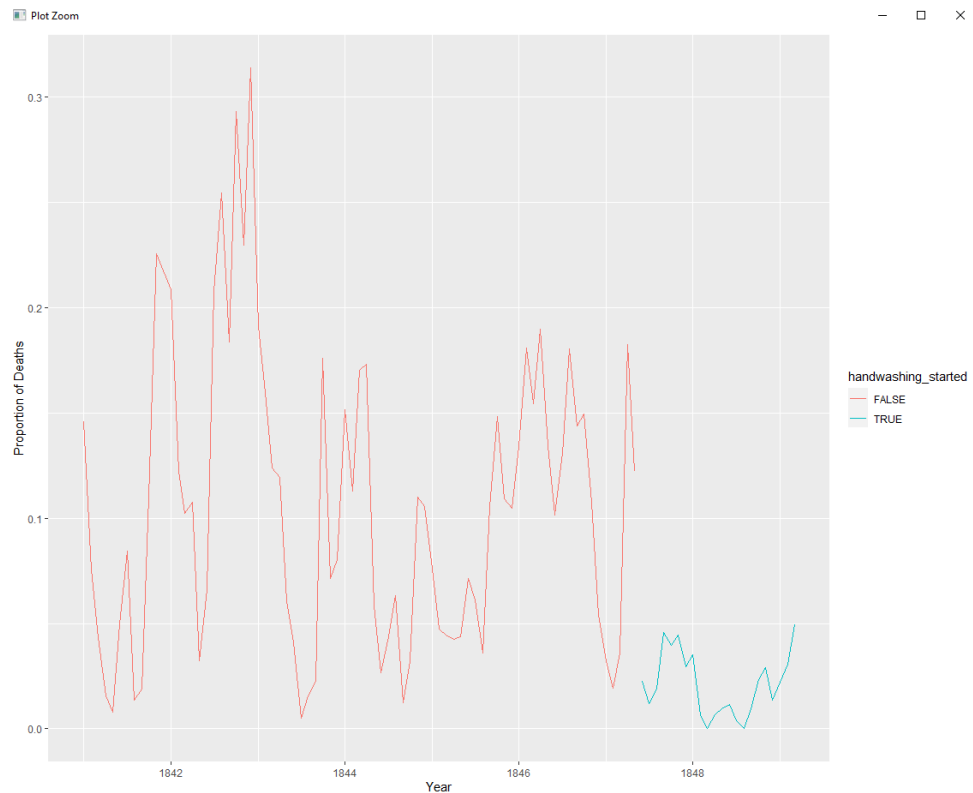


Ilustración 7: Evolución de muertes antes y después del lavado de manos

Por estas razones, podemos afirmar que las recomendaciones del Doctor permitió reducir la proporción de muertes en neonatos y dicha recomendación esta soportada por el análisis de los datos pues ser redujo la proporción de mortandad de 10.5% a 2.1%

```
2 TRUE 0.0211
> # Calculating a 95% Confidence interval using t.test
> test_result <- t.test( proportion_deaths ~ handwashing_started, data = monthly)
> test_result

welch Two Sample t-test

data: proportion_deaths by handwashing_started
t = 9.6101, df = 92.435, p-value = 1.445e-15
alternative hypothesis: true difference in means between group FALSE and group TRUE is not equal to 0
95 percent confidence interval:
 0.06660662 0.10130659
sample estimates:
mean in group FALSE mean in group TRUE
0.10504998 0.02109338

>
> # The data Semmelweis collected points to that:
> doctors_should_wash_their_hands <- TRUE
> doctors_should_wash_their_hands
[1] TRUE
> |
```

Segundo Caso: Análisis del consumo alcohol en Rusia

En el desarrollo de este caso de uso se utilizara la base de dato alcohol alojada en MySQL EC2 AWS instancia en conformidad a lo que se muestra en la siguiente ilustración. Asimismo, el dataset tiene 1615 rows y 7 columns

```

- attr(*, "problems")=<externalptr>
> summary(alcohol)
      year      region      wine      beer      vodka      champagne      brandy
Min.   :1998   Length:1615   Min.   : 0.100   Min.   : 0.40   Min.   : 0.05   Min.   :0.100   Min.   :0.000
1st Qu.:2002   Class  :character 1st Qu.: 3.575   1st Qu.: 32.40   1st Qu.: 8.30   1st Qu.:0.800   1st Qu.:0.200
Median :2007   Mode  :character   Median : 5.400   Median : 49.97   Median :11.50   Median :1.200   Median :0.400
Mean   :2007                                Mean   : 5.628   Mean   : 51.26   Mean   :11.82   Mean   :1.313   Mean   :0.527
3rd Qu.:2012                                3rd Qu.: 7.378   3rd Qu.: 67.40   3rd Qu.:15.00   3rd Qu.:1.665   3rd Qu.:0.700
Max.   :2016                                Max.   :18.100   Max.   :207.30   Max.   :40.60   Max.   :5.560   Max.   :2.300
NA's   :63                                NA's   :58      NA's   :61      NA's   :63      NA's   :66

> ndatos <- nrow(alcohol)
> ndatos
[1] 1615
>

```

year	region	wine	beer	vodka	champagne	brandy
2016	Moscow Oblast	8.7	99.1	9.6	2.2	1.3
2016	Murmansk Oblast	8.1	47.6	12.2	2.5	1.9
2016	Nenets Autonomous Okrug	11.7	59.6	11.5	2.6	1
2016	Nizhny Novgorod Oblast	6.3	56.6	6.5	1.1	0.7
2016	Novgorod Oblast	10.6	47.4	8.5	1.4	0.9
2016	Novosibirsk Oblast	6.7	46.6	4.9	1.3	0.6
2016	Omsk Oblast	5	53.2	3.5	1.1	0.5
2016	Orenburg Oblast	4.5	42.4	3.3	1	0.3
2016	Oryol Oblast	6.4	44.2	4.1	1.2	0.5
2016	Penza Oblast	5.9	51.1	4.4	0.6	0.3
2016	Perm Krai	5.5	51.5	7.1	0.9	0.7
2016	Primorsky Krai	6.8	72.4	8.6	2.2	0.8
2016	Pskov Oblast	10.4	36.5	6.4	1.5	0.8
2016	Altai Republic	4.5	42.2	6.6	1	0.3
2016	Komi Republic	9.6	63.4	12.5	1.6	1.2
2016	Tuva Republic	1.9	59.5	2.4	0.5	0.2
2016	Rostov Oblast	4.2	55.4	3.8	1.1	0.5
2016	Ryazan Oblast	6.3	41.1	4.9	1.2	0.6
2016	Samara Oblast	6.2	59.9	4.3	1.2	0.6
2016	Saint Petersburg	7.1	33.7	6.3	1.9	1.2
2016	Saratov Oblast	4.6	42.1	2.9	0.8	0.4
2016	Sakhalin Oblast	7.6	58.3	12.8	2.3	1.3
2016	Sverdlovsk Oblast	7.6	55.5	7.3	1.7	1
2016	Sevastopol	5.4	22.9	7	1.8	1.1
2016	Republic of North Ossetia-Alania	0.5	17.3	0.4	0.4	0.1
2016	Smolensk Oblast	7.9	48	6.8	1.6	0.8
2016	Stavropol Krai	3.8	57.4	2.3	1.1	0.5
2016	Tambov Oblast	4.8	58.2	3	0.8	0.4
2016	Republic of Tatarstan	4.4	59.1	9	0.6	0.5
2016	Tver Oblast	7.8	63.6	7.9	1.6	0.7
2016	Tomsk Oblast	5.1	43.4	4.5	1.4	0.6
2016	Tula Oblast	5.5	43.1	4.7	1	0.5
2016	Tyumen Oblast	5.1	73.4	8.4	1.9	1.1
2016	Udmurt Republic	5.5	58.1	8.6	0.9	0.6
2016	Ulyanovsk Oblast	6.6	46.9	4.6	1.1	0.5
2016	Khabarovsk Krai	7	73.7	11.2	2.6	0.9
2016	Republic of Khakassia	4.3	79.1	4.7	1.4	0.5
2016	Khanty-Mansi Autonomous Okrug - Yugra	4.3	67.5	9.2	2	1.3
2016	Chelyabinsk Oblast	5.3	42.6	5.1	1.2	0.6
2016	Chechen Republic	NULL	1.2	NULL	NULL	NULL
2016	Chuvash Republic	5	42.3	7.7	0.7	0.4
2016	Chukotka Autonomous Okrug	3.9	34	11.6	1.8	1.1
2016	Sakha (Yakutia) Republic	4.3	56.1	8.2	1.8	0.5
2016	Yamalo-Nenets Autonomous Okrug	4.5	75.8	8.2	1.7	1.3
2016	Yaroslavl Oblast	10.2	38	8.9	1.4	1

Ilustración 8 AWS Query a la DB Alcohol

Al explorar el conjunto de datos, se puede observar que existe datos perdidos sobre las ventas de los productos alcohólicos en ciertas regiones y años según la información resumen de la imagen adjunta

```

# =====
# DATOS PERDIDOS
# =====

# install.packages("VIM")
library(VIM)

# Mostrar cuales columnas tienen valores perdidos
cidx_perd <- which(colsums(is.na(alcohol))!=0)
cidx_perd

# Cantidad de valores perdidos en las columnas
nperdidos <- colsums(is.na(alcohol[,cidx_perd]))
nperdidos

# Porcentaje de valores perdidos en las columnas
pperdidos <- 100*nperdidos/ndatos
pperdidos

```

```

> # install.packages("VIM")
> library(VIM)
> # Mostrar cuales columnas tienen valores perdidos
> cidx_perd <- which(colSums(is.na(alcohol))!=0)
> cidx_perd
      wine      beer      vodka champagne      brandy
       3        4        5          6        7
>
> # Cantidad de valores perdidos en las columnas
> nperdidos <- colSums(is.na(alcohol[,cidx_perd]))
> nperdidos
      wine      beer      vodka champagne      brandy
       63       58       61        63       66
>
> # Porcentaje de valores perdidos en las columnas
> pperdidos <- 100*nperdidos/ndatos
> pperdidos
      wine      beer      vodka champagne      brandy
  3.900929  3.591331  3.777090  3.900929  4.086687
> |

```

Ilustración 9: Proporción de Datos Perdidos

Luego de explorar la data relacionada utilizando un esquema de visualización agregada se observa que la proporción de datos perdidos es menor igual a 0.04 en el peor de los casos por tanto se decide simplemente no considerar esos datos faltantes alineados a las buenas prácticas para su tratamiento vistos en el desarrollo del curso.

- 1 % de datos faltantes: trivial (el método de imputación no tiene mayor impacto)
- 1 a 10 % de datos faltantes: manejable (requiere un método “simple”)
- 10 a 20 % de datos faltantes: requiere métodos sofisticados (puede requerir método “propio”)
- Más del 20 % de datos faltantes: interpretación perjudicial (ya se perdió “demasiado”)

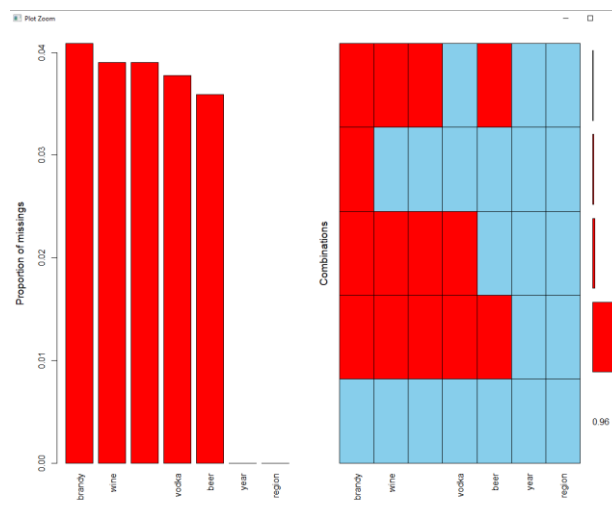


Ilustración 10: Proporción de Datos Perdidos

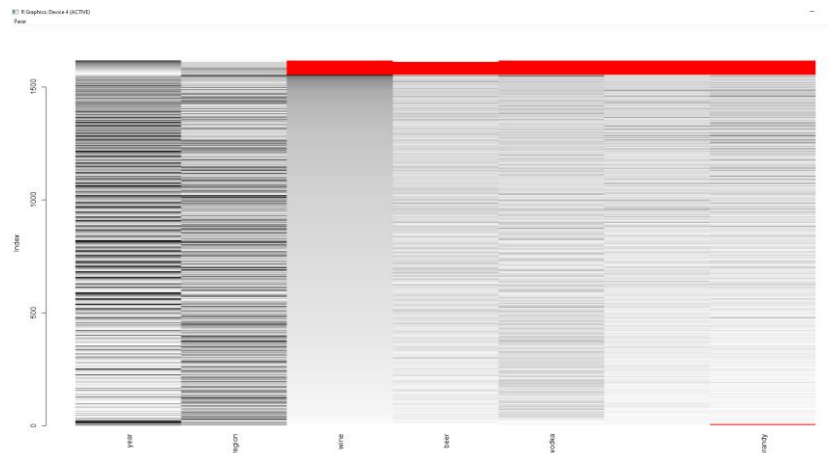


Ilustración 11: Patrones de Datos Perdidos

Por tanto se decide ejecutar el código para depurar la data eliminando los registros en Null o NA

```
# Boxplots paralelos ("parallel boxplots")
VIM::pbox(alcohol[3:7], pos=1) # pos=1 indica que se desea mostrar la variable 1

### Al tener un total de datos perdidos para cada producto menor al 4% la opción de eliminar los datos perdidos tiene
### conformidad con las buenas prácticas de imputación de datos |

alcohol <- na.omit(alcohol)
summary(alcohol)
```

```

> # Grafico de Matriz ("Matrix plot") ... En RStudio previamente usar: x11()
> matrixplot(alcohol)
> # Boxplots paralelos ("Parallel boxplots")
> VIM::pbox(alcohol[3:7], pos=1) # pos=1 indica que se desea mostrar la variable 1
> alcohol <- na.omit(alcohol)
> summary(alcohol)
```

	year	region	wine	beer	vodka	champagne	brandy
Min.	:1998	Length:1549	Min. : 0.100	Min. : 1.00	Min. : 0.40	Min. :0.100	Min. :0.000
1st Qu.:	:2002	Class :character	1st Qu.: 3.600	1st Qu.: 32.60	1st Qu.: 8.40	1st Qu.:0.800	1st Qu.:0.200
Median :	:2007	Mode :character	Median : 5.400	Median : 50.10	Median :11.50	Median :1.200	Median :0.400
Mean :	:2007		Mean : 5.639	Mean : 51.52	Mean :11.85	Mean :1.316	Mean :0.527
3rd Qu.:	:2012		3rd Qu.: 7.400	3rd Qu.: 67.50	3rd Qu.:15.00	3rd Qu.:1.680	3rd Qu.:0.700
Max. :	:2016		Max. :18.100	Max. :207.30	Max. :40.60	Max. :5.560	Max. :2.300

