Minería de datos: PRA1 - Selección y preparación de un juego de datos

Autor: Jesús González Leal

Mayo 2021

Contents

Introducción

Objetivos analíticos

El mundo moderno de principios de este siglo XXI ha experimentado un aumento del comercio internacional. Más aun, en estos momentos en los que la pandemía del Covid asola nuestro mundo, el **e-commerce** ha experimentado un aumento considerable en su crecimiento. Este cambio evidencia una tendencia en la sociedad en cuanto a hábitos de consumo se refiere. Según la consultora eMarketer, especializada en comercio electrónico, en un estudio realizado antes de la pandemia, ya estimaba un crecimiento en las ventas de retail online para el año 2020 del 12, 5%, con una proyección para el 2023 de los 40.120 millones de dólares a nivel mundial

(fuente: Expansión economía digital - https://www.expansion.com/economia-digital/2020/08/20/5f3d852f468aeb11628b45c3.html).

El crecimiento en las ventas realizadas mediante portales web, exige la recogida de información y el tratamiento de **grandes volúmenes de datos**, a la par de una exigencia en los tiempos de respuesta, que permitan

adaptarse rápidamente a los cambios del sector. Es por ello, que se hace necesario un procesamiento de datos los más industrializado posible.

El objetivo de nuestro estudio se centrará en procesar un dataset que cuente con un volumen considerable de de transacciones, demostrando que es posible su procesamiento en corto plazo, y respondiendo a preguntas analíticas que permitan ayudar a determinar las estrategias de negocio.

Descripción del dataset

Para la realización de nuestro estudio, hemos contado con un dataset que contiene transacciones de venta on-line de una empresa con base en UK del sector retail, ocurridas entre diciembre del 2019 y diciembre del 2011. El dataset, lo componen más de un millón de registros con 8 atributos que describirán la temporalidad de la transacción, el artículo, el usuario que ha realizado la compra, así como su país de procedencia y la cantidad y el precio unitario de la venta.

El dataset trabajado pertenece a UCI - Machine Learning Repository, y fue donado en Septiembre del 2019. En el siguiente link puede encontrarse la información referente a la fuente de datos, así como el enlace para su descarga.

https://archive.ics.uci.edu/ml/datasets/Online+Retail+II

Debido al componente multinacional del dataset, que contiene transacciones de más de 40 países, se enriquecerá con datos aportados por el Banco Mundial. En concreto, buscaremos el Producto Interior Bruto de los países en los años de observación, con el fin de determinar **si un mayor o menos PIB** influye en el volumen de artículos comprados.

La elección del dataset se encuentra alienada con el problema de negocio planteado en la introducción. Permite la aplicación de algoritmos de clusterización, tanto supervisados, como no supervisados, así como de reglas de asociación, con el fin de poder responder a las preguntas que negocio determine.

Estudio Preliminar de los datos

En este capítulo, vamos a realizar la carga del dataset y a realizar un estudio que nos permita tener una idea a alto nivel de sus particularidades.

Carga de librerías

```
#library(ggplot2)
library(lubridate)

#library(gridExtra)
#library(kableExtra)
library(ggbiplot)
library(dplyr)
```

Carga del dataset

```
# Parameters
path <- '.\\data'
file <- 'online_retail_II.xlsx'
sheet_names <- readxl::excel_sheets(paste0(path,'\\', file))

# Load excels sheets into dataframe
for (i in 1:length(sheet_names)){
```

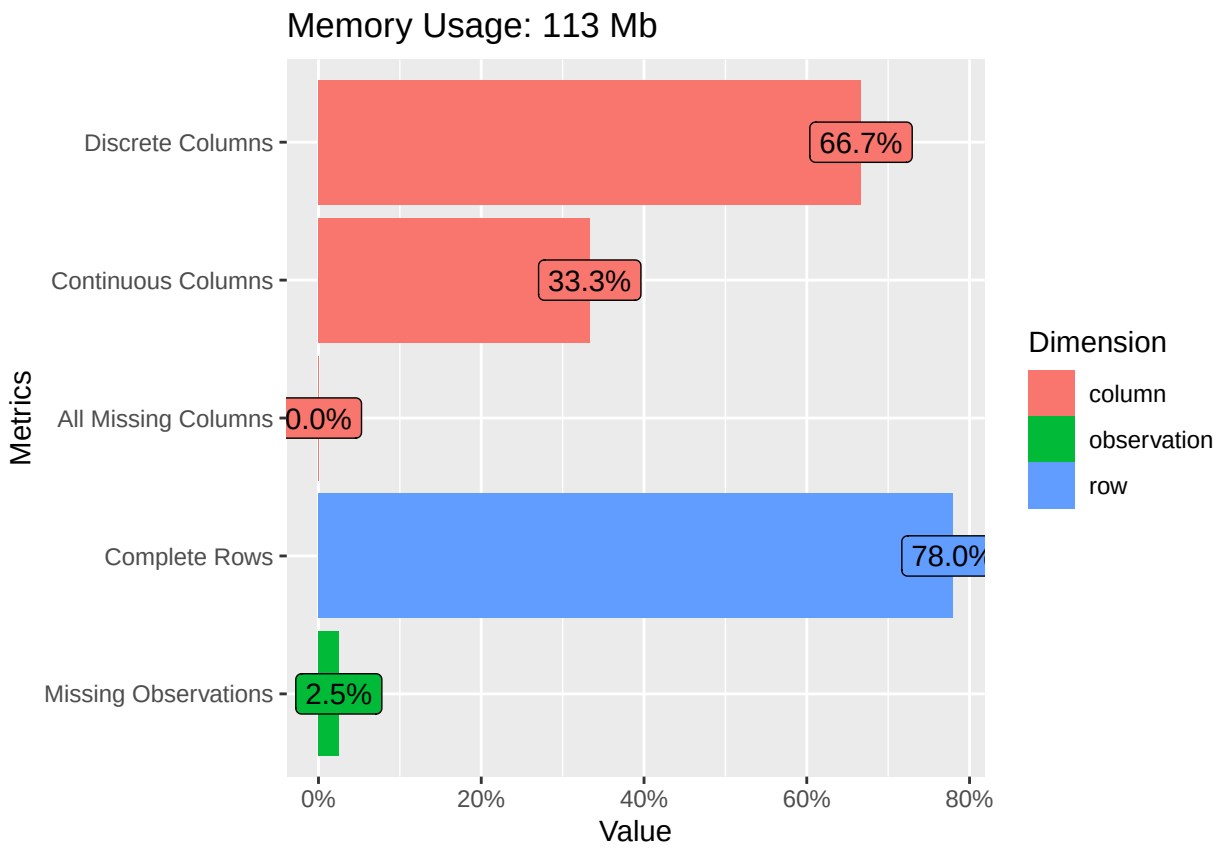
```
  df <- readxl::read_excel(path= paste0(path,'\\', file), sheet=i,  col_names=T)
  df$sheet_name <- sheet_names[i]

  if (i==1){
    retail.tb <- df
  }

  retail.tb <- rbind(retail.tb, df)
}
```

```
cat(paste('Número de observaciones: ',
          dim(retail.tb)[1], '; Número de variables: ',
          dim(retail.tb)[2]))
```

```
## Número de observaciones:  1592832 ; Número de variables:  9
```

El dataset tiene un ancho de 9 variables y un largo de casi 1.6 Millones de filas.

```
DataExplorer::plot_intro(retail.tb)
```



El gráfico superior nos muestra una visión rápida de las características del dataset. Posteriormente trataremos los valores nulos.

Vamos a revisar las primera líneas del dataset.

```
retail.tb
```

```
## # A tibble: 1,592,832 x 9
##    Invoice StockCode Description              Quantity InvoiceDate         Price
##    <chr>   <chr>     <chr>                       <dbl> <dttm>              <dbl>
```

```
##  1 489434  85048     "15CM CHRISTMAS GLASS B~       12 2009-12-01 07:45:00  6.95
##  2 489434  79323P    "PINK CHERRY LIGHTS"           12 2009-12-01 07:45:00  6.75
##  3 489434  79323W    "WHITE CHERRY LIGHTS"          12 2009-12-01 07:45:00  6.75
##  4 489434  22041     "RECORD FRAME 7\" SINGL~       48 2009-12-01 07:45:00  2.1
##  5 489434  21232     "STRAWBERRY CERAMIC TRI~       24 2009-12-01 07:45:00  1.25
##  6 489434  22064     "PINK DOUGHNUT TRINKET ~       24 2009-12-01 07:45:00  1.65
##  7 489434  21871     "SAVE THE PLANET MUG"          24 2009-12-01 07:45:00  1.25
##  8 489434  21523     "FANCY FONT HOME SWEET ~       10 2009-12-01 07:45:00  5.95
##  9 489435  22350     "CAT BOWL"                     12 2009-12-01 07:46:00  2.55
## 10 489435  22349     "DOG BOWL , CHASING BAL~       12 2009-12-01 07:46:00  3.75
## # ... with 1,592,822 more rows, and 3 more variables: Customer ID <dbl>,
## #   Country <chr>, sheet_name <chr>
```

A continuación, se muestra la descripción de las variables.