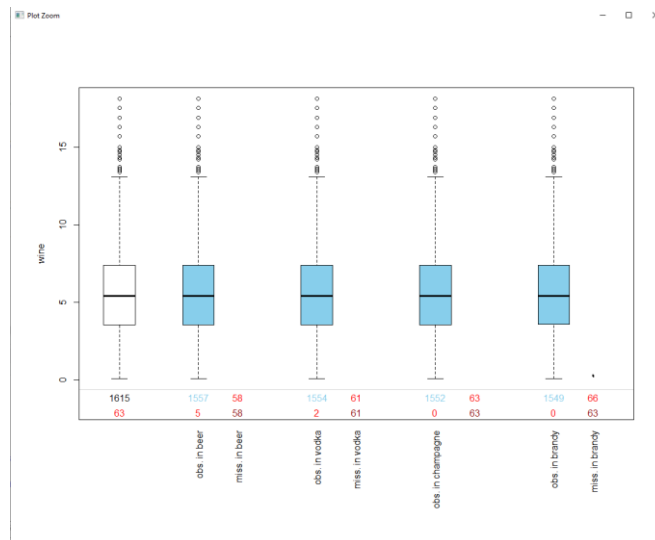


Ilustración 12: Datos Perdidos versus Observados

Al ser la ciudad de San Petersburg la ciudad patrón de ventas en donde se aplicó exitosamente la campaña de marketing, conviene visualizar cuales son las tendencias de ventas en esta región específica utilizando la librería ggpubr de Rstudio

```
##San Petersburg Alcohol Sales
ventasSaintPetersburg <- alcohol %>% filter(region == "Saint Petersburg") %>% group_by(year) %>% select(region,wine,beer,vodka,champagne,brandy)
ventasSaintPetersburg
View(ventasSaintPetersburg)
#Evolucion de Ventas Saint Petersburg
wine <- ggplot(ventasSaintPetersburg,aes(x=year,y= wine )) + geom_line()
beer <- ggplot(ventasSaintPetersburg,aes(x=year,y= beer )) + geom_line()
vodka <- ggplot(ventasSaintPetersburg,aes(x=year,y= vodka )) + geom_line()
champagne <- ggplot(ventasSaintPetersburg,aes(x=year,y= champagne )) + geom_line()
brandy <- ggplot(ventasSaintPetersburg,aes(x=year,y= brandy )) + geom_line()
# install.packages("ggpubr")
library(ggpubr)
final_plot <- annotate_figure(
  ggarrange(wine, beer, vodka, champagne,brandy, ncol=2, nrow=3),
  top = text_grob("Venta Saint Petersburg Products by Year", size = 20))
final_plot
```

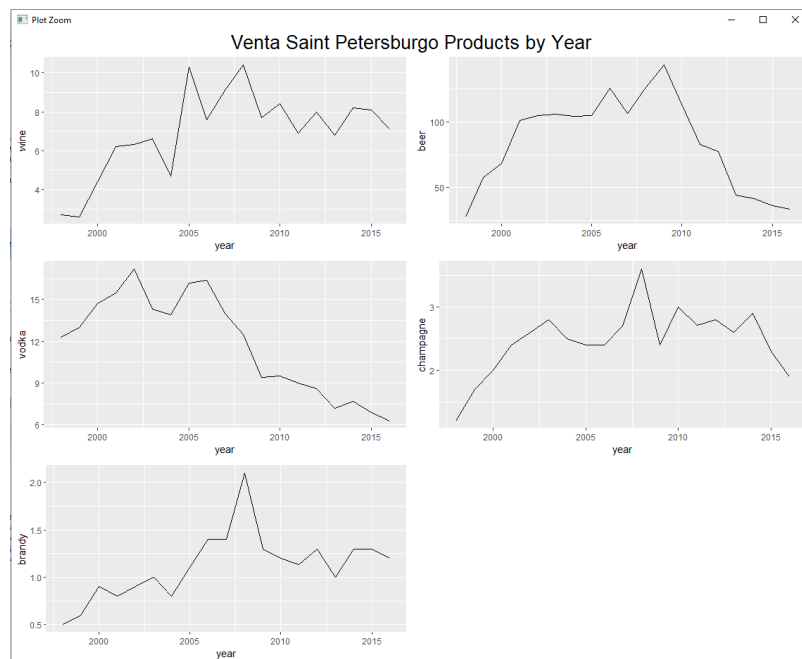


Ilustración 13: Saint Petersburg Produc Sales by Year Evolution

Luego, conviene evaluar la venta a lo largo de los años de cada producto: wine, beer, vodka, champagne, brandy identificando la tendencia anual y los TOP N regiones con mayor venta en cada línea de producto con el objetivo de identificar regiones top año en Rusia.

Se elaboró vistas interactivas en Shiny App para poder visualizar la evolución de las ventas por año

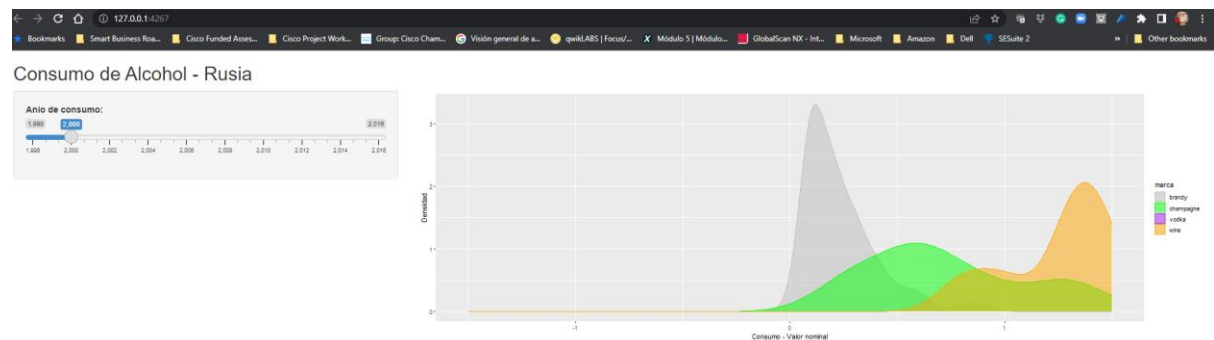


Ilustración 14: Shiny App View

```
##wine Sales Analysis
winesalesyearly <- alcohol %>% group_by(year) %>% select(year,region,wine) %>%top_n(1,wine)
winesalesyearly
topwinesalesyearly<- ggplot(winesalesyearly,aes(x=year,y= wine, color = winesalesyearly$region )) +labs(title = "Top Region Sales of wine Per year ") + geom_point()
topwinesalesyearly
df1 <- alcohol %>% group_by(year) %>% select(year,region,wine)
df1
df1yearly <- ggplot(df1,aes(x=year,y= wine )) + geom_col() + labs(title ="Top wine sales per Year")
df1yearly
```

Ilustración 15: Codigo Wine

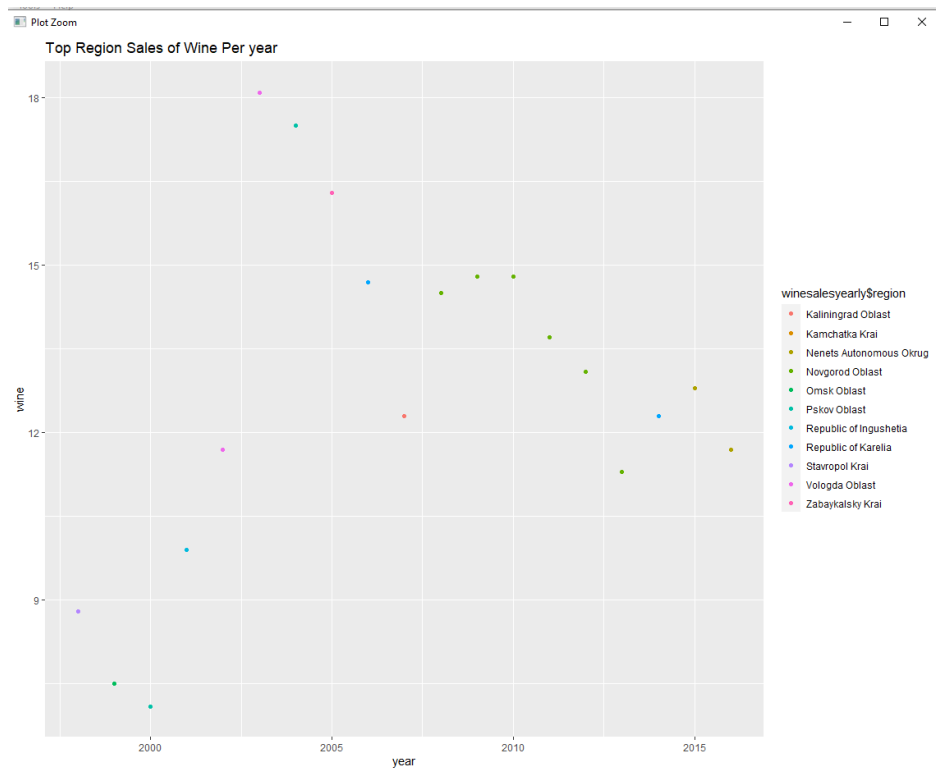


Ilustración 16: Top Wine Region Sales By Year

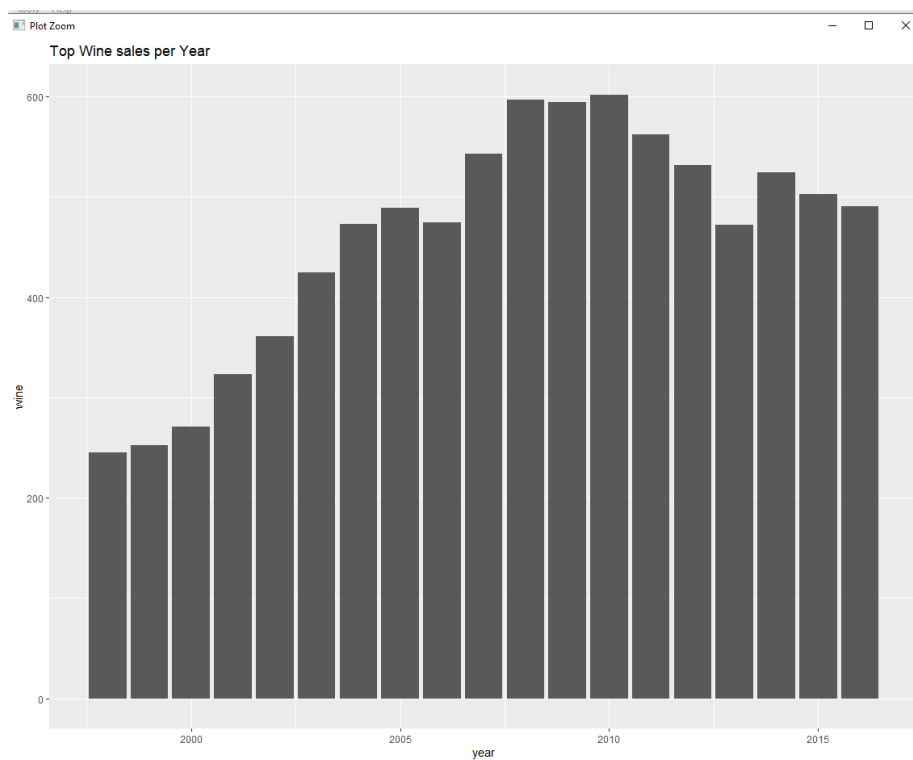


Ilustración 17: Wine Sales Evolution by Year

```
##beer sales Analysis
beersalesyearly <- alcohol %>% group_by(year) %>% select(year,region,beer) %>%top_n(1,beer)
beersalesyearly

topbeersalesyearly<- ggplot(beersalesyearly,aes(x=year,y= beer, color = beersalesyearly$region ))+labs(title = "Top Region Sales of beer Per year ") + geom_point()
topbeersalesyearly

df2 <- alcohol %>% group_by(year) %>% select(year,region,beer)
df2
df2yearly <- ggplot(df2,aes(x=year,y= beer )) + geom_col() + labs(title ="Top Beer sales per Year")
df2yearly
```