- **Invoice**. Código de identificación de la transacción. Gracias a este campo, podremos conocer para la analítica, respuestas sobre las transacciones con mayor venta, en relación a país, y temporalidad. También nos permitirá realizar asociaciones de productos del tipo, si compró un determinado producto, existe una probabilidad de que compre estos otros.
- **StockCode**. Código de identificación del producto.Al igual que el anterior, nos puede servir para establecer una analítica que responda a preguntas sobre los productos más vendidos y cuando se han producido estas ventas.
- **Description**. Descripción informativa del producto.
- **Quantity**. Cantidad de productos vendidos.
- **InvoiceDate**. Fecha y hora de la transacción. Tenemos una granularidad de minutos. Posteriormente, extraeremos los componentes principales de esta variable, fecha, mes año, para poder trabajar con un subset de datos más manejable.
- **Price**. Precio del producto vendido. Nuestro alcance dentro de la analítica se centrará en las ventas. Por lo tanto, a continuación convertiremos las variables Cantidad y Precio a una única variable que exprese la cantidad vendida.
- **Customer**. Id. Identificador del cliente. Gracias a esta variable, podremos contestar a preguntas sobre los mejores clientes, que nos permitirían realizar campañas de seguimiento especial.
- **Country**. País de la transacción. Al igual que la anterior variable, esta variable nos permitirá contestar a preguntas relativas a los países que concentran el core de las ventas de la empresa.
- **Sheet_name**. Variable de control que contiene el nombre de la página del fichero.

Data Wrangling

En este apartado nos concentramos en comprobar la limpieza y normalización del dataset, asegurándonos de dejarlo preparado para su análisis descriptivo.

Primeramente, vamos a examinar si en el dataset contiene observaciones (registros), duplicados .

```
a <- length(retail.tb)
b <- length(unique(retail.tb))
if (a==b) {cat('No existen registros duplicados')}
```

```
## No existen registros duplicados
```

Como siguiente paso, vamos a realizar una Exploración del tipo de datos de las variables.

```
glimpse(retail.tb)
```

```
## Rows: 1,592,832
## Columns: 9
## $ Invoice       <chr> "489434", "489434", "489434", "489434", "489434", "48943~
## $ StockCode     <chr> "85048", "79323P", "79323W", "22041", "21232", "22064", ~
## $ Description   <chr> "15CM CHRISTMAS GLASS BALL 20 LIGHTS", "PINK CHERRY LIGH~
## $ Quantity      <dbl> 12, 12, 12, 48, 24, 24, 24, 10, 12, 12, 24, 12, 10, 18, ~
## $ InvoiceDate   <dttm> 2009-12-01 07:45:00, 2009-12-01 07:45:00, 2009-12-01 07~
## $ Price         <dbl> 6.95, 6.75, 6.75, 2.10, 1.25, 1.65, 1.25, 5.95, 2.55, 3.~
## $ `Customer ID` <dbl> 13085, 13085, 13085, 13085, 13085, 13085, 13085, 13085, ~
## $ Country       <chr> "United Kingdom", "United Kingdom", "United Kingdom", "U~
## $ sheet_name    <chr> "Year 2009-2010", "Year 2009-2010", "Year 2009-2010", "Y~
```

En el caso de **Invoice**, la primera variable, aunque aparentemente son numéricos, vemos que su tipo de datos es literal. Vamos a explorar si podemos realizar una transformación a numérico entero.

```
retail.tb$Invoice[stringr::str_which(head(retail.tb$Invoice, 1000), '^[A-Z]')]
```

```
##   [1] "C489449" "C489449" "C489449" "C489449" "C489449" "C489449" "C489449"
##   [8] "C489449" "C489449" "C489459" "C489459" "C489459" "C489459" "C489459"
##  [15] "C489459" "C489459" "C489459" "C489459" "C489459" "C489459" "C489459"
##  [22] "C489459" "C489476" "C489503" "C489503" "C489504" "C489518" "C489518"
##  [29] "C489518" "C489524" "C489527" "C489527" "C489527" "C489528" "C489530"
##  [36] "C489534" "C489535" "C489535" "C489538" "C489538" "C489541" "C489543"
```

Vemos que efectivamente, esta variable contiene algunos valores codificados como literales, por lo que **no realizamos** ningún cambio en el tipo de datos.

"**Custumer ID**' incorpora un espacio en medio del nombre. Aunque de entrada ésto no representa ningún problema, vamos a eliminar el espacio para una mejor normalización del nombre del campo.

```
retail.tb$CustomerID <-retail.tb$`Customer ID`
retail.tb$`Customer ID` <- NULL
head(unique(retail.tb$CustomerID))
```

```
## [1] 13085 13078 15362 18102 12682 18087
```

A la variable **Country**, vamos a cambiarla de tipo a factor.

```
retail.tb$Country <- as.factor(retail.tb$Country)
unique(retail.tb$Country)
```

```
##  [1] United Kingdom       France               USA
##  [4] Belgium              Australia            EIRE
##  [7] Germany              Portugal             Japan
## [10] Denmark              Nigeria              Netherlands
## [13] Poland               Spain                Channel Islands
## [16] Italy                Cyprus               Greece
## [19] Norway               Austria              Sweden
## [22] United Arab Emirates Finland              Switzerland
## [25] Unspecified          Malta                Bahrain
## [28] RSA                  Bermuda              Hong Kong
## [31] Singapore            Thailand             Israel
## [34] Lithuania            West Indies          Lebanon
## [37] Korea                Brazil               Canada
## [40] Iceland              Saudi Arabia         Czech Republic
## [43] European Community
```

```
## 43 Levels: Australia Austria Bahrain Belgium Bermuda Brazil ... West Indies
```

Podemos observar que la salida del dataset lo componen ventas de 43 países diferentes.

Enriqueciendo el dataset

La variable InvoiceDate contiene el datetime de la transacción. Como comentamos anteriormente, nos es conveniente poder enriquecer el dataset descomponiendo la variable en la fecha para mejorar el análisis.

```
retail.tb$Date <- date(retail.tb$InvoiceDate)
retail.tb$Year <- year(retail.tb$InvoiceDate)
retail.tb$Month <- month(retail.tb$InvoiceDate)
retail.tb$YearMonth <- paste0(retail.tb$Year, retail.tb$Month)
retail.tb[1:5, c('InvoiceDate','Year','Month','YearMonth')]
```

```
## # A tibble: 5 x 4
##   InvoiceDate          Year Month YearMonth
##   <dttm>              <dbl> <dbl> <chr>
## 1 2009-12-01 07:45:00  2009    12 200912
## 2 2009-12-01 07:45:00  2009    12 200912
## 3 2009-12-01 07:45:00  2009    12 200912
## 4 2009-12-01 07:45:00  2009    12 200912
## 5 2009-12-01 07:45:00  2009    12 200912
```

Se han añadido columnas con el año, el mes y la concatenación del año, mes. Posteriormente, nos va a permitir agregar los datos por año y por mes, facilitando el análisis.

En el dataset tenemos también las variables **Price** y **Quantity**. Como comentamos anteriormente, y veremos más adelante, en el análisis de posibles outliers nos va a interesar realizar la analítica por la venta. Construimos la variable de observación Sale.

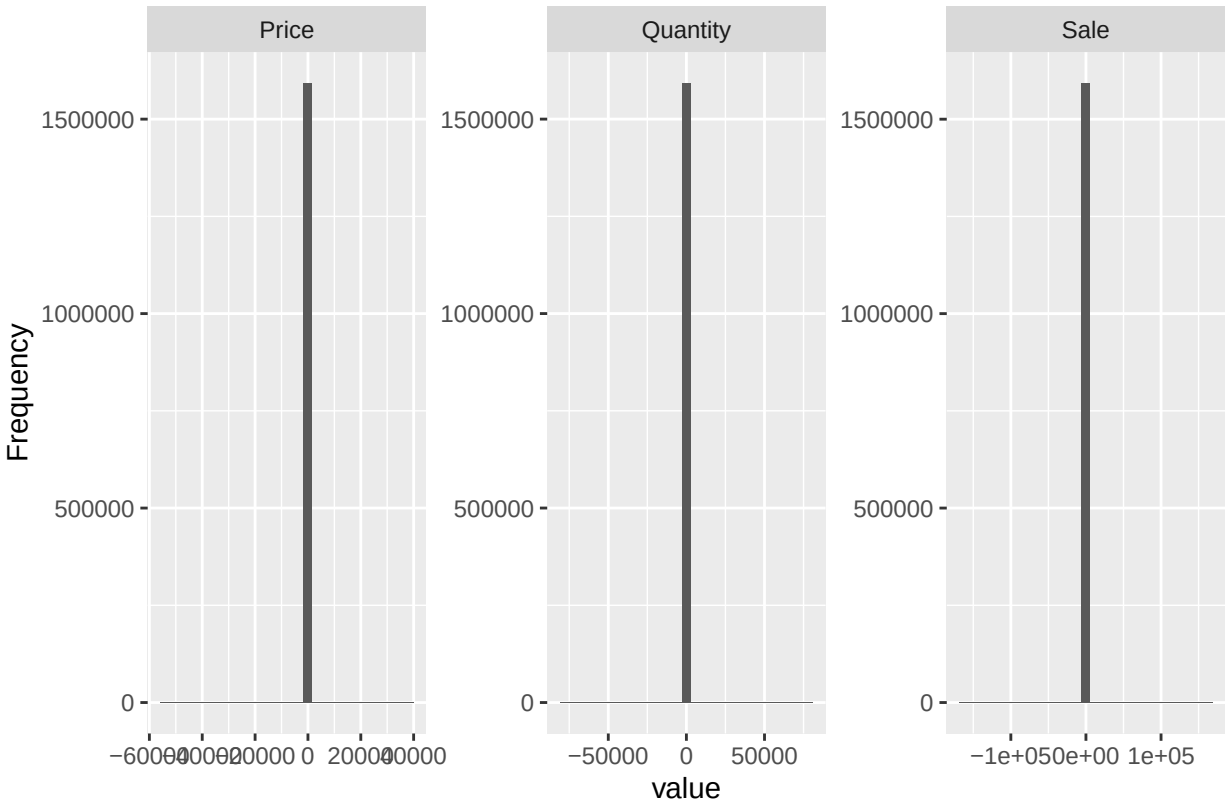
```
retail.tb$Sale <- retail.tb$Quantity * retail.tb$Price
head(unique(retail.tb$Sale),10)
```

```
##  [1]  83.40  81.00 100.80  30.00  39.60  59.50  30.60  45.00  98.10  17.85
```

El dataset que manejamos en este estudio se ha reducido a sólo una variable cuantitativa (producto de las otras dos, y por tanto con una alta correlación).

Finalmente, examinemos la distribución de los datos

```
DataExplorer::plot_histogram(retail.tb[, c('Price', 'Quantity', 'Sale')])
```



Vemos que las tres variables muestran casi en su totalidad, valores pequeños.

Carga de datos del World Bank

Debido a que uno de los objetivos de este estudio es utilizar métodos como el PCA o el SVD para encontrar la influencia de las variables en el dataset, vamos a enriquecerlo añadiendo datos del Producto Interior Bruto (PIB) de los países, durante los años de observación. Para ello hemos acudido a los datos abiertos del Banco Mundial.

Los datos cargados y sus metadatos pueden ser consultados en el siguiente enlace:

https://data.worldbank.org/indicator/NY.GDP.PCAP.CD?end=2011&start=2009

```
file <- 'PIB_World.csv'
pib.world <- read.csv(paste0(path,'\\', file), sep=',',
                      skip= 4, stringsAsFactors= F, header=T)
pib.world <- pib.world[, c('Country.Name', 'X2009', 'X2010', 'X2011')]
names(pib.world) <- c('Country', '2009', '2010', '2011')
# Modificado United States por USA
pib.world[pib.world$Country=='United States',]$Country <-'USA'
head(pib.world)
```

```
##       Country      2009      2010       2011
## 1       Aruba 24630.454 23512.603 24985.9933
## 2 Afghanistan   438.076   543.303   591.1628
## 3      Angola  3122.781  3587.884  4615.4680
## 4     Albania  4114.140  4094.350  4437.1429
## 5     Andorra 43503.186 40852.667 43335.3289
## 6  Arab World  5180.581  5926.713  6867.6002
```

Tenemos el indicador por año desnormalizado (un año por columna). Necesitamos realizar un pre-tratamiento

para hacer los datos compatibles con la unión con nuestro dataset de ventas.

```
col_names <- c('Country', 'PIB', 'Year')
pib.world.t <- data.frame(matrix(ncol=3, nrow=0))
names(pib.world.t) <- col_names
for (i in 1:nrow(pib.world)){
  df <- (t(pib.world[i, 2:4]))
  df <- cbind(rep(pib.world[i,1], 3), df)
  df <- cbind( df, as.integer(rownames(t(pib.world[i, 2:4]))))
  df <- as.data.frame(df, col.names = col_names)
  rownames(df) <-NULL
  names(df) <- col_names

  pib.world.t <- rbind(pib.world.t, df)
}
pib.world.t$PIB <- as.numeric(pib.world.t$PIB)
pib.world.t$Year <- as.integer(pib.world.t$Year)
head(pib.world.t)
```

```
##       Country        PIB Year
## 1       Aruba 24630.4537 2009
## 2       Aruba 23512.6026 2010
## 3       Aruba 24985.9933 2011
## 4 Afghanistan   438.0760 2009
## 5 Afghanistan   543.3030 2010
## 6 Afghanistan   591.1628 2011
```

Aunque ya tenemos la tabla de datos preparada para agregarla a los datos de venta, lo realizaremos posteriormente cuando tengamos un dataset agregado.

Detección de valores nulos

En nuestra siguiente sección, vamos a detectar valores nulos en el dataset.

```
sapply(retail.tb, function(x) sum(is.na(x)))
```

```
##     Invoice   StockCode Description    Quantity InvoiceDate       Price
##           0           0        7310           0           0           0
##     Country  sheet_name  CustomerID        Date        Year       Month
##           0           0      350934           0           0           0
##   YearMonth        Sale
##           0           0
```

Dos variables presentan valores desconocidos, Description y CustomerId. Asignamos un valor conocido a todos los identificadores de cliente (CustomerID) y de artículo (Description), que vienen en blanco. Aunque queda fuera del alcance de este estudio, el que un dataset contenga valores nulos se debe de ver como una incidencia dentro del flujo. Por ello, es necesario realizar una analítica centrándonos en estos valores, investigando si la causa se debe, por ejemplo, a un error en la entrada de datos, con el fin de subsanarlo.

```
retail.tb$CustomerID[is.na(retail.tb$CustomerID)==T]  <- 999999
retail.tb$Description[is.na(retail.tb$Description)==T] <- 'Artículo desconocido'

sapply(retail.tb, function(x) sum(is.na(x)))
```

```
##     Invoice   StockCode Description    Quantity InvoiceDate       Price
##           0           0           0           0           0           0
##     Country  sheet_name  CustomerID        Date        Year       Month
```

```
##           0           0           0           0           0           0
##   YearMonth        Sale
##           0           0
```

Una vez finalizada la fase de limpieza de los datos, mostramos el dataset resultante

```
retail.tb
```

```
## # A tibble: 1,592,832 x 14
##    Invoice StockCode Description     Quantity InvoiceDate         Price Country
##    <chr>   <chr>     <chr>              <dbl> <dttm>              <dbl> <fct>
##  1 489434  85048     "15CM CHRISTMA~       12 2009-12-01 07:45:00  6.95 United ~
##  2 489434  79323P    "PINK CHERRY L~       12 2009-12-01 07:45:00  6.75 United ~
##  3 489434  79323W    "WHITE CHERRY ~       12 2009-12-01 07:45:00  6.75 United ~
##  4 489434  22041     "RECORD FRAME ~       48 2009-12-01 07:45:00  2.1  United ~
##  5 489434  21232     "STRAWBERRY CE~       24 2009-12-01 07:45:00  1.25 United ~
##  6 489434  22064     "PINK DOUGHNUT~       24 2009-12-01 07:45:00  1.65 United ~
##  7 489434  21871     "SAVE THE PLAN~       24 2009-12-01 07:45:00  1.25 United ~
##  8 489434  21523     "FANCY FONT HO~       10 2009-12-01 07:45:00  5.95 United ~
##  9 489435  22350     "CAT BOWL"            12 2009-12-01 07:46:00  2.55 United ~
## 10 489435  22349     "DOG BOWL , CH~       12 2009-12-01 07:46:00  3.75 United ~
## # ... with 1,592,822 more rows, and 7 more variables: sheet_name <chr>,
## #   CustomerID <dbl>, Date <date>, Year <dbl>, Month <dbl>, YearMonth <chr>,
## #   Sale <dbl>
```