Ilustración 18: Codigo R Beer

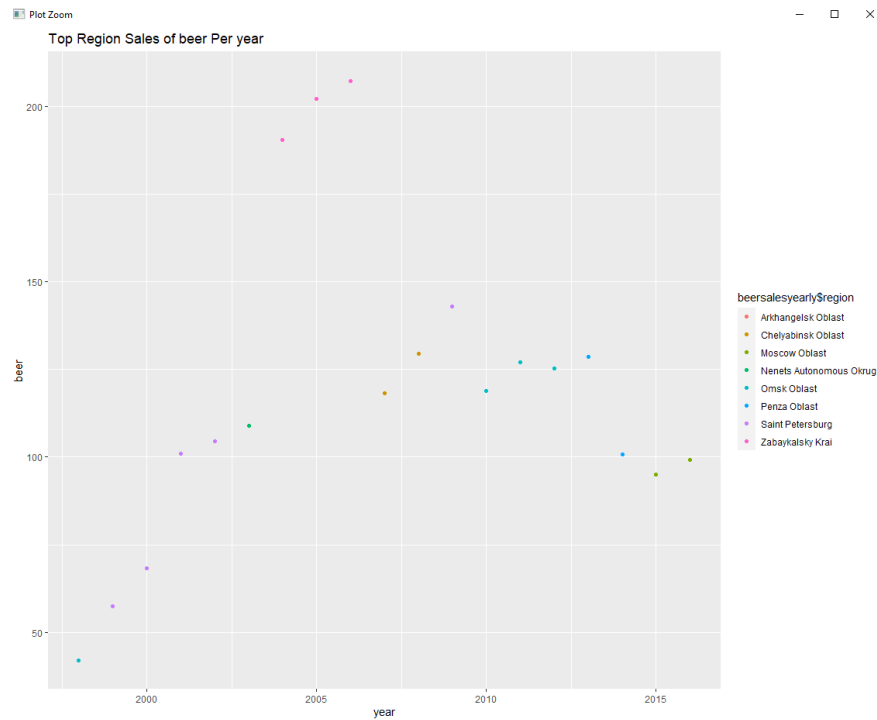
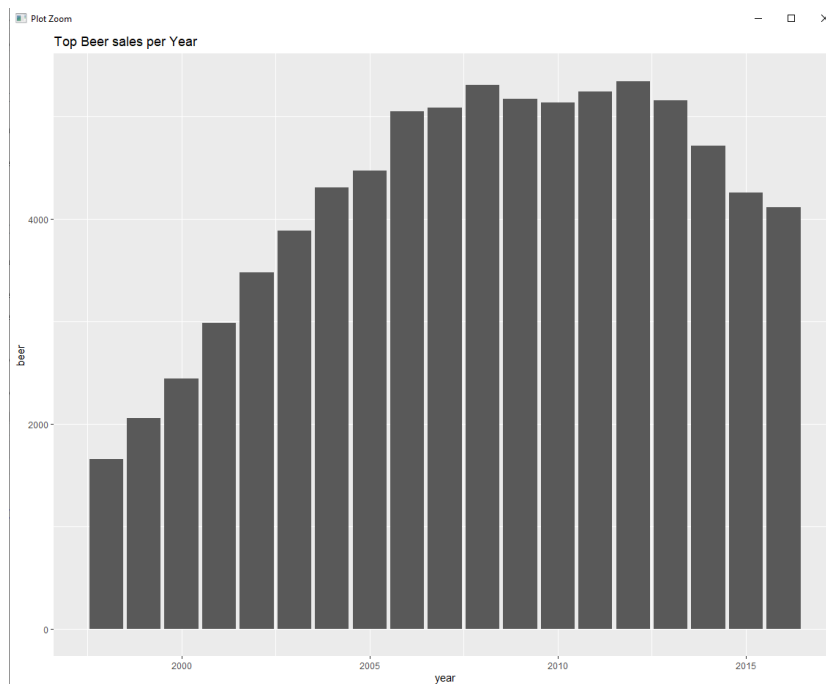


Ilustración 19: Top Beer Region by Year



```

#--yearly
##vodka Sales Analysis

vodka$yearly <- alcohol %>% group_by(year) %>% select(year,region,vodka) %>%top_n(1,vodka)
vodka$yearly

topvodka$yearly<- ggplot(vodka$yearly,aes(x=year,y= vodka, color = vodka$yearly$region )) +labs(title = "Top Region Sales of vodka Per year ") + geom_point()
topvodka$yearly

df3 <- alcohol %>% group_by(year) %>% select(year,region,vodka)
df3
df3yearly <- ggplot(df3,aes(x=year,y= vodka )) + geom_col() + labs(title ="Top vodka sales per Year")
df3yearly

##Chamoaone sales Analysis

```

Ilustración 20: Vodka Code

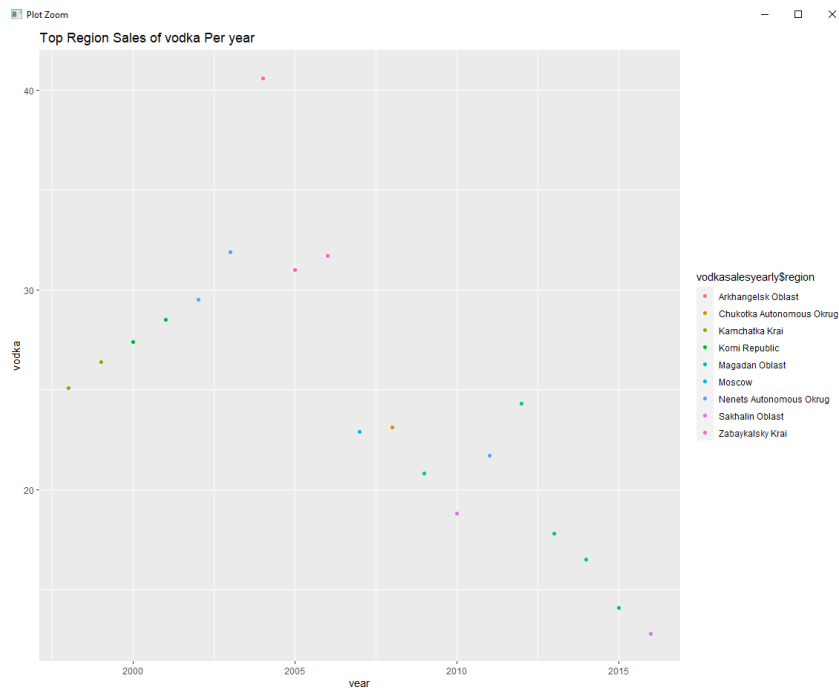


Ilustración 21: Vodka TOP Region Sales by Year

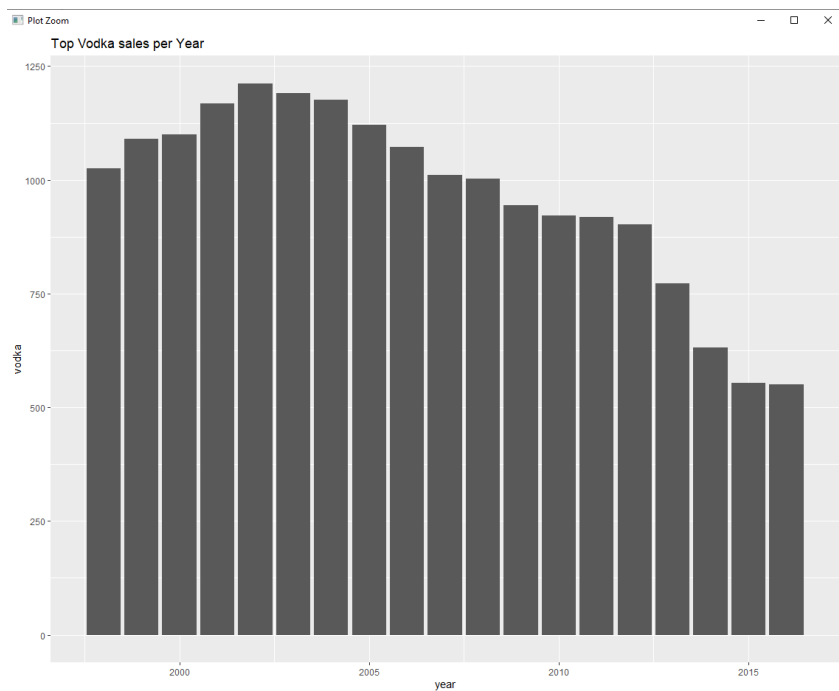


Ilustración 22: Vodka Evolucion Sales by Year

```
##Champagne sales Analysis
champagnesalesyearly <- alcohol %>% group_by(year) %>% select(year,region,champagne) %>%top_n(1,champagne)
champagnesalesyearly
topchampagnesalesyearly<- ggplot(champagnesalesyearly,aes(x=year,y= champagne, color = champagnesalesyearly$region )) +labs(title = "Top Region Sales of champagne per year ") + geom_point()
topchampagnesalesyearly
df4 <- alcohol %>% group_by(year) %>% select(year,region,champagne)
df4
df4yearly <- ggplot(df4,aes(x=year,y= champagne )) + geom_col() + labs(title = "Top Champagne Sales per Year")
df4yearly
```

Ilustración 23: Champagne RStudio Code

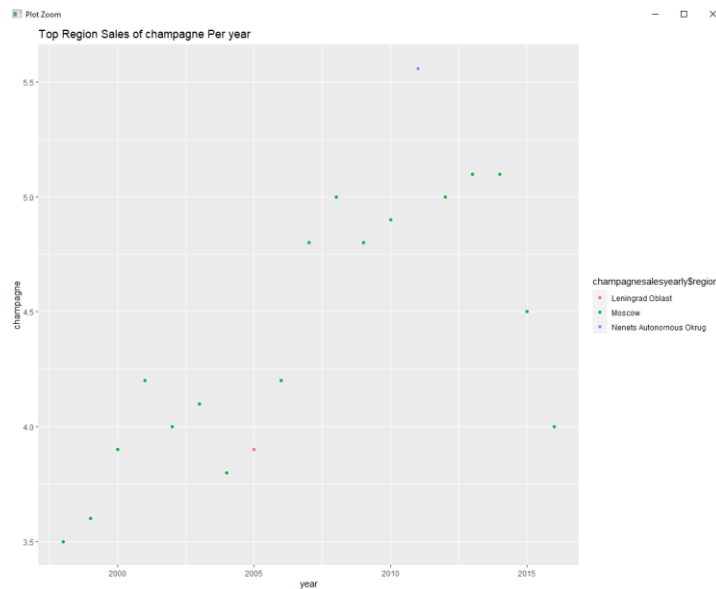


Ilustración 24: TOP Regional Champagne Sales by Year

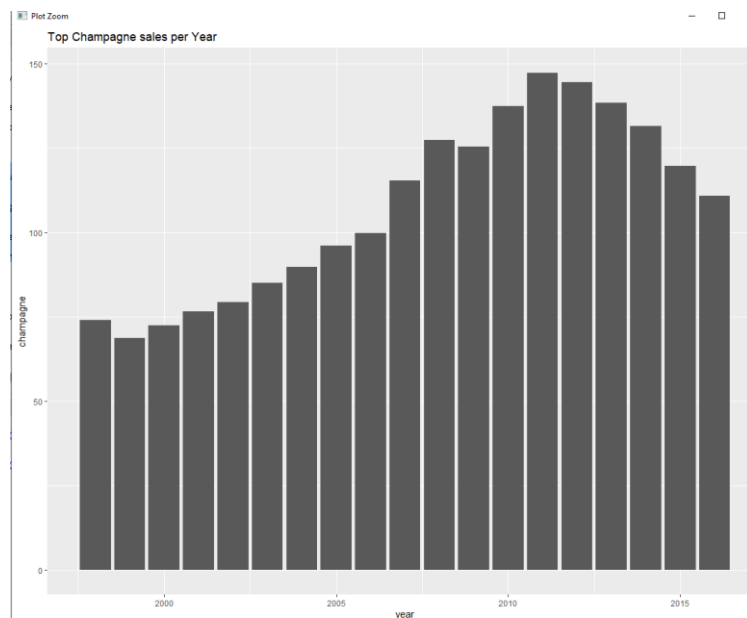


Ilustración 25: Champagne Sales Evolution by Year

```
##Brandy Sales Analysis

brandysalesyearly <- alcohol %>% group_by(year) %>% select(year,region,brandy) %>%top_n(1,brandy)
brandysalesyearly
topbrandysalesyearly<- ggplot(brandysalesyearly,aes(x=year,y= brandy, color = brandysalesyearly$region )) +labs(title = "Top Region Sales of Brandy per year ") + geom_point()
topbrandysalesyearly
df5 <- alcohol %>% group_by(year) %>% select(year,region,brandy)
df5
df5yearly <- ggplot(df5,aes(x=year,y= brandy )) + geom_col() + labs(title = "Top Brandy sales per Year")
df5yearly
```

Ilustración 26: Brandy Code View

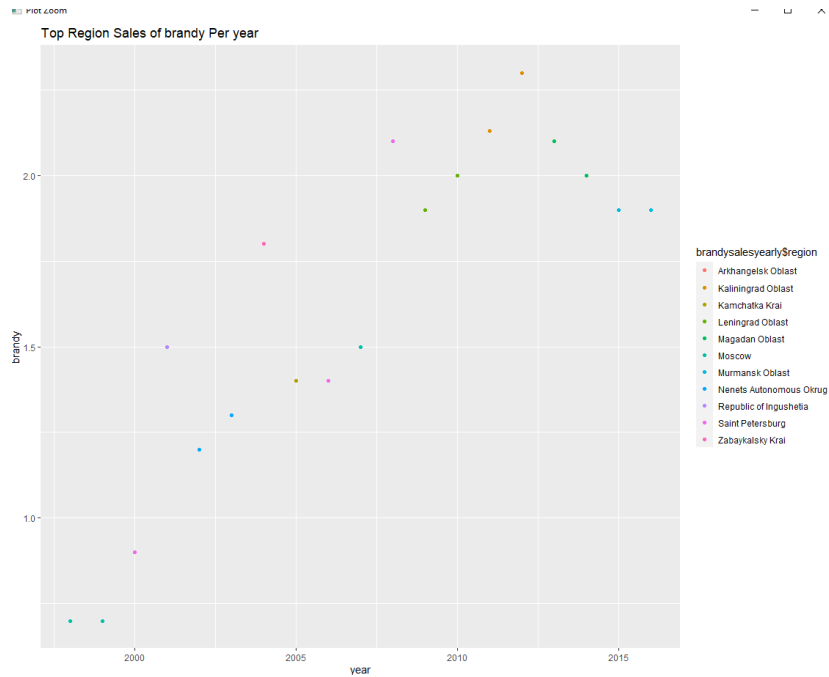


Ilustración 27: Top Brandy Region per Year

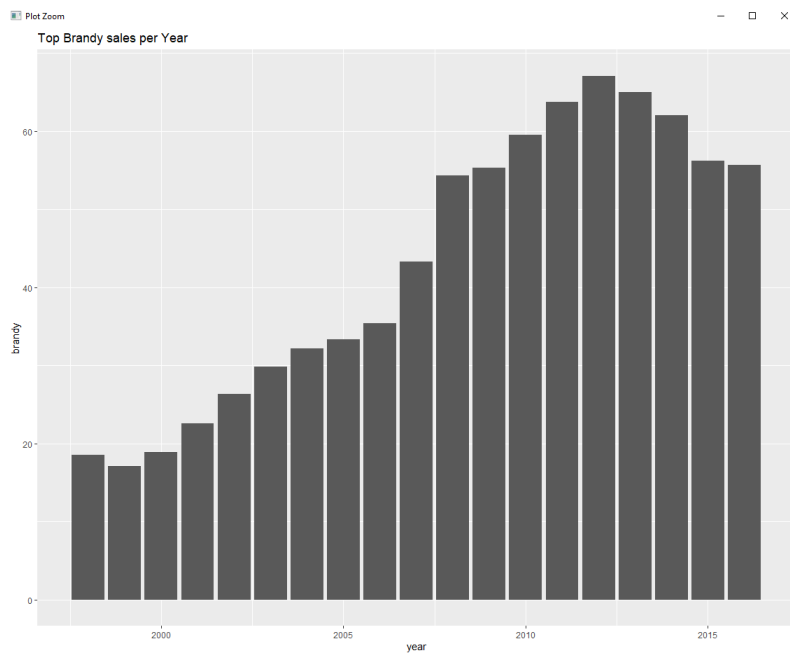


Ilustración 28: Brandy Sales Evolution by Year

A continuacion exploramos la vista boxplot y pairs entender la proporcion de datos outliers y evaluar las posibles correlaciones lado a lado de cada uno de los productos alcolicos. Observandose que el producto beer tiene mas outliers definidos . Por otro lado se observa cierta correlacion positiva entre las venta de Champagne y Brandy.