Detección de valores extremos

Realizamos un análisis de estadísticas básicas del juego de datos seleccionado.

```
summary(retail.tb)
```

```
##    Invoice            StockCode          Description           Quantity
##  Length:1592832     Length:1592832     Length:1592832     Min.   :-80995.00
##  Class :character   Class :character   Class :character   1st Qu.:     1.00
##  Mode  :character   Mode  :character   Mode  :character   Median :     3.00
##                                                           Mean   :    10.07
##                                                           3rd Qu.:    10.00
##                                                           Max.   : 80995.00
##
##   InvoiceDate                      Price                     Country
##  Min.   :2009-12-01 07:45:00   Min.   :-53594.36   United Kingdom:1467182
##  1st Qu.:2010-05-16 14:41:00   1st Qu.:     1.25   EIRE           :  27536
##  Median :2010-10-18 10:00:00   Median :     2.10   Germany        :  25753
##  Mean   :2010-11-01 17:35:42   Mean   :     4.66   France         :  20102
##  3rd Qu.:2011-04-03 14:24:00   3rd Qu.:     4.21   Netherlands    :   7909
##  Max.   :2011-12-09 12:50:00   Max.   : 38970.00   Spain          :   5089
##                                                    (Other)        :  39261
##   sheet_name          CustomerID          Date                 Year
##  Length:1592832     Min.   : 12346   Min.   :2009-12-01   Min.   :2009
##  Class :character   1st Qu.: 14375   1st Qu.:2010-05-16   1st Qu.:2010
##  Mode  :character   Median : 16153   Median :2010-10-18   Median :2010
##                     Mean   :232278   Mean   :2010-11-01   Mean   :2010
##                     3rd Qu.: 17977   3rd Qu.:2011-04-03   3rd Qu.:2011
##                     Max.   :999999   Max.   :2011-12-09   Max.   :2011
##
##      Month         YearMonth              Sale
```

```
##  Min.    : 1.000   Length:1592832     Min.    :-168469.60
##  1st Qu.: 4.000   Class :character   1st Qu.:      3.75
##  Median : 8.000   Mode  :character   Median :      9.90
##  Mean   : 7.478                      Mean   :     18.10
##  3rd Qu.:11.000                      3rd Qu.:     17.70
##  Max.   :12.000                      Max.   : 168469.60
##
```

Examinando el cuadro superior, vemos una variabilidad muy grande entre las variables Quantity, Price y Sale. Vamos a examinarlas con más atención.

```
summary(retail.tb$Quantity)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## -80995.00      1.00      3.00     10.07     10.00  80995.00
```

```
cat('Varianza de Sale: ', var(retail.tb$Sale))
```

```
## Varianza de Sale:  65781.12
```

Una variabilidad tan grande en los extremos nos indica que nuestro dataset contiene transacciones con **grandes ventas puntuales**, ya que anteriormente vimos que la práctica totalidad de los datos se concentran en pequeñas ventas . Vamos a visualizar las transacciones con cantidades mayores de 10.000 unidades

```
retail.tb[retail.tb$Quantity> 10000, c('YearMonth', 'Quantity', 'Price', 'Sale')]
```

```
## # A tibble: 15 x 4
##    YearMonth Quantity Price    Sale
##    <chr>        <dbl> <dbl>   <dbl>
##  1 20102        19152  0.1    1915.
##  2 20103        12960  0.1    1296
##  3 20103        12480  0.1    1248
##  4 20103        12960  0.1    1296
##  5 20103        12744  0.1    1274.
##  6 20105        10200  0         0
##  7 20102        19152  0.1    1915.
##  8 20103        12960  0.1    1296
##  9 20103        12480  0.1    1248
## 10 20103        12960  0.1    1296
## 11 20103        12744  0.1    1274.
## 12 20105        10200  0         0
## 13 20111        74215  1.04  77184.
## 14 201111       12540  0         0
## 15 201112       80995  2.08 168470.
```

Aunque las cantidades son grandes, debido al pequeño precio, las ventas no son elevadas.

No podemos considerar que estemos delante de outliers. De momento lo mantendremos, aunque es probable que tengamos que examinar estas ventas por separado ya que nos perjudicará su visión conjunta en la analítica descriptiva.

Vamos a examinar los datos extremos desde el punto de vista de la venta.

```
summary(retail.tb$Sale)
```

```
##       Min.    1st Qu.     Median       Mean    3rd Qu.       Max.
## -168469.60       3.75       9.90      18.10      17.70  168469.60
```

```
print(retail.tb[abs(retail.tb$Sale)> 50000,
                c('Invoice', 'YearMonth', 'Quantity', 'Price', 'Sale')])
```

```
## # A tibble: 6 x 5
##   Invoice YearMonth Quantity     Price     Sale
##   <chr>   <chr>        <dbl>     <dbl>    <dbl>
## 1 A506401 20104            1 -53594.    -53594.
## 2 A506401 20104            1 -53594.    -53594.
## 3 541431  20111        74215      1.04   77184.
## 4 C541433 20111       -74215      1.04  -77184.
## 5 581483  201112       80995      2.08  168470.
## 6 C581484 201112      -80995      2.08 -168470.
```

Es interesante ver que para cantidades superiores a 50k, todas las transacciones se encuentran **anuladas.** Es muy posible que se deban a **errores de caja** al marcar la cantidad. Aunque en una agregación por venta, quedan las ventas compensadas, vamos a eliminar las 4 últimas.

```
invoices <- c('541431', 'C541433', '581483', 'C581484')
print(retail.tb %>% filter(Invoice %in% invoices))
```

```
## # A tibble: 4 x 14
##   Invoice StockCode Description      Quantity InvoiceDate         Price Country
##   <chr>   <chr>     <chr>               <dbl> <dttm>              <dbl> <fct>
## 1 541431  23166     MEDIUM CERAMIC ~    74215 2011-01-18 10:01:00  1.04 United ~
## 2 C541433 23166     MEDIUM CERAMIC ~   -74215 2011-01-18 10:17:00  1.04 United ~
## 3 581483  23843     PAPER CRAFT , L~    80995 2011-12-09 09:15:00  2.08 United ~
## 4 C581484 23843     PAPER CRAFT , L~   -80995 2011-12-09 09:27:00  2.08 United ~
## # ... with 7 more variables: sheet_name <chr>, CustomerID <dbl>, Date <date>,
## #   Year <dbl>, Month <dbl>, YearMonth <chr>, Sale <dbl>
```

```
retail.tb <- retail.tb[!retail.tb$Invoice %in% invoices, ]

summary(retail.tb$Sale)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## -53594.36      3.75      9.90     18.10     17.70  38970.00
```

```
cat('Varianza de Sale: ', var(retail.tb$Sale))
```

```
## Varianza de Sale:  22663.84
```

Seguimos teniendo mucha variabilidad entre los valores máximos y mínimos respecto al primer y el tercer cuartil. Podemos abordar este asunto de dos formas diferentes:

- Estableciendo unas fronteras por encima del tercer y cuarto cuartil y eliminando los valores extremos. Para ello, podemos utilizar técnicas estadísticas para medir la dispersión.
- Discretizando las ventas mediante segmentos. Tiene la ventaja de que no perderemos información relevante, pero aumentamos la granularidad del dato.

En el siguiente apartado, exploraremos la segunda opción.

Agregación de los datos

Debido a que el número de registros es elevado, vamos a disminuir la granularidad del dataset agrupando los datos por año y mes

```
retail.tb.SalesByContryAndYearMonth <- retail.tb %>%
  select(YearMonth, Year, Month, Country, Sale)  %>%
  group_by(YearMonth, Year, Month, Country) %>%
  summarise(Sale = sum(Sale))
```

```
## `summarise()` has grouped output by 'YearMonth', 'Year', 'Month'. You can override using the `.groups
```

```
retail.tb.SalesByContryAndYearMonth
```

```
## # A tibble: 594 x 5
## # Groups:   YearMonth, Year, Month [25]
##    YearMonth  Year Month Country           Sale
##    <chr>     <dbl> <dbl> <fct>            <dbl>
##  1 200912     2009    12 Australia         100.
##  2 200912     2009    12 Austria          3997.
##  3 200912     2009    12 Belgium           895.
##  4 200912     2009    12 Channel Islands  1891.
##  5 200912     2009    12 Cyprus           6850.
##  6 200912     2009    12 Denmark          2769.
##  7 200912     2009    12 EIRE            38256.
##  8 200912     2009    12 Finland          1098.
##  9 200912     2009    12 France          11110.
## 10 200912     2009    12 Germany         19565.
## # ... with 584 more rows
```

Aunque hemos disminuido la granularidad, si examinamos a la variable Country en un año concreto, vemos que todavía es muy alta.

```
length(unique(filter(retail.tb.SalesByContryAndYearMonth,
                     Year==2010)$Country))
```

```
## [1] 40
```

En efecto, para el año 2010, tenemos 40 países que realizaron transacciones. Podemos resumir los datos viendo cuales han realizado un número mayor de transacciones y su volumen.

Con el fin de agregar aun más los datos, crearemos otro dataset que contenga los datos por año.

```
retail.tb.SalesByContryAndYear <- retail.tb %>%
  select(Year, Country, Sale)  %>%
  group_by(Year, Country) %>%
  summarise(Sale = sum(Sale))
```

```
## `summarise()` has grouped output by 'Year'. You can override using the `.groups` argument.
```

```
summary(retail.tb.SalesByContryAndYear[
  retail.tb.SalesByContryAndYear$Year == 2010, 'Sale'])
```

```
##       Sale
##  Min.   :     281
##  1st Qu.:    5238
##  Median :   12808
##  Mean   :  455706
##  3rd Qu.:   49725
##  Max.   :15610986
```

Al agrupar los datos por año, el valor máximo se nos ha disparado por encima de los 15M, estando muy lejos del tercer cuartil y de la media. Manejar estos número tal elevados no es útil, por lo que valor a realizar una conversión a miles de la variable Sale

```
retail.tb.SalesByContryAndYear <-
  retail.tb.SalesByContryAndYear %>%
  mutate(Sales.k = round(Sale / 1000,1))

retail.tb.SalesByContryAndYearMonth <-
```

```
  retail.tb.SalesByContryAndYearMonth %>%
  mutate(Sales.k = round(Sale / 1000,1))
```

Como último paso, vamos a añadir los datos del PIB del país, extraídos anteriormente del Banco Mundial.

```
retail.tb.SalesByContryAndYear <- merge(
  retail.tb.SalesByContryAndYear, pib.world.t, all.x = T)
```

```
retail.tb.SalesByContryAndYearMonth <- merge(
  retail.tb.SalesByContryAndYearMonth, pib.world.t, all.x = T)
```

Examinemos los países para los que no hemos conseguido cruzar los datos del PIB.

```
unique(
  retail.tb.SalesByContryAndYearMonth[
    is.na(
      retail.tb.SalesByContryAndYearMonth$PIB)==T, 'Country'])
```

```
## [1] Channel Islands    EIRE               Hong Kong          Korea
## [5] RSA                Unspecified        West Indies        European Community
## 43 Levels: Australia Austria Bahrain Belgium Bermuda Brazil ... West Indies
```

Existen diferentes técnicas para tratar estos datos, como la de imputar la media o eliminar los registros implicados. Nos vamos a decantar por la segunda opción.

```
retail.tb.SalesByContryAndYearMonth <-
  retail.tb.SalesByContryAndYearMonth[is.na(
    retail.tb.SalesByContryAndYearMonth$PIB)==F,]

retail.tb.SalesByContryAndYear <-
  retail.tb.SalesByContryAndYear[is.na(
    retail.tb.SalesByContryAndYear$PIB)==F,]
```

Ahora que ya tenemos los datasets preparados, vamos a realizar una clasificación del volumen de las ventas.

Discretización de los datos

A continuación vamos a discretizar la variable Sale, creando una serie de intervalos para los dos datasets creados anteriormente.

```
summary(retail.tb.SalesByContryAndYear$Sales.k)
```

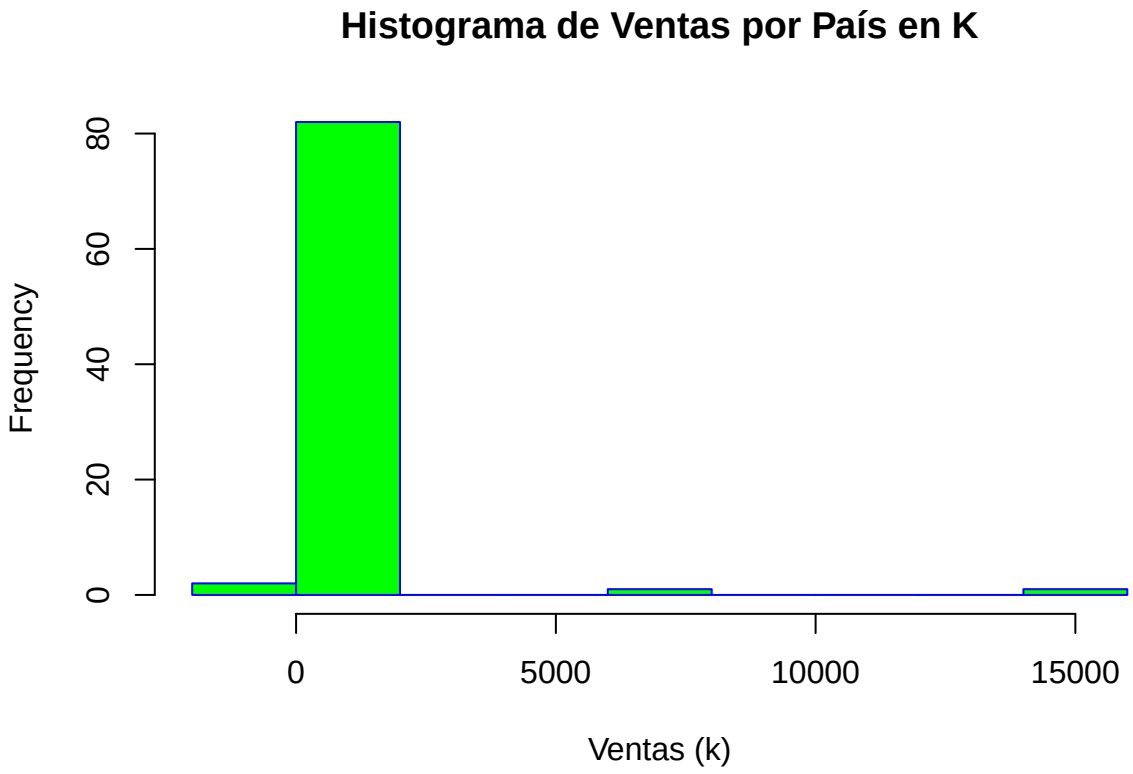
```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
##    -0.40     1.75     8.95   322.71    31.15 15611.00
```

```
cat(paste('Varianza de la variable Sale (en miles): '),
    var( retail.tb.SalesByContryAndYear$Sales.k))
```

```
## Varianza de la variable Sale (en miles):  3458646
```

Como hemos apreciado anteriormente, la variable que contiene las ventas presenta una variación muy alta entre los valores extremos. Examinemos en donde se concentran los datos.

```
hist(retail.tb.SalesByContryAndYear$Sales.k,
     main = 'Histograma de Ventas por País en K',
     xlab = 'Ventas (k)',
     border = 'blue',
     col = 'green'
     )
```



Vemos que los datos están concentrados en el extremo derecho, casi en un 100%. El mantener valores tan elevados, nos va a distorsionar en exceso, así que definiremos una clase para las ventas superiores a las 5000k, y para el resto realizaremos el proceso de discretización. De esta manera, conseguimos mantener estos valores elevados para la analítica, afectando lo menor posible al resto.

```
ClassNumber <- function(vec){
  # Determinamos los parámetros
  n <- length(vec)
  k <- c()
  # Raíz cuadrada
  k[1] <- ceiling(sqrt(n))

  # Sturges
  k[2] <- ceiling(1+ log(n, 2))

  # Scott
  #cw <- retail.tb.SalesByContryAndYear <-
  As = 3.5*sd(vec)*n^(-1/3) #Amplitud teórica
  k[3] = ceiling(diff(range(vec))/As)

  return(k)
}

k<- ClassNumber(retail.tb.SalesByContryAndYear$Sales.k)

print('Determinación del Número de clases: ')
```

```
## [1] "Determinación del Número de clases: "
```

```
cat(paste('Mediante la raíz cuadrada: '), k[1], '\n')
```

```
## Mediante la raíz cuadrada:  10
```

```
cat(paste('Mediante la regla de Sturges: '), k[2], '\n')
```

```
## Mediante la regla de Sturges:  8
```

```
cat(paste('Mediante la regla de Scott: '), k[3], '\n')
```

```
## Mediante la regla de Scott:  11
```

Nos quedaremos con la regla de Sturges, por ser la que nos ofrece menor valor, para construir los intervalos. Ahora miraremos la amplitud de los intervalos. La manera más sencilla es determinar la amplitud como una constante entre los intervalos.

```
data <- retail.tb.SalesByContryAndYear[retail.tb.SalesByContryAndYear$Sales.k <5000, ]
A = round(diff(range(data$Sales.k))/(k[2]-1),0)
A
```

```
## [1] 208
```

Para que sea más visual, hemos ajustado la amplitud para que no contenta decimales, aunque lo más correcto sería mantener la precisión, que en nuestro caso es de 1 decimal.