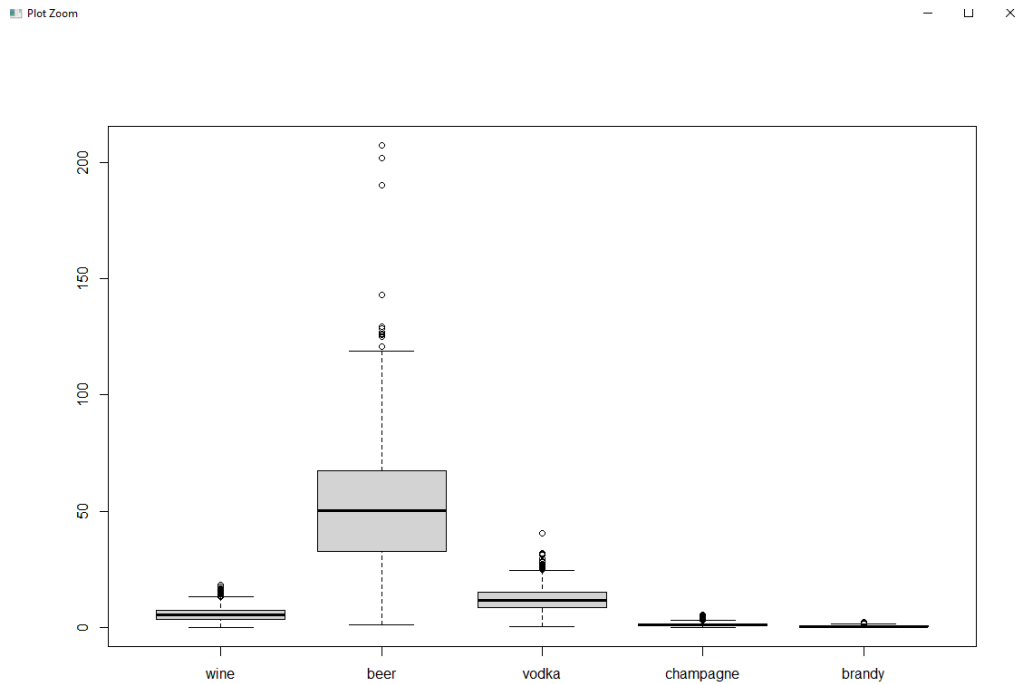


Ilustración 29: Visualizacion Boxplot y Ouliers data

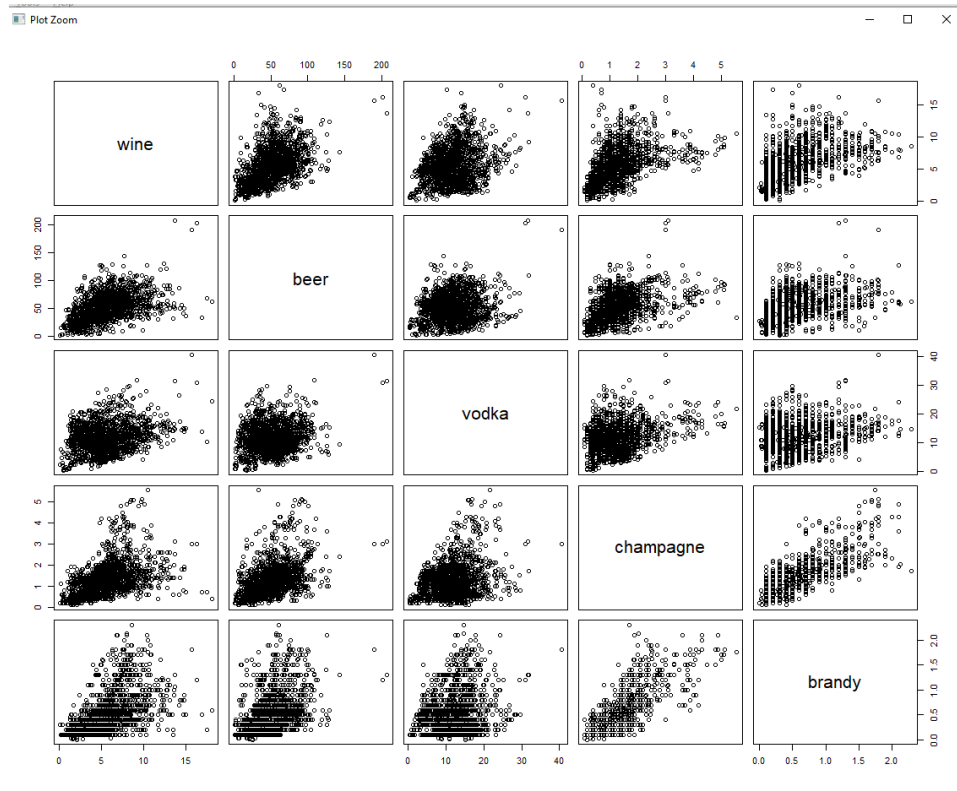


Ilustración 30: Grafico de Correlación de las Ventas de los Productos

Luego se evalúa el puntaje Z para los outliers además de evaluar los valores atípicos mediante los mecanismos QQ y Chi cuadrado, Mahalanobis donde se obtiene el $k=20.51$

```
# Outliers usando la Puntuación Z
#
is.outlier_z <- function(x, k=2) {
  return(abs(scale(x)) > k)      # scale: (x-media)/desv_est
}

# Boxplots para 5 producto (de manera independiente)
# Índices (T/F) de los wine atípicos
idx_outliers_z <- is.outlier_z(alcohol$wine, k=3)
which(idx_outliers_z)

# wine atípicos
alcohol$wine[idx_outliers_z]

# Registros asociados con los wine atípicos
alcohol[idx_outliers_z, ]
alcoholdata <- alcohol %>% select(wine,beer,vodka,champagne,brandy)
alcoholdata
```

```

# Distribución Chi-Cuadrado: Punto de Corte
p <- 1-0.001
dof = ncol(dfa)
k <- (qchisq(p, dof))
print("el valor de k es: ")
1] "el valor de k es: "
k
1] 20.51501
idx_outliers <- which(dm2 > k)
idx_outliers
[1] 200 261 282 364 411 419 446 449 501 505 515 523 541 587 596 604 668 690 771 852 996 1009
23] 1014 1077 1095 1098 1158 1173 1176 1239 1254 1257 1339
dfa[idx_outliers,] # Registros con valores atípicos
A tibble: 33 x 5
  wine beer vodka champagne brandy
<dbl> <dbl> <dbl> <dbl> <dbl>
1 6.3 61.8 18.8 3.9 0.6
2 9.9 9.5 4 1.3 1.5
3 5 78.9 19.7 4.2 0.7
4 4.8 85.7 20.8 4 0.9
5 9.2 109 31.9 1.7 1.3
6 18.1 61.7 24.6 0.4 0.6
7 6.2 89.1 23 4.1 0.9
8 9.2 109 31.9 1.7 1.3
9 14.2 85.1 26.7 0.6 0.7
0 15.7 190. 40.6 3 1.8
... with 23 more rows

```

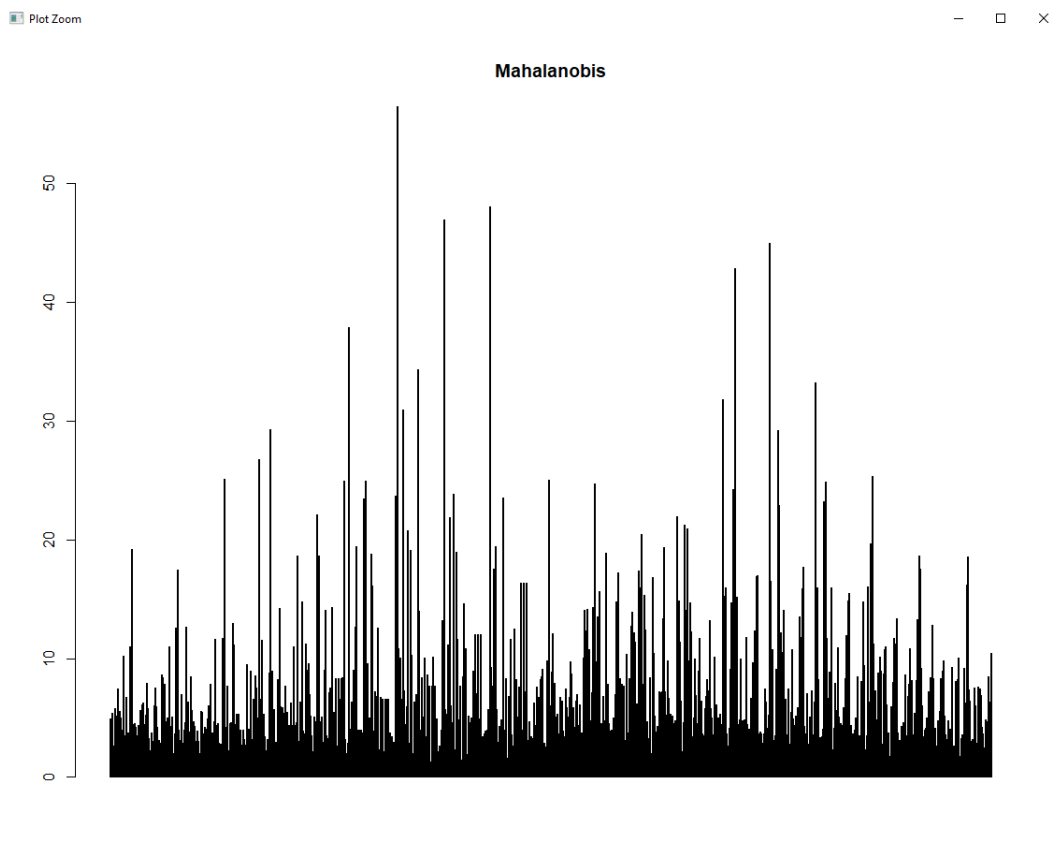


Ilustración 31: Outliers Data

Funcionalmente para visualizar la distribución de los valores atípicos se implementa la codificación para obtener la visualización de ojiva y Q-Q de χ^2 .

```

# Gráfico de Ojiva
plot(sort(dm2), ppoints(nrow(dfa)), xlab="DM al cuadrado ordenada",
     ylab="Probabilidad Acumulada")
abline(v = qchisq(p,dof), col = "red")

# QQ-plot:
x <- qchisq(ppoints(nrow(dfa)), dof)
y <- dm2
qqplot(x, y, main=expression("Q-Q plot para"~{chi^2}[nu==6]))
abline(0, 1, col="red")