Seguidamente calculamos los extremos. El valor del extremos de la izquierda se construye a partir del valor mínimo de la serie, menos la mitad de la precisión.

```
L1 <- min(retail.tb.SalesByContryAndYear$Sales.k)-1 / (2*0.1)
L1
```

```
## [1] -5.4
```

Calculamos los límites de los intervalos:

```
L <- c()
L[1] <- L1
n_class <- k[2]-1
for (i in 2:n_class) {
  L[i] <- round(L[i-1] + A,0)
}
L[8]<- max(retail.tb.SalesByContryAndYear$Sales.k)
```

Definimos las etiquetas.

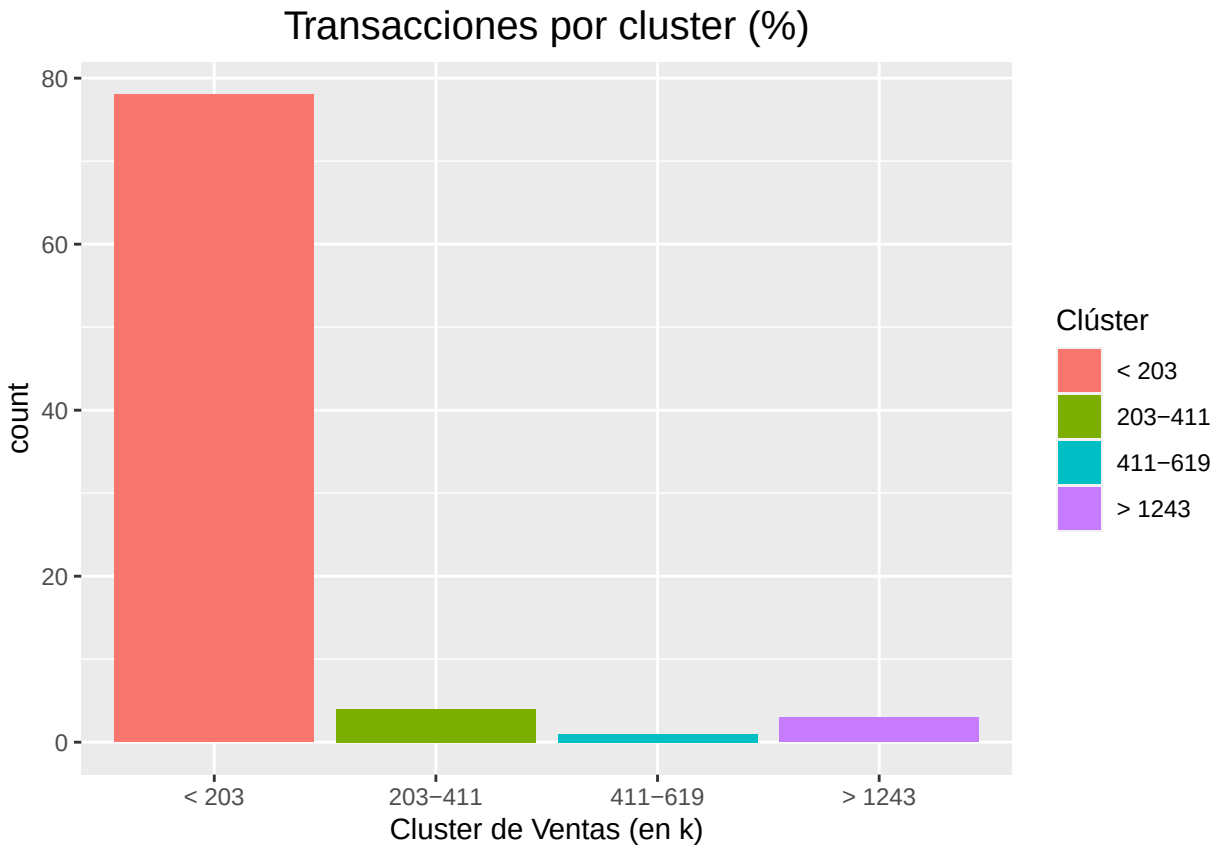
```
Sales.k.labels <- c('< 203', '203-411', '411-619', '619-827',
                    '827-1035', '1035-1243', '> 1243')
```

y realizamos los intervalos

```
retail.tb.SalesByContryAndYear$Sales.M <- cut(
  retail.tb.SalesByContryAndYear$Sales.k,
  breaks = L, labels = Sales.k.labels)
```

Visualicemos el resultado de la discretización.

```
ggplot(retail.tb.SalesByContryAndYear,
       aes(Sales.M, fill= Sales.M)) + geom_bar() + theme(plot.title = element_text(size = 15,
    hjust = 0.5)) +labs(title = "Transacciones por cluster (%)",
    x = "Cluster de Ventas (en k)", fill = "Clúster")
```



En el gráfico superior, constatamos que la gran mayoría de las transacciones realizadas por país y año, se concentran en el primer grupo.

Con el dataset ya preparado, es hora de analizar los datos para responder a las preguntas de negocio.

Enlaces externos: https://www.rpubs.com/JoanClaverol/488759

Análisis exploratorio

En este capítulo realizaremos el análisis que dará respuestas analíticas. Finalizamos el anterior capítulo viendo que un porcentaje muy importante de las transacciones se realizan para valores inferiores a los 203 k. Antes de entrar en detalle con estos valores, desde negocio querrán conocer que componen las transacciones importantes de la compañía.

```
retail.tb.SalesByContryAndYear[
  retail.tb.SalesByContryAndYear$Sales.M == '> 1243', ]
```

```
##    Year        Country     Sale Sales.k      PIB Sales.M
## 23 2009 United Kingdom  1455312  1455.3 38713.14  > 1243
## 61 2010 United Kingdom 15610986 15611.0 39435.84  > 1243
## 99 2011 United Kingdom  7511064  7511.1 42038.57  > 1243
```

En efecto, vemos que el mayor volumen de ventas lo tenemos en UK durante los tres años del estudio.

Ahora, centrándonos de nuevo en la primera clase, la que compone el grueso del negocio, vamos a ver cuantos, y que países son los que más contribuyen a las ventas.

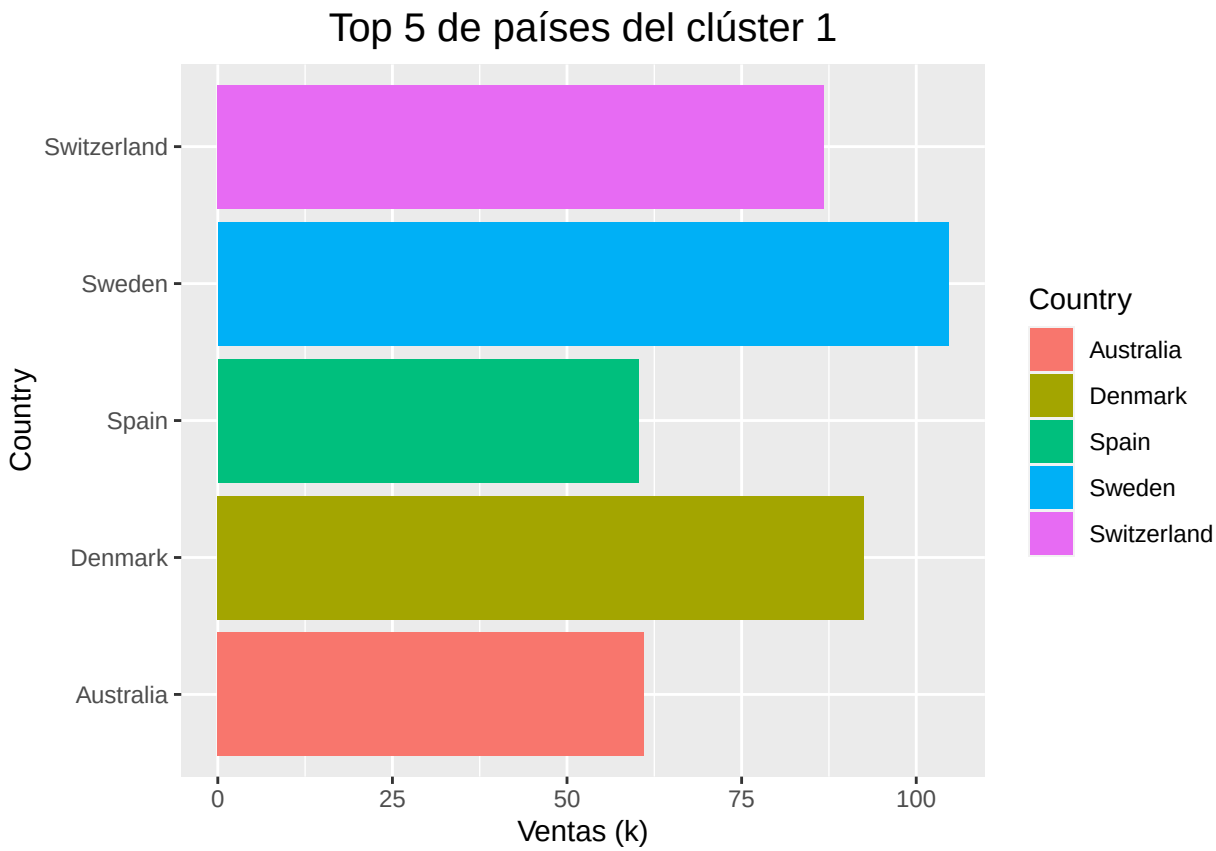
```
data <- retail.tb.SalesByContryAndYear[
  retail.tb.SalesByContryAndYear$Sales.M == '< 203', ]
```

```
length(unique(data$Country))
```

```
## [1] 34
```

Como es lógico, en esta categoría se concentran la mayoría de países. Por ello vamos a ver los 5 países que más intervienen en el año 2010,

```
data.top5 <- data %>%
  filter(Year == 2010) %>%
  arrange(desc(Sale)) %>%
  top_n(5, Sales.k)

ggplot(data.top5, aes(Sales.k, Country, fill= Country)) +
  geom_bar(stat="identity",na.rm=TRUE) + theme(plot.title = element_text(size = 15,
    hjust = 0.5)) +labs(title = "Top 5 de países del clúster 1",
    x = "Ventas (k)")
```



Vamos a examinar en ese año en cuanto se estima la contribución de estos países respecto al resto de las transacciones.

```
countries <- as.character(data.top5$Country)
```

```
total2010_top5 <- retail.tb.SalesByContryAndYear %>%
  filter(Country %in% countries & Year == 2010) %>%
  summarise(total2010_top5 = sum(Sales.k))

total2010 <- retail.tb.SalesByContryAndYear %>%
  filter(Year == 2010) %>%
```

```
  summarise(total2010 = sum(Sales.k))

cat(paste('El top 5 de países en 2010 contribuyen sólo en ',
          round(as.double(total2010_top5) /
                  as.double(total2010) * 100, 2), '% del total'))
```

```
## El top 5 de países en 2010 contribuyen sólo en  2.32 % del total
```

Por tanto, vemos que están las ventas muy estratificadas sobre todo el mundo.

¿Son los países más ricos, en los que estamos teniendo más ventas? Para responder a esta pregunta, vamos a utilizar el campo PIB importado desde el Banco Mundial, centrándonos en el análisis del año 2010, como hasta el momento.

```
retail.tb.SalesByContryAndYear %>%
  filter(Year == 2010) %>% top_n(10, PIB) %>% arrange(desc(Sales.k))
```

```
##    Year     Country      Sale Sales.k       PIB Sales.M
## 1  2010 Netherlands 506101.84   506.1  50950.03 411-619
## 2  2010      Sweden 104553.72   104.6  52869.04   < 203
## 3  2010     Denmark  92458.48    92.5  58041.40   < 203
## 4  2010 Switzerland  86812.94    86.8  74605.72   < 203
## 5  2010   Australia  61008.20    61.0  52022.13   < 203
## 6  2010      Norway  10975.64    11.0  87693.79   < 203
## 7  2010         USA   8829.24     8.8  48467.52   < 203
## 8  2010   Singapore   8075.54     8.1  47236.96   < 203
## 9  2010     Bermuda   2506.28     2.5 101875.28   < 203
## 10 2010      Canada   2433.32     2.4  47448.01   < 203
```

```
retail.tb.SalesByContryAndYear %>%
  filter(Year == 2010) %>% summarise(mean(Sales.k))
```

```
##   mean(Sales.k)
## 1      529.8879
```

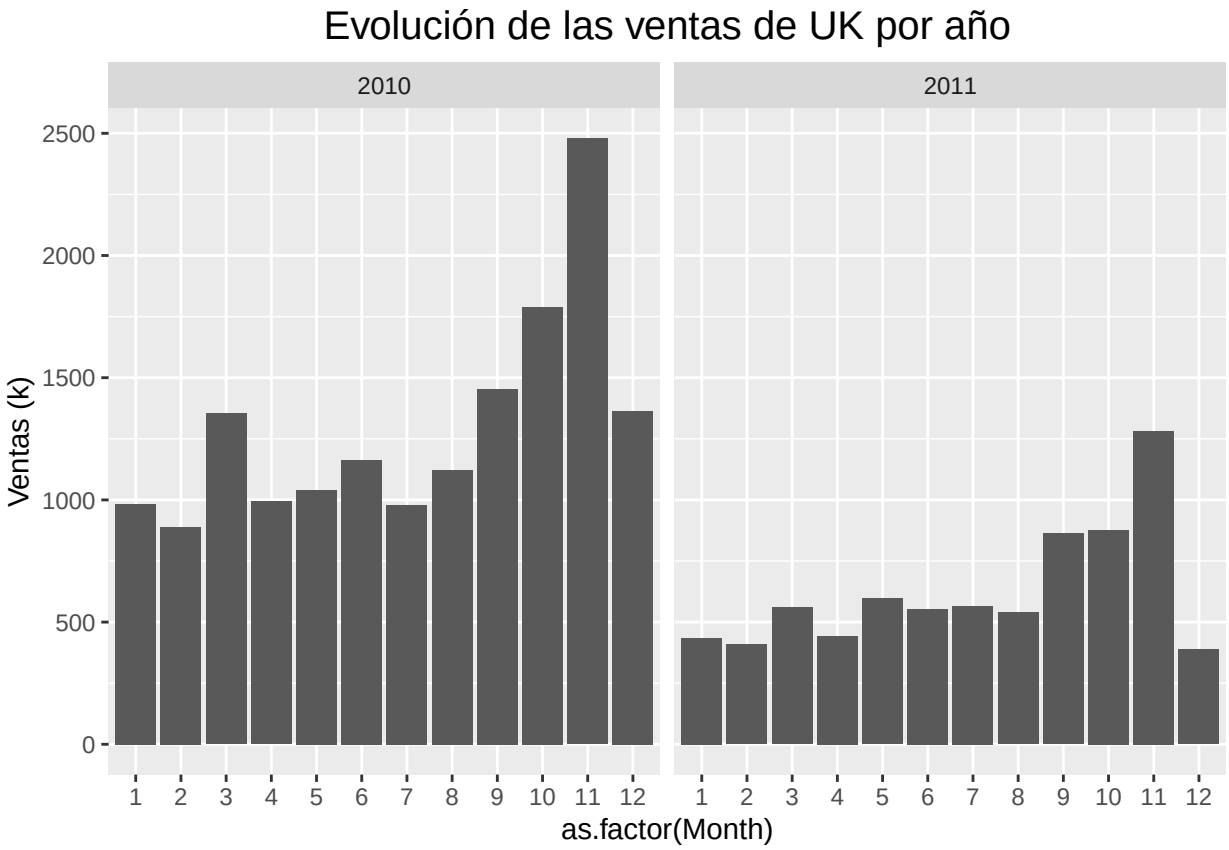
Si lo comparamos con la media, no vemos que el PIB influya significativamente en las ventas. De hecho, su correlación es muy baja.

```
cor(retail.tb.SalesByContryAndYear$Sales.k,
    retail.tb.SalesByContryAndYear$PIB)
```

```
## [1] 0.005056968
```