```

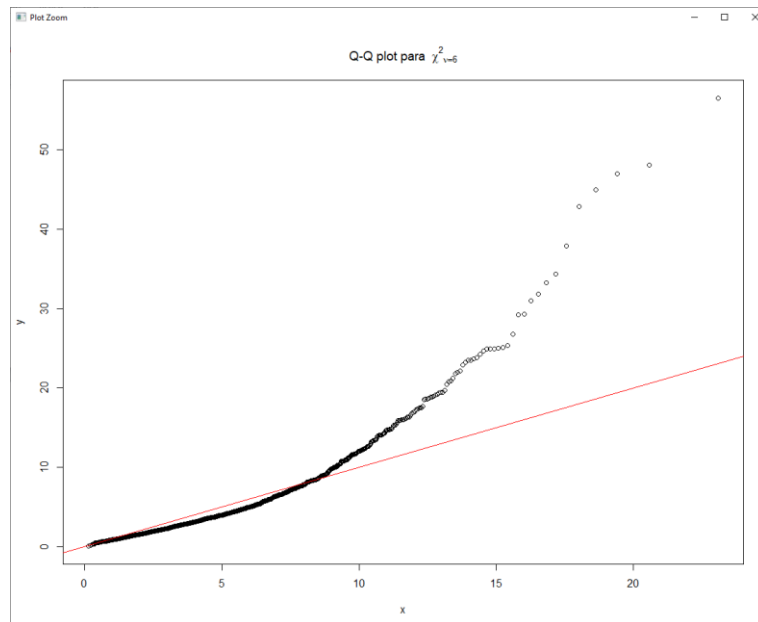



Ilustración 32: Grafico de Q-Q

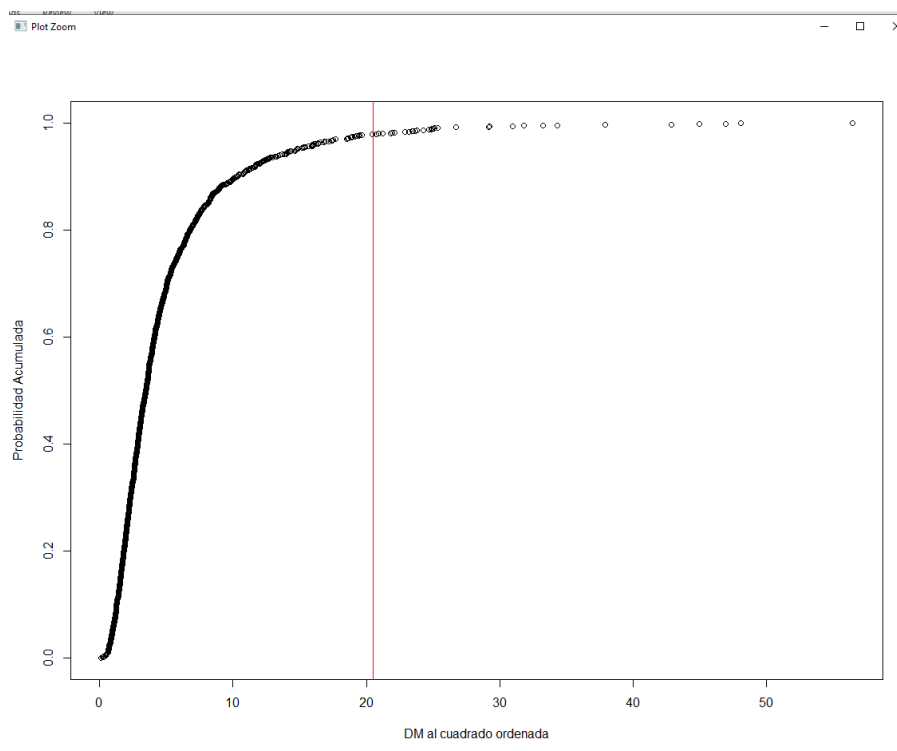


Ilustración 33: Grafico de Ojiva con Outiers

Finalmente, volvemos hacer las mismas visualizaciones descartando los valores atípicos identificados en el paso anterior y volvemos a graficar la Ojiva y Q-Q:

```

# Exclusión de valores atípicos (un valor atípico)
idx_excluido <- which.max(dm2)
dfa[idx_excluido, ]
dfa_clean <- dfa[-idx_excluido, ]

# Distancia de Mahalanobis
dm2 <- mahalanobis(dfa_clean, colMeans(dfa_clean), cov(dfa_clean))
plot(sort(dm2), ppoints(nrow(dfa_clean)), xlab="DM al cuadrado ordenada",
      ylab="Probabilidad Acumulada")
abline(v = qchisq(p,dof), col = "red")

idx_outliers <- which(dm2 > k)
idx_outliers

# QQ-plot dfa_clean:
x <- qchisq(ppoints(nrow(dfa_clean)), dof)
y <- dm2
qqplot(x, y, main=expression("Q-Q plot para"~{chi^2}[nu==6]))
abline(0, 1, col="red")

```

```

1 12.7 190. 40.0 5 1.0
> dfa_clean <- dfa[-idx_excluido, ]
>
> # Distancia de Mahalanobis
> dm2 <- mahalanobis(dfa_clean, colMeans(dfa_clean), cov(dfa_clean))
> plot(sort(dm2), ppoints(nrow(dfa_clean)), xlab="DM al cuadrado ordenada",
+      ylab="Probabilidad Acumulada")
> abline(v = qchisq(p,dof), col = "red")
>
> idx_outliers <- which(dm2 > k)
> idx_outliers
[1] 200 261 282 364 411 419 446 449 501 514 522 540 586 595 603 667 689 770 851 995 1008 1013
[23] 1076 1094 1097 1157 1172 1175 1238 1253 1256 1338
>
> # QQ-plot dfa_clean:
> x <- qchisq(ppoints(nrow(dfa_clean)), dof)
> y <- dm2
> qqplot(x, y, main=expression("Q-Q plot para"~{chi^2}[nu==6]))
> abline(0, 1, col="red")

```

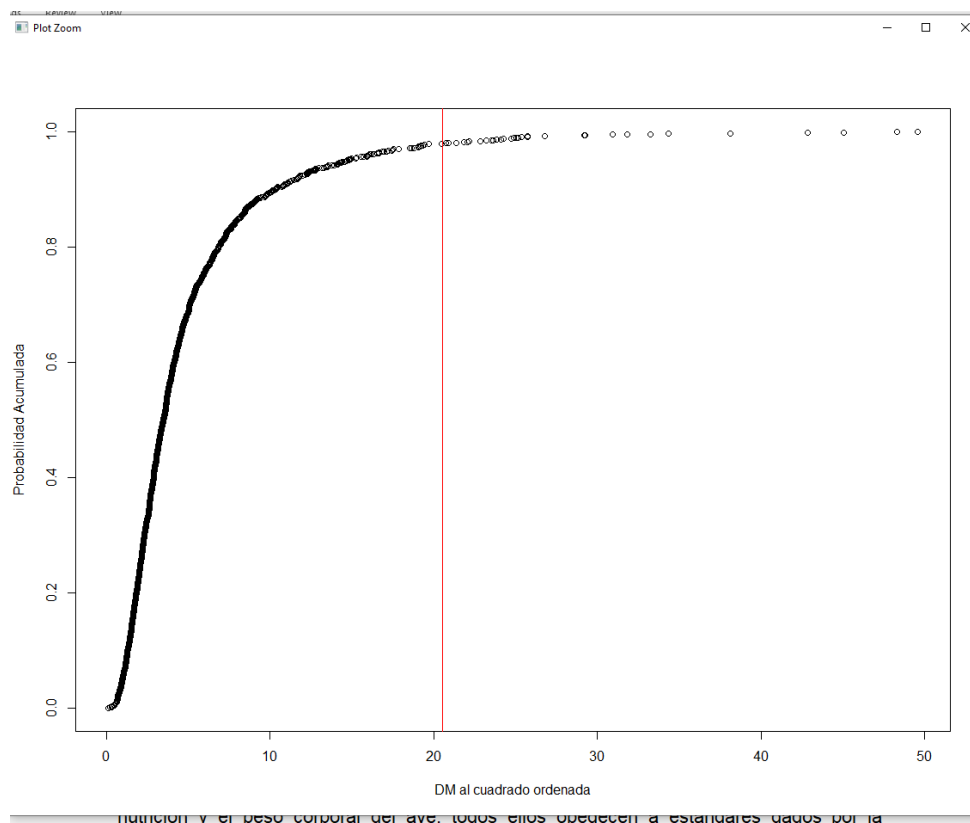


Ilustración 34: Grafico de Ojiva con limpieza de outliers

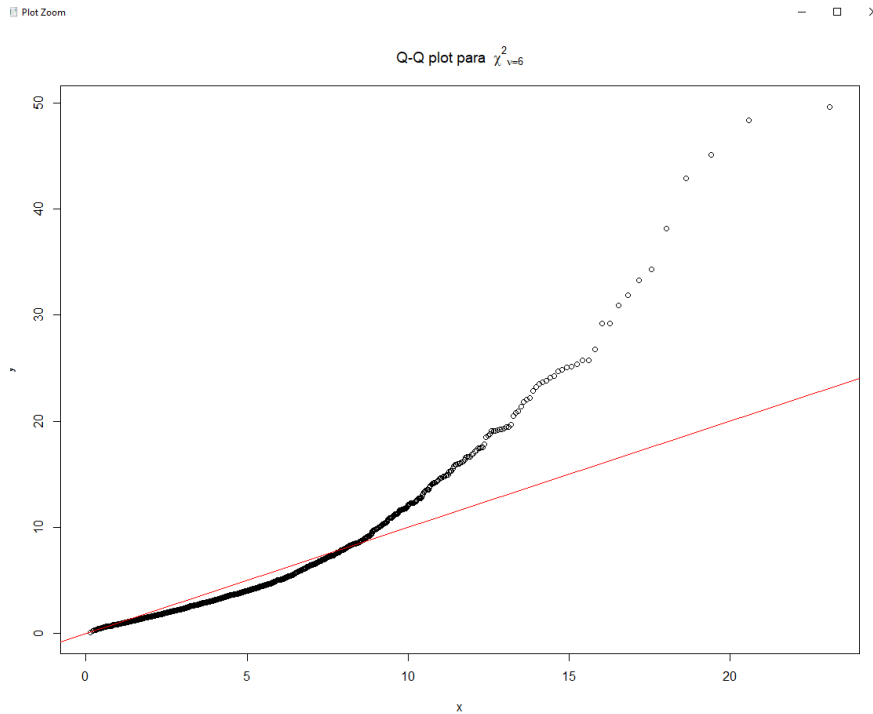


Ilustración 35: Grafico de Q-Q con limpieza de Outliers

Tercer Caso: Control de Pesos en la Crianza de Aves

El contexto de este caso está en la industria avícola de la empresa peruana Grupo Santa Elena S.A. que tiene como principal actividad la crianza de aves de corral, específicamente pollos engorde. El proceso de crianza de pollos busca lograr el máximo de kilogramos de carne de pollo y para ello debe cuidar sus parámetros productivos como la mortalidad, la nutrición y el peso corporal del ave; todos ellos obedecen a estándares dados por la genética del ave, sistema de crianza y sexo.

El control del peso por lo general se hace en cada semana, pero en la última etapa de la crianza se acorta a dos días. Esto significa que se registran los muestreos por cada galpón de crianza distinguiendo los machos de las hembras a los 7, 14, 21, 28, 35, 28, 40 y 42 días de edad de crianza.



Procedimiento del análisis exploratorio a los pesos de control de las aves

Paso 1: Se obtuvo los datos de una muestra de 1206 registros en formato CSV con la siguiente estructura:

ID_GALPON: Identificador del corral de crianza que es una división del galpón que agrupa aves del mismo sexo.

SEXO: Se identifica con "M" a los machos y con "H" a las hembras.

DIA07: Peso del en gramos a los 7 días de crianza.

DIA14: Peso del en gramos a los 14 días de crianza.

DIA21: Peso del en gramos a los 21 días de crianza.

DIA28: Peso del en gramos a los 28 días de crianza.

DIA35: Peso del en gramos a los 35 días de crianza.

DIA38: Peso del en gramos a los 38 días de crianza.

DIA40: Peso del en gramos a los 40 días de crianza.

DIA42: Peso del en gramos a los 42 días de crianza.

La siguiente es una muestra de los datos obtenidos:

ID_GALPON	SEXO	DIA07	DIA14	DIA21	DIA28	DIA35	DIA38	DIA40	DIA42
20201-2002-11-H	H	183	466	853			2220	2350	2480
20201-2002-11-M	M	178	478	960			2670	2790	2960
20201-2002-12-H	H	183	468	892			2270	2390	2460
20201-2002-12-M	M	183	468	892			2270	2390	2460
20201-2003-01-H	H	173	456	800	1300	1750	2000		2300
20201-2003-01-M	M	173	455	825	1400	2024		2100	
20201-2003-02-H	H	171	443	791	1280	1500			
20201-2003-02-M	M	171	443	791	1280	2068	2300		2640
20201-2003-03-H	H	172	453	808	1280	1771	2080	2150	2380

Paso 2: Mediante codificación Python se logró conectar a la base de datos alojado en el servidor de Atlas Mongo DB.

```
import pymongo
```

```
client = pymongo.MongoClient("mongodb://jgonzalez:rBR3AVVufD5bmQdg@cluster0-shard-00-00.4puwy.mongodb.net:27017,cluster0-shard-00-01.4puwy.mongodb.net:27017,cluster0-shard-00-02.4puwy.mongodb.net:27017/myFirstDatabase?ssl=true&replicaSet=atlas-2q64zi-shard-0&authSource=admin&retryWrites=true&w=majority")
```

```
db = client['BD_GRANJAS']
```

```
db.create_collection('pesos')
```

Paso 3: La carga de datos desde el archivo CSV a la base de datos de MongoDB se ejecutó comandos Python.

```
import csv
```

```
import pandas as pd
```

```
data = pd.read_csv('datos.csv', sep=";")
```

```
df = pd.DataFrame(data)
```

```
lista = []
```

```
row = {"_id": 1,
      'ID_GALPON': "",
      'SEXO': "",
      'DIA07': 0,
      'DIA14': 0,
      'DIA21': 0,
      'DIA28': 0,
      'DIA35': 0,
      'DIA38': 0,
      'DIA40': 0,
      'DIA42': 0}
```

```
n = len(df)
```

```
for i in range(n):
```

```
    row['_id'] = i
```

```

row['ID_GALPON'] = df.loc[i, "ID_GALPON"]
row['SEXO'] = df.loc[i, "SEXO"]
row['DIA07'] = df.loc[i, "DIA07"]
row['DIA14'] = df.loc[i, "DIA14"]
row['DIA21'] = df.loc[i, "DIA21"]
row['DIA28'] = df.loc[i, "DIA28"]
row['DIA35'] = df.loc[i, "DIA35"]
row['DIA38'] = df.loc[i, "DIA38"]
row['DIA40'] = df.loc[i, "DIA40"]
row['DIA42'] = df.loc[i, "DIA42"]
db.pesos.insert_one(row) #Insertando el documento

```

Paso 4: Conectividad y obtención de datos de Atlas MongoDB de la base de datos DB_GRANJA y la colección pesos

```

#Recuperando documentos de la colección
mycol = db["pesos"]
#Comprobando
for x in mycol.find().limit(5):
    print(x)

```

Paso 5: Lectura y recorrido de los datos obtenidos. El recojo de los datos como Collection se agregó a una lista para luego llevarlo a un objeto DataFrame Pandas

```

#Recuperando PESOS de las aves Macho
myquery = {"SEXO": "M"}
mydoc = mycol.find(myquery)
data_list = []
for x in mydoc:
    data_list.append(x)
df_machos = pd.DataFrame(data_list)

#Recuperando PESOS de las aves Hembra
myquery = {"SEXO": "H"}
mydoc = mycol.find(myquery)
data_list = []
for x in mydoc:
    data_list.append(x)
df_hembras = pd.DataFrame(data_list)

```

Paso 6: Explorando datos de los pesos mediante gráficos estadísticos. Se uso la librería matplotlib para plotear las gráficas.

Primero se recuperaron los pesos iniciales de 7 a 28 días.

```

import matplotlib.pyplot as plt
xaxes = ['DIA07', 'DIA14', 'DIA21', 'DIA28']
tleyen = ['upper right', 'upper right', 'upper left', 'upper right']

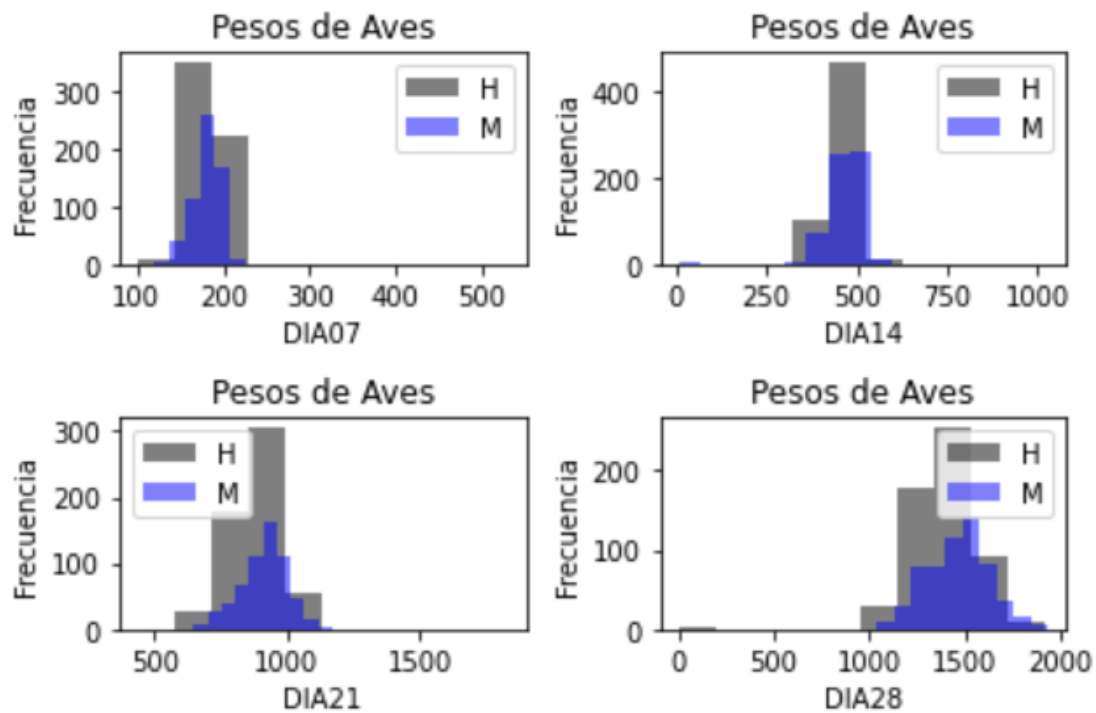
f,a = plt.subplots(2,2)
a = a.ravel()
for idx,ax in enumerate(a):
    ax.set_title('Pesos de Aves')
    ax.set_xlabel(xaxes[idx])
    ax.set_ylabel('Frecuencia')
    #Pesos de Hembras
    edad = xaxes[idx]
    df_gr_h = df_hembras[df_hembras[edad].notna()]
    ax.hist(df_gr_h[edad], alpha=0.5, color="black", label= "H")
    #Pesos de Machos
    edad = xaxes[idx]
    df_gr_m = df_machos[df_machos[edad].notna()]
    ax.hist(df_gr_m[edad], alpha=0.5, color="blue", label= "M")
    leg = ax.legend(loc=tleyen[idx], frameon=True)

plt.tight_layout()

```

Las gráficas de frecuencia se superponen las distribuciones de los pesos de hembras y machos. Se observa que los pesos de los machos tienen menor variabilidad frente a los pesos de las hembras. Las gráficas evidencian el peso según edad teniendo una media de 181 gramos a los siete días, luego sube a 460 gramos a los 14 días, sube a 881 gramos a los 21 días y llega a los 1394 gramos a los 28 días. A los 28 días ya se empiezan a evaluar a las aves para su saca o entrega como producto terminado como pollo en pie.

	_id	DIA07	DIA14	DIA21	DIA28
count	603.000000	586.000000	589.000000	566.000000	564.000000
mean	594.870647	181.279863	460.422750	881.199647	1394.641844
std	365.135787	21.711949	55.563559	98.470753	171.531052
min	0.000000	101.000000	20.000000	439.000000	1.000000
25%	253.000000	172.000000	435.000000	821.250000	1299.500000
50%	597.000000	183.000000	469.000000	888.000000	1400.000000
75%	922.000000	192.000000	490.000000	937.750000	1501.250000
max	1233.000000	530.000000	1030.000000	1830.000000	1920.000000



Por separado recuperamos los pesos a partir de los 35 días, es decir en edades disponibles para su comercialización. Se evidencia que la media a los 35 días es de 1990 gramos, pasa a 2210 a los 38 días, llega a los 2339 gramos a los 40 y cierra con 2441 gramos a los 42 días. Son pesos medios entre machos y hembras, lo que sirve para hacer cálculos de rendimiento del proceso por los kilogramos de carne de pollo como resultados.

	DIA35	DIA38	DIA40	DIA42
count	552.000000	457.000000	413.000000	369.000000
mean	1989.878623	2209.582057	2339.242131	2441.124661
std	227.005716	255.711328	340.366150	345.341467
min	1500.000000	1620.000000	1.000000	2.000000
25%	1829.750000	2020.000000	2153.000000	2275.000000
50%	1980.000000	2170.000000	2300.000000	2412.000000
75%	2125.250000	2360.000000	2547.000000	2603.000000
max	3260.000000	3270.000000	3284.000000	3367.000000


```

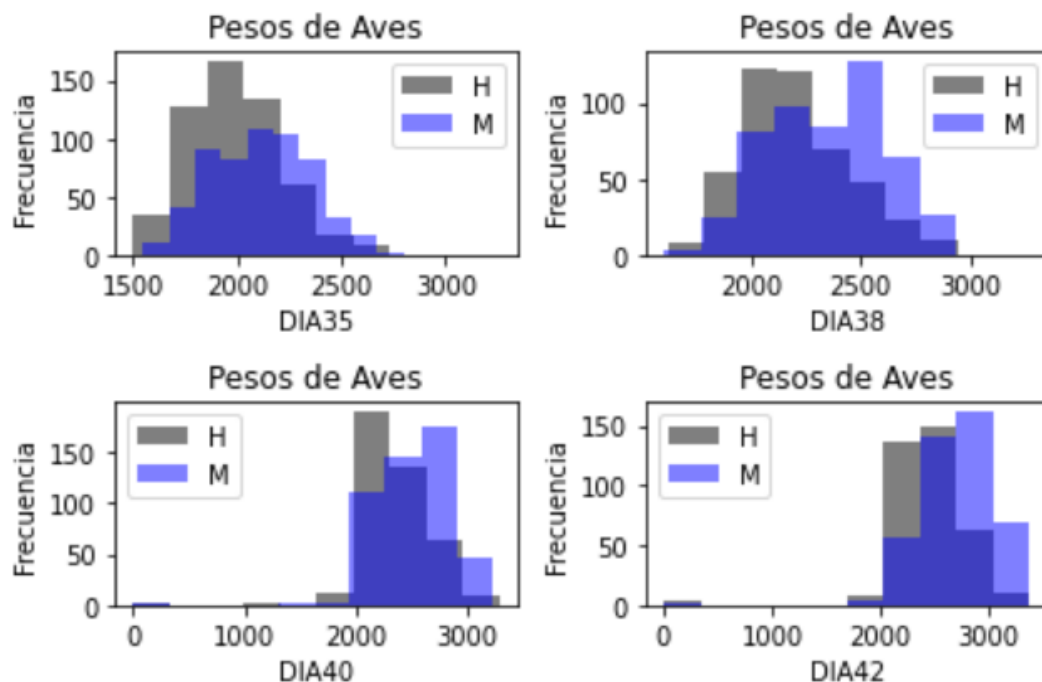
xaxes = ['DIA35', 'DIA38', 'DIA40', 'DIA42']
tleyen = ['upper right', 'upper right', 'upper left', 'upper left']

f, a = plt.subplots(2, 2)
a = a.ravel()
for idx, ax in enumerate(a):
    ax.set_title('Pesos de Aves')
    ax.set_xlabel(xaxes[idx])
    ax.set_ylabel('Frecuencia')
    #Pesos de Hembras
    edad = xaxes[idx]
    df_gr_h = df_hembras[df_hembras[edad].notna()]
    ax.hist(df_gr_h[edad], alpha=0.5, color="black", label= "H")
    #Pesos de Machos
    edad = xaxes[idx]
    df_gr_m = df_machos[df_machos[edad].notna()]
    ax.hist(df_gr_m[edad], alpha=0.5, color="blue", label= "M")
    leg = ax.legend(loc=tleyen[idx], frameon=True)

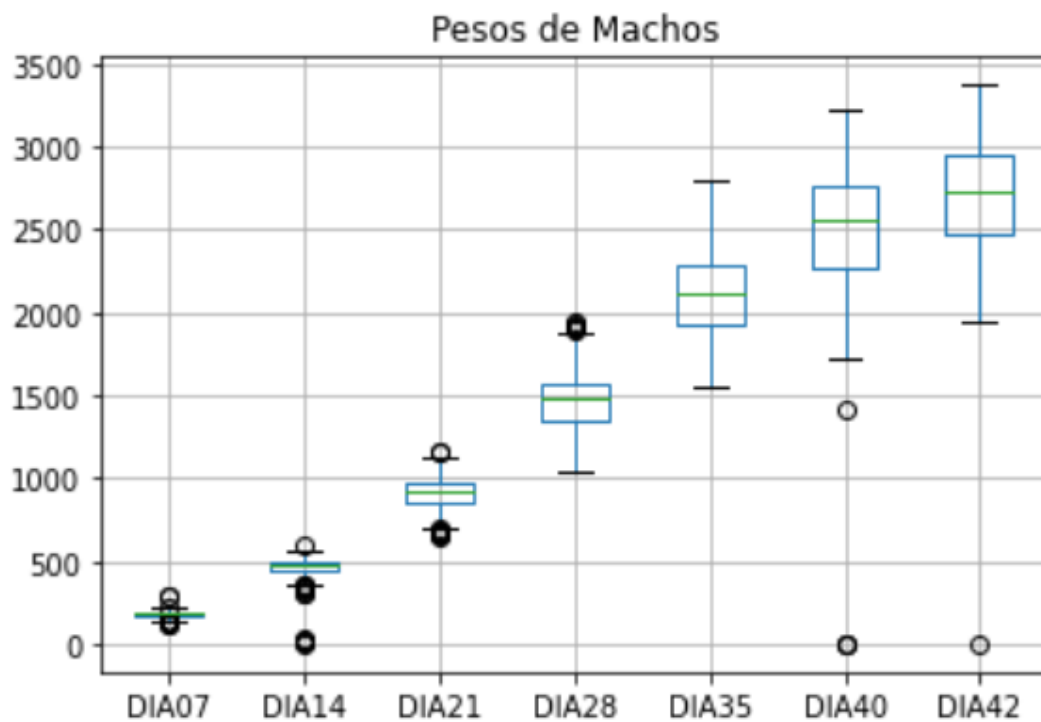
plt.tight_layout()

```

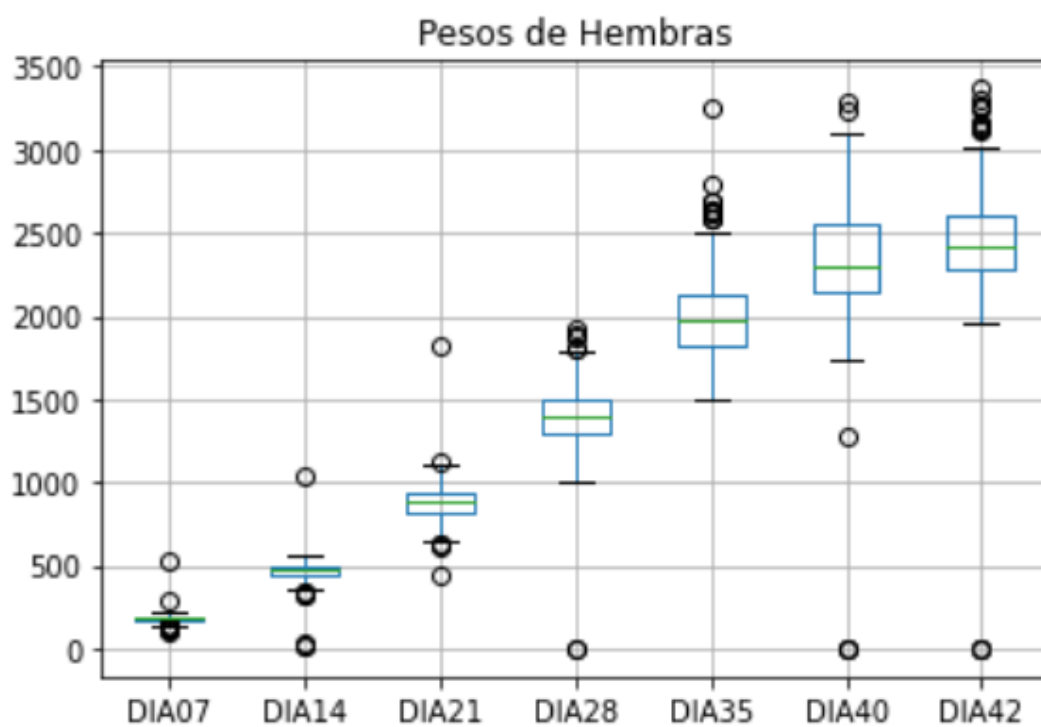
Se observa que los pesos de las aves macho terminan siendo mayores al de las hembras respetando el patrón estándar.



En el análisis de los “outliers” por sexo se observa en los pesos de las aves machos que en las primeras semanas hay mucho ruido y esto lo evidenciamos en la siguiente gráfica de cajas con la acentuación de las circunferencias a las 7 y 14 días.



En el caso de los pesos de las hembras el ruido de los “outliers” se da al inicio y al final de la crianza



Cuarto Caso Análisis de Outliers de precios en autos usados de la marca BMW

Para el cuarto caso, se analizará la data de venta de carros usados de la marca BMW del año 2020, aplicando las herramientas aprendidas en clase para determinar que atributos son considerados "outliers". La data contiene información de precio, año de fabricación, modelo, transmisión, millas, tipo de combustible, impuesto, millas por galón y tamaño de motor (en cilindros). Teniendo en cuenta esta información podríamos analizar si los outliers del precio de los autos usados, de la marca BMW, es determinado por el año de fabricación, tipo de transmisión, millas, millas por galón, impuesto y/o tamaño de motor o si existe algún otro tipo de factor que podría estar determinando estos precios.

Model – Modelos de la marca BMW

Year – Año de fabricación

Price – Precio de venta

Transmission – Tipo de transmisión del vehículo

Mileage – Millas recorridas

fuelType – Tipo de combustible

tax – Impuesto anual

mpg – Millas por galón

engineSize – Tamaño del motor en centímetros cúbicos

Metodología

Al igual que en los 2 primeros casos, se optó por utilizar el servicio MySQL instalado en una instancia de AWS EC2 con la finalidad de establecer una conexión al entorno de desarrollo y visualización de RStudio, habilitando así el acceso a todos los integrantes del grupo.

```
##### Conexion a MYSQL #####
db <- dbConnect(RMySQL::MySQL(),
               dbname = "mibd",
               host = "ec2-54-164-109-177.compute-1.amazonaws.com",
               user = "usuario",
               password = rstudioapi::askForPassword("Database password"),
               port = 3306)

data <- dbGetQuery(db, 'SELECT * FROM bmw')
data
```

Ejemplo del database bmw.csv

```
> data
```

	model	year	price	transmission	mileage	fuelType	tax	mpg	engineSize
1	5 Series	2014	11200	Automatic	67068	Diesel	125	57.6	2.0
2	6 Series	2018	27000	Automatic	14827	Petrol	145	42.8	2.0
3	5 Series	2016	16000	Automatic	62794	Diesel	160	51.4	3.0
4	1 Series	2017	12750	Automatic	26676	Diesel	145	72.4	1.5
5	7 Series	2014	14500	Automatic	39554	Diesel	160	50.4	3.0
6	5 Series	2016	14900	Automatic	35309	Diesel	125	60.1	2.0
7	5 Series	2017	16000	Automatic	38538	Diesel	125	60.1	2.0
8	2 Series	2018	16250	Manual	10401	Petrol	145	52.3	1.5
9	4 Series	2017	14250	Manual	42668	Diesel	30	62.8	2.0
10	5 Series	2016	14250	Automatic	36099	Diesel	20	68.9	2.0
11	x3	2017	15500	Manual	74907	Diesel	145	52.3	2.0
12	1 Series	2017	11800	Manual	29840	Diesel	20	68.9	2.0
13	x3	2016	15500	Automatic	77823	Diesel	125	54.3	2.0
14	2 Series	2015	10500	Manual	31469	Diesel	20	68.9	2.0
15	x3	2017	22000	Automatic	19057	Diesel	145	54.3	2.0
16	3 Series	2017	16500	Manual	16570	Diesel	125	58.9	2.0
17	3 Series	2017	14250	Automatic	55594	Other	135	148.7	2.0
18	3 Series	2017	16000	Automatic	45456	Diesel	30	64.2	2.0
19	1 Series	2017	15500	Automatic	22812	Diesel	20	68.9	1.5
20	4 Series	2014	14000	Automatic	47348	Diesel	125	60.1	2.0
21	1 Series	2015	9700	Automatic	75124	Diesel	20	70.6	2.0

En primer lugar, se verificó que la data no tuviera datos perdidos antes de hacer el análisis de outliers. En este caso, no se encontraron datos perdidos.

```
> #datos perdidos??
> sum(is.na(data))
[1] 0
```

Luego, se realizó el índice Z para detectar outliers de la columna precio.

```
#función de índice z para detectar outliers
is.outlier_z <- function(x, k=2) {
  return(abs(scale(x)) > k) # scale: (x-media)/desv_est
}
```

Ejemplo de los precios atípicos obtenidos al realizar el índice Z

```
> # Índices (T/F) de los precios atípicos
> idx_outliers_z <- is.outlier_z(data$price, k=3)
> which(idx_outliers_z)
[1] 190 242 361 363 375 420 557 599 637 720 722 735 769 810 876 910 914 929 1205 1225 1351
[22] 1355 1360 1630 1671 1672 1705 1744 1746 1772 1814 1849 1908 1959 1997 2018 2027 2066 2068 2074 2097 2109
[43] 2129 2459 2543 2544 2545 2567 2603 2610 2648 2725 2892 2910 2939 3046 3068 3101 3118 3127 3176 3220 3352
[64] 3410 3415 3462 3498 3512 3542 3545 3546 3639 3656 3682 3824 3911 3912 3918 3999 4008 4138 4150 4249 4281
[85] 4351 4385 4438 4466 4515 4527 4544 4572 4586 4612 4770 4776 4777 4798 4806 4834 4925 4962 5056 5141 5164
[106] 5174 5328 5363 5449 5459 5501 5535 5571 5588 5589 5599 5714 5727 5792 5878 5882 5899 5904 5914 5937 5950
[127] 6025 6057 6061 6069 6116 6117 6136 6139 6178 6180 6260 6271 6359 6361 6368 6375 6446 6447 6478 6513 6517
[148] 6591 6593 6607 6643 6661 6739 6747 6750 6790 6818 6860 6870 6882 6885 6907 6942 6959 6986 7011 7013 7024
[169] 7025 7045 7062 7063 7086 7107 7113 7139 7273 7298 7395 7751 7769 7803 7826 7868 7898 8487 8772 8823 9131
[190] 9144 9146 9793 9800 9829 9909 9984 9985 9994 10052 10056 10111

> # Precios atípicos
> data$price[idx_outliers_z]
[1] 61898 78000 72000 60990 67400 67000 63000 58898 68898 69898 84898 67898 58898 74990 67500 62898 64900 72990
[19] 66540 57870 64772 68930 63320 74226 57680 67120 76970 64690 68522 65352 88980 74480 61797 72430 71980 58991
[37] 67940 56980 56980 61380 69146 59990 62990 65050 77990 67990 77880 63990 78490 69990 57990 63980 89990
[55] 77880 59993 67990 74140 64980 59790 56980 58480 59990 56980 56990 56980 56990 58940 61895 61083 65000 123456
[73] 59995 74980 64400 64750 59000 66000 62980 61980 59980 64995 65000 57980 59000 64750 69940 56995 69995 64995
[91] 56995 59995 59995 60995 57950 68300 89990 65520 64103 58662 56990 59990 56990 63995 62995 74995 73950 99990
[109] 72000 58990 62995 58990 62991 69991 66991 61792 69995 62995 61854 81140 66750 57800 58000 73000 72500 59995
[127] 59950 66281 70000 64999 79991 69991 65000 71000 63999 64999 57000 59999 73990 59990 57990 61682 63806 78386
[145] 62995 70686 59995 73890 70995 67986 64652 58700 66290 68720 56995 63295 58994 69993 64995 72798 61995 67989
[163] 59989 60995 69995 59991 73990 56995 79566 69995 59973 59694 71990 56995 77995 61875 61485 59995 76990 65000
[181] 61550 62850 60000 68850 60000 59988 73900 58900 58988 74988 66988 64789 57000 72000 67950 65948 69948 69995
[199] 65000 57000 69880

> # Registros asociados con los precios atípicos
> data[idx_outliers_z, ]
  model year price transmission mileage fuelType tax mpg engineSize
190 8 Series 2019 61898 Semi-Auto 2650 Diesel 145 39.8 3.0
242 x7 2020 78000 Semi-Auto 5000 Diesel 150 31.4 3.0
361 m5 2019 72000 Semi-Auto 3545 Petrol 145 24.1 4.4
363 x5 2020 60990 Semi-Auto 3500 Diesel 145 33.6 3.0
375 x7 2019 67400 Semi-Auto 5600 Diesel 150 33.6 3.0
420 x7 2019 67000 Semi-Auto 4900 Diesel 145 33.6 3.0
557 8 Series 2020 63000 Semi-Auto 3000 Diesel 145 40.4 3.0
599 x5 2019 58898 Semi-Auto 100 Diesel 150 37.7 3.0
637 x7 2019 68898 Automatic 5203 Petrol 145 24.8 3.0
720 8 Series 2019 69898 Automatic 4075 Petrol 145 26.7 3.0
722 8 Series 2019 84898 Semi-Auto 3185 Petrol 145 24.4 4.4
735 8 Series 2019 67898 Semi-Auto 7903 Diesel 145 38.2 3.0
```

Asimismo, se usó la regla de Tukey para verificar con ambos métodos de valores atípicos univariados.

```
# Outliers usando la regla de Tukey
is.outlier <- function(x, k=1.5) {
  return(x < quantile(x,0.25)-k*IQR(x) | x > quantile(x,0.75)+k*IQR(x))
}
```

Ejemplo de los precios atípicos obtenidos al realizar la regla de Tukey

```
> # Indices (T/F) de precios atipicos
> idx_outliers <- is.outlier(data$price, k=3)
>
> # precios "outlier"
> data$price[idx_outliers]
[1] 78000 72000 67400 67000 68898 69898 84898 67898 74990 67500 72990 68930 74226 67120 76970 68522 88980 74480
[19] 72430 71980 67940 69146 77990 67990 77880 78490 69990 89990 77880 67990 74140 72990 123456 74980 69940 69995
[37] 69995 68300 89900 74995 73950 99950 72000 69991 66991 69995 81140 73000 72500 70000 79991 69991 71000 73990
[55] 78386 70686 73890 70995 67986 68720 69993 72798 67989 69995 73990 79566 69995 71990 77995 76990 68850 73900
[73] 74988 66988 72000 67950 69948 69995 69880
>
> # Registros asociados con los precios "outliers"
> data[idx_outliers,]
  model year  price transmission mileage fueltype tax  mpg engine size
242     X7 2020 78000     Semi-Auto   5000   Diesel 150  31.4      3.0
361     M5 2019 72000     Semi-Auto   3545   Petrol 145  24.1      4.4
375     X7 2019 67400     Semi-Auto   5600   Diesel 150  33.6      3.0
420     X7 2019 67000     Semi-Auto   4900   Diesel 145  33.6      3.0
637     X7 2019 68898     Automatic  5203   Petrol 145  24.8      3.0
720     8 Series 2019 69898     Automatic  4075   Petrol 145  26.7      3.0
722     8 Series 2019 84898     Semi-Auto   3185   Petrol 145  24.4      4.4
735     8 Series 2019 67898     Semi-Auto   7903   Diesel 145  38.2      3.0
810     X7 2019 74990     Semi-Auto   9200   Diesel 145  31.4      3.0
876     i8 2019 67500     Automatic  6000   Hybrid 140 141.2     1.5
929     8 Series 2019 72990     Semi-Auto   8200   Petrol 145  26.7      4.4
1355    M5 2019 68930     Automatic   123   Petrol 145  26.9      4.4
1630    i8 2019 74226     Automatic    10   Hybrid 135 141.2     1.5
1672    X7 2019 67120     Automatic    10   Diesel 145  33.6      3.0
1705    X7 2019 76970     Semi-Auto    10   Diesel 145  31.4      3.0
1746    X6 2019 68522     Semi-Auto 11330   Diesel 145  33.2      3.0
1814    8 Series 2019 88980     Semi-Auto    88   Petrol 145  24.4      4.4
1849    X7 2019 74480     Automatic  7278   Diesel 145  31.4      3.0
1959    X7 2019 72430     Semi-Auto  6391   Diesel 145  33.6      3.0
1997    X7 2020 71980     Semi-Auto  8695   Diesel 150  31.4      3.0
2027    i8 2019 67940     Automatic  1110   Hybrid 135 141.2     1.5
2097    X7 2019 69146     Automatic  2567   Petrol 145  24.8      3.0
2543    X7 2020 77990     Semi-Auto  5656   Diesel 150  31.4      3.0
2544    X6 2020 67990     Semi-Auto  5656   Diesel 150  33.2      3.0
2545    X7 2019 77880     Semi-Auto  6506   Diesel 150  31.4      3.0
2603    X7 2020 78490     Semi-Auto  4919   Diesel 145  31.4      3.0
2610    X7 2020 69990     Semi-Auto  5087   Diesel 145  33.6      3.0
2910    M4 2017 89990     Semi-Auto  1336   Petrol 145  33.2      3.0
2939    X7 2019 77880     Semi-Auto  6506   Diesel 145  31.4      3.0
3068    X7 2019 67990     Semi-Auto  6331   Petrol 145  24.8      3.0
3101    X7 2019 74140     Semi-Auto   3300   Diesel 145  33.6      3.0
```

Finalmente, se decidió por usar las librerías de ggplot2 y ggpubr para graficar boxplots de los datos outliers para cada caso. Donde se realizaron 6 graficas:

Modelo // Precio

Precio // Millas

Precio // Tax

Precio // Año

Precio // Millas por hora

Precio // Tamaño del motor

```

# Boxplots de precios // variables
bp1 <- data %>%
  mutate(outlier = ifelse(is.outlier(price), price, as.numeric(NA))) %>%
  ggplot(., aes(x = 1, y = price)) +
  geom_boxplot(fill="lightblue") +
  geom_text(aes(label = outlier), na.rm = TRUE, hjust = -0.3) +
  theme_bw()
bp1

bp2 <- data %>%
  mutate(outlier = ifelse(is.outlier(mileage), price, as.numeric(NA))) %>%
  ggplot(., aes(x = 1, y = mileage)) +
  geom_boxplot(fill="lightblue") +
  geom_text(aes(label = outlier), na.rm = TRUE, hjust = -0.3) +
  theme_bw()
bp2

bp3 <- data %>%
  mutate(outlier = ifelse(is.outlier(tax), price, as.numeric(NA))) %>%
  ggplot(., aes(x = 1, y = tax)) +
  geom_boxplot(fill="lightblue") +
  geom_text(aes(label = outlier), na.rm = TRUE, hjust = -0.3) +
  theme_bw()
bp3

bp4 <- data %>%
  mutate(outlier = ifelse(is.outlier(year), price, as.numeric(NA))) %>%
  ggplot(., aes(x = 1, y = year)) +
  geom_boxplot(fill="lightblue") +
  geom_text(aes(label = outlier), na.rm = TRUE, hjust = -0.3) +
  theme_bw()
bp4

bp5 <- data %>%
  mutate(outlier = ifelse(is.outlier(mpg), price, as.numeric(NA))) %>%
  ggplot(., aes(x = 1, y = mpg)) +
  geom_boxplot(fill="lightblue") +
  geom_text(aes(label = outlier), na.rm = TRUE, hjust = -0.3) +
  theme_bw()
bp5

bp6 <- data %>%
  mutate(outlier = ifelse(is.outlier(engineSize), price, as.numeric(NA))) %>%
  ggplot(., aes(x = 1, y = engineSize)) +
  geom_boxplot(fill="lightblue") +
  geom_text(aes(label = outlier), na.rm = TRUE, hjust = -0.3) +
  theme_bw()
bp6

# Generar una sola gráfica
final_plot <- annotate_figure(
  ggarrange(bp1, bp2, bp3, bp4, bp5, bp6, ncol=3, nrow=3),
  top = text_grob("Análisis Univariado de Valores Extremos", size = 15))
final_plot

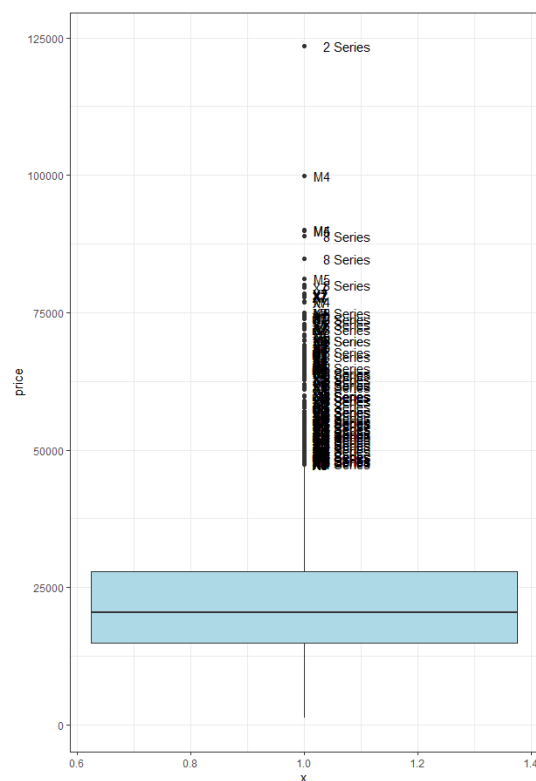
```

Resultados

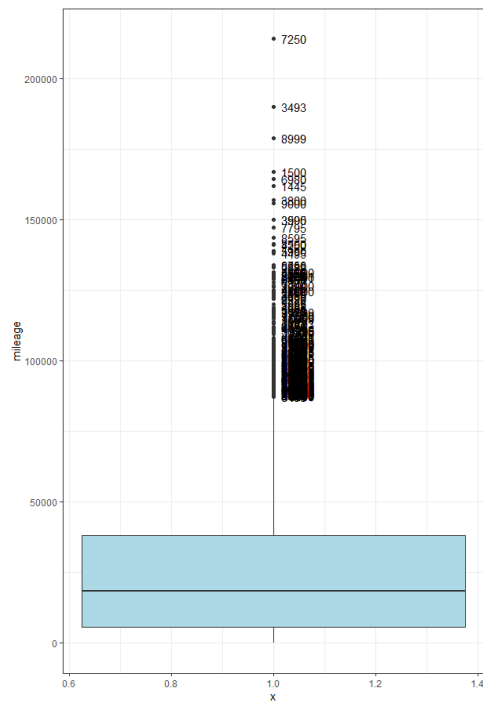
Cómo resultado se esperaba obtener una característica específica que determine el precio por encima del promedio en la venta de autos usados de la marca BMW. Y si bien se puede concluir que el precio del auto será mayor mientras más reciente sea el año de fabricación, menos millas tenga el vehículo y más grande sea el motor. También existen casos en donde aparenta haber otro atributo que no se encuentra en la base de datos que este influenciando el precio. Por otro lado, se puede pensar que hubo errores al ingresar la data, como por ejemplo con el auto que fue vendido por el precio más alto:

	model	year	price	transmission	mileage	fuelType	tax	mpg	engineSize
3639	2 Series	2015	123456	Semi-Auto	33419	Diesel	20	68.9	2.0

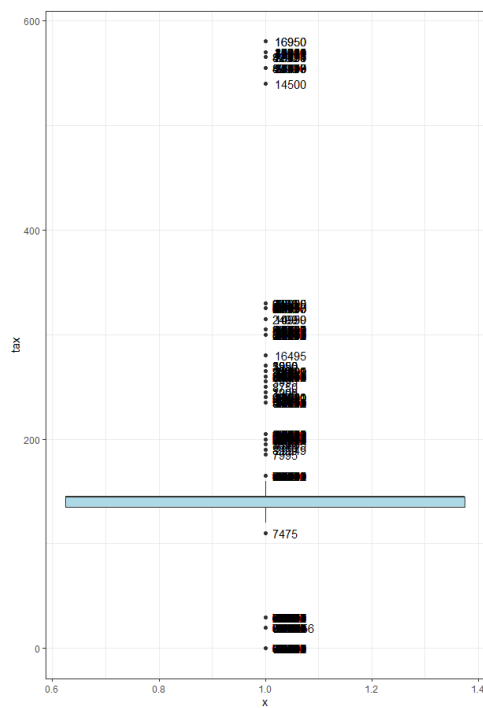
Precio = \$123456



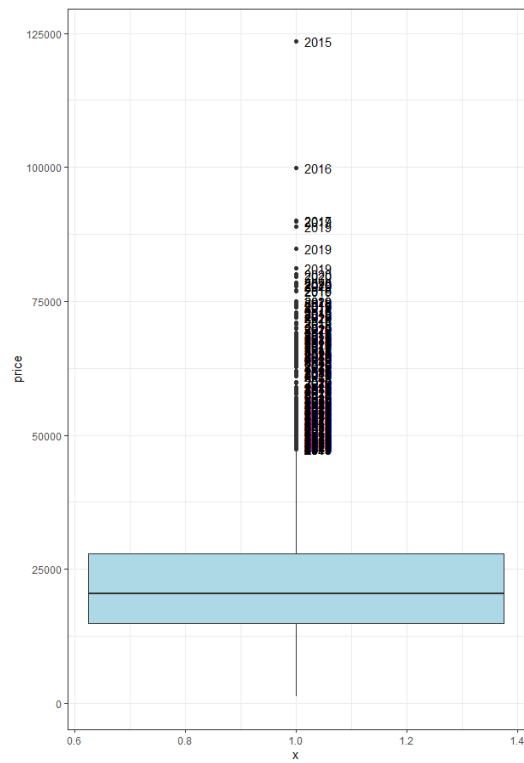
Boxplot 1 precios atípicos sobre modelo de BMW



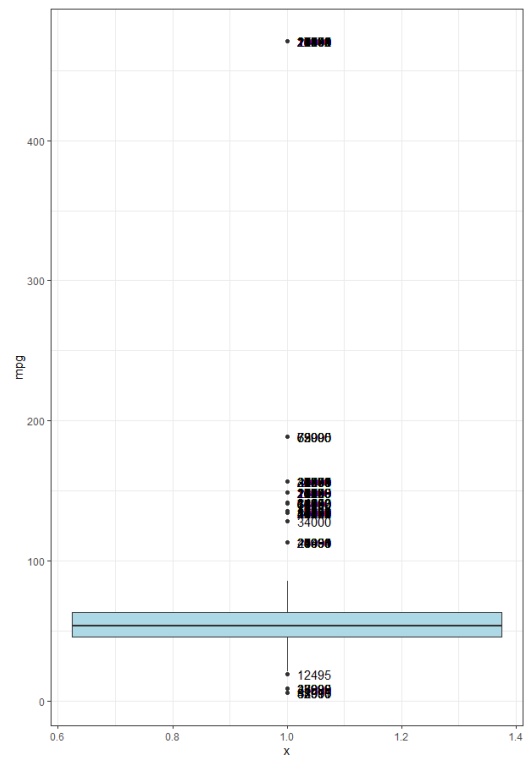
Boxplot 2 precios atípicos sobre millas acumuladas del vehículo



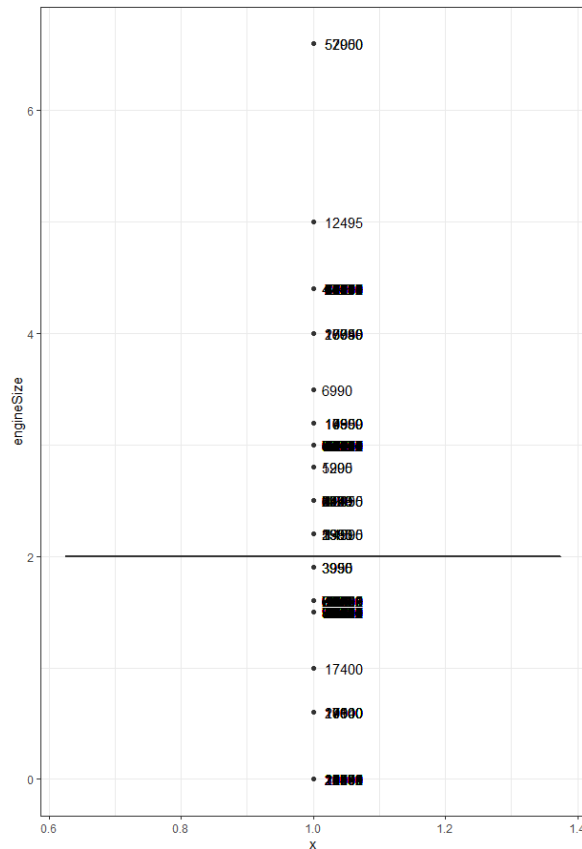
Boxplot 3 precios atípicos sobre impuesto anual



Boxplot 4 precios atípicos sobre año de fabricación



Boxplot 5 precios atípicos sobre millas por galón



Boxplot 6 precios atípicos sobre tamaño de motor

Conclusiones y Recomendaciones

Primer caso: Se replica la resolución de la hipótesis planteada por el dr.Semmelweis que sostuvo que el lavado de manos previa antes de una nueva atención de parteras reducía la mortandad de los niños en el SXIX, pues ser redujo la proporción de mortandad de 10.5% a 2.1%.

Segundo Caso: Se utilizan diversos esquemas de visualización interactiva, gestión de datos perdidos, análisis outsiders, para ofrecer al área de marketing las top 10 regiones idóneas para replicar las estrategias de mercadeo que fueron exitosas en Saint Petersburg.

Tercer Caso: Se realiza un análisis exploratorio de los pesos de las aves vivas aplicando graficas de distribución, bloxplot, etc. para detectar los outliers.

Cuarto Caso: En síntesis, se puede confirmar la hipótesis de que los atributos de año, millas y tamaño de motor son los que más interfieren en un precio por encima del promedio de los vehículos. Igualmente, se recomienda hacer otro tipo de análisis, cómo por ejemplo clustering, para hallar una predicción de precio más exacta con la data. Otra observación o recomendación que se encontró, es que dentro del atributo de “fuelType” se encontraron 36

vehículos con la variable “Other”, esto bien podría ser datos perdidos u otras fuentes de combustible, como por ejemplo gnv o glp, pero que no están explicitadas en la base de datos.

En relación a la tecnología se creyó convenientes utilizar los servicios de bigdata basados en nube como Atlas Mondo DB y MySQL/EC2 on AWS ya que ofrecen agilidad, rápida escalabilidad y flexibilidad para implementar nuevos casos de uso si la necesidad de involucrarse en proyectos complejos de implementación de centro de datos onpremises

La experiencia y usabilidad de Atlas Mondo DB y el entorno Google Collab Python ha sido satisfactoria a pesar de utilizar el free tier. Al mismo tiempo los servicios MySQL desplegados en AWS EC2 no obstante el DNS que identifica públicamente la EC2 Free tier cambia dinámicamente por la propia naturaleza del servicio. Por esta razón, se recomienda adquirir el servicio de Route53 para poder registrar un dominio estático y por consiguiente no se requeriría actualizar las conexiones de Rstudio y Powershell.

Las librerías gráficas estadísticas tanto en Rstudio y en Google Collab Python son de fácil uso, despliegue y ayudan a la rápida exploración y visualización de datos

Bibliografía

- Hoja resumen de Shiny: <https://shiny.rstudio.com/articles/cheatsheet.html>
- Funciones Rstudio-Shiny: <https://shiny.rstudio.com/reference/shiny/1.6.0/>
- Irizarry, R. A. (2019). Introduction to data science: Data analysis and prediction algorithms with R. CRC Press: <https://rafalab.github.io/dsbook>
- AWS EC2 Service: https://aws.amazon.com/es/ec2/?trk=58ace84c-cd27-448f-9f64-ec1187db737b&sc_channel=ps&sc_campaign=acquisition&sc_medium=ACQ-P|PS-GO|Brand|Desktop|SU|Compute|EC2|LATAMO|ES|Text&sc_kwid=AL!4422!3!590500029733!e!!g!!aws%20ec2&ef_id=Cj0KCQjwma6TBhDIARIsAOKuANxKSCLz9mI94wIRc7ddYjXGrTZJ-6GJAYryvit2tA0g0TUyCmQKp8aAvhpEALw_wcB:G:s&s_kwid=AL!4422!3!590500029733!e!!g!!aws%20ec2
- Documentación Mongo Atlas Service : <https://www.mongodb.com/atlas>

Anexos

- GitRepository: <https://github.com/u712596/TrabajoFinal>

- CollabRepository Tercer Caso: https://colab.research.google.com/drive/1-XuGPiT2Hlv_KU48hFXHGWNnO6S6Mv14?usp=sharing

Shiny links Section:

<https://kylxjac.shinyapps.io/clinica/>