Hemos visto, que UK es el país en donde se realizan el grueso de las ventas. Vamos a visualizar mensualmente su evolución a lo largo de los años.

```
data <- retail.tb.SalesByContryAndYearMonth %>%
  filter(Year %in% c(2010, 2011) & Country == 'United Kingdom') %>%
  select(Country, Year, Month, Sales.k) %>% arrange(Month)

ggplot(data, aes(y=Sales.k, x= as.factor(Month))) +
  geom_bar(stat="sum") +
    facet_wrap(~Year) +
    theme(plot.title =
            element_text(size = 15, hjust = 0.5),
          legend.position = "none") +
    labs(title = "Evolución de las ventas de UK por año",
         y = "Ventas (k)")
```



Reducción de la dimensionalidad

Bajo el nombre de reducción de la dimensionalidad, se encuentran un conjunto de técnicas estadísticas supervisadas y no supervisadas, que buscan escoger las características principales de un dataset con el fin de que al seleccionar estas características, el dataset resultante sea inferior en número de dimensiones, y la información perdida sea mínima.

En este capítulo aplicaremos dos métodos de reducción de la dimensionalidad a nuestro dataset, el PCA y el SVD.

Estudio PCA

El Principal Component Analysis (PCA) es un método estadístico de reducción de la dimensionalidad del tipo no supervisado, que se basa en definir unos vectores principales sobre el espacio geométrico que forman los puntos del dataset. Cada componente sigue la dirección en la que los datos muestran una mayor varianza y que a la vez se busca, que esté lo mínimamente correlacionada con la componente anterior. Una no correlación perfecta entre componentes se muestra en que las direcciones de los vectores son ortogonales.

El algoritmo PCA es uno de los más utilizados para la compresión de datos y la eliminación de redundancias. Como inconveniente mencionar que es un algoritmo altamente sensible a los outliers, cosa que se ha de tener muy en cuenta a la hora de utilizarlo.

Vamos a realizar el análisis de componentes principales (PCA) sobre el dataset.

```
set.seed(1234)
col_names <- c('Year', 'Month', 'Sales.k', 'PIB')

cor(retail.tb.SalesByContryAndYearMonth[, col_names])
```

```
##                Year       Month     Sales.k         PIB
## Year     1.00000000 -0.21413217 -0.07303725  0.14563734
## Month   -0.21413217  1.00000000  0.04077309  0.01405668
## Sales.k -0.07303725  0.04077309  1.00000000 -0.03703957
## PIB      0.14563734  0.01405668 -0.03703957  1.00000000
```

Vemos que existe una cierta correlación entre año y mes, lo cual es lógico, pero no con la variable de ventas ni PIB.

Antes de lanzar el análisis vamos a estandarizar las variables para tener media cero y desviación típica de 1. Este paso es conveniente, ya que la matriz de datos se encuentra formada por variables con diferentes magnitudes y rangos.

```
retail.SalesYearMonth.pca <- prcomp(
  na.omit(retail.tb.SalesByContryAndYearMonth[, col_names]),
  center = TRUE, scale. = TRUE)

summary(retail.SalesYearMonth.pca)
```

```
## Importance of components:
##                           PC1    PC2    PC3    PC4
## Standard deviation     1.1321 1.0074 0.9848 0.8566
## Proportion of Variance 0.3204 0.2537 0.2425 0.1835
## Cumulative Proportion  0.3204 0.5741 0.8165 1.0000
```

Debido a la escasa correlación entre las variables, la proporción de varianza no proporciona buenos resultados. Si nos fijamos en los valores acumulados, utilizando PC1, PC2 y PC3, podemos representar el casi el 82 % del modelo.

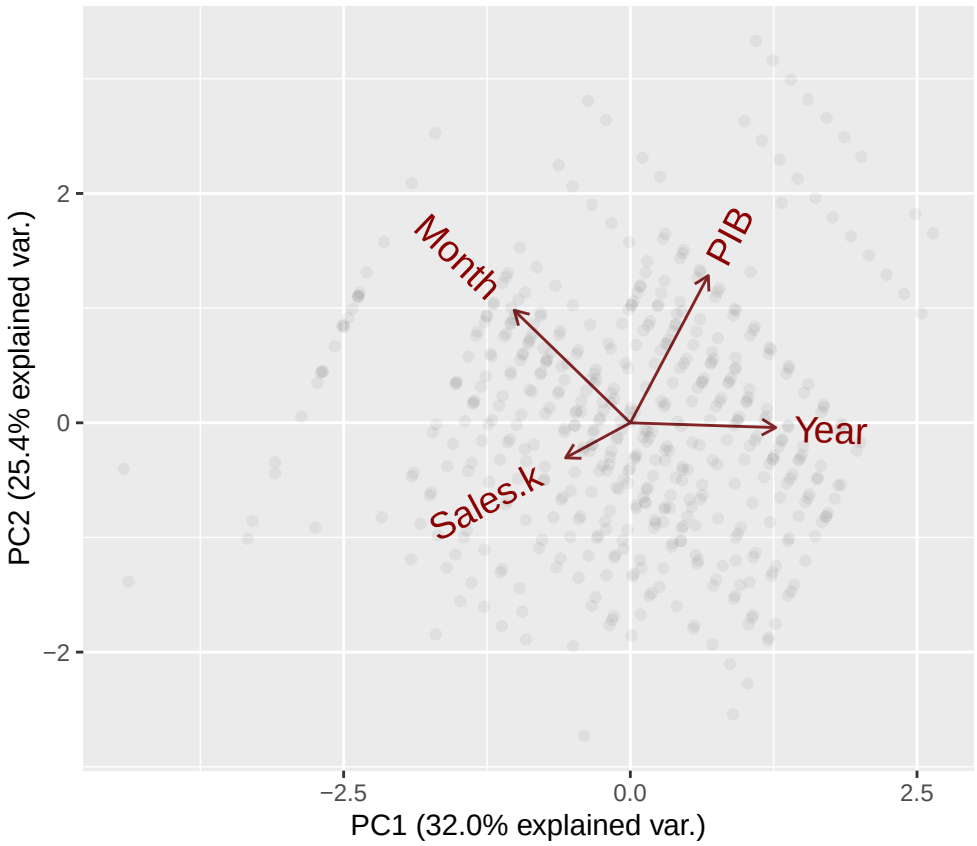
Veamos los vectores propios

```
retail.SalesYearMonth.pca$rotation
```

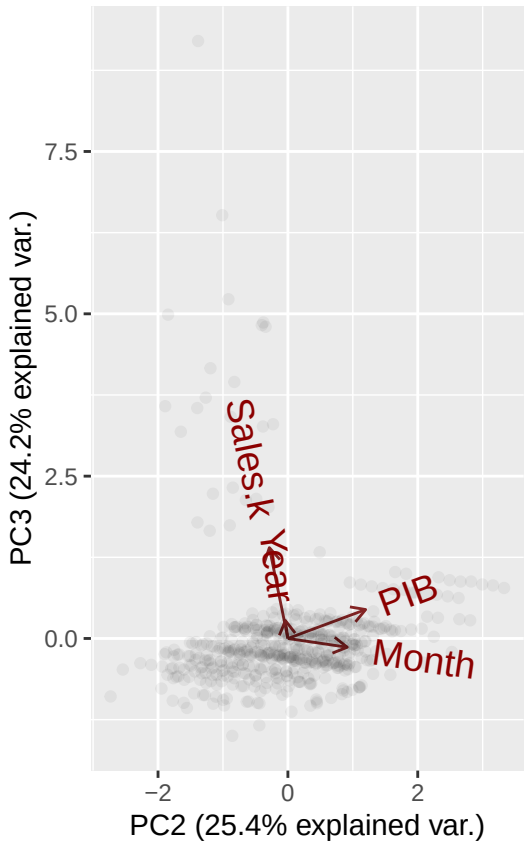
```
##                PC1         PC2        PC3         PC4
## Year     0.6862974 -0.02540031  0.1843622 -0.70310832
## Month   -0.5478765  0.59611979 -0.0901558 -0.57995212
## Sales.k -0.3057711 -0.18591443  0.9325829 -0.04721118
## PIB      0.3678740  0.78066116  0.2969369  0.40873626
```

Vamos a ver visualmente que componentes tienen mayor influencia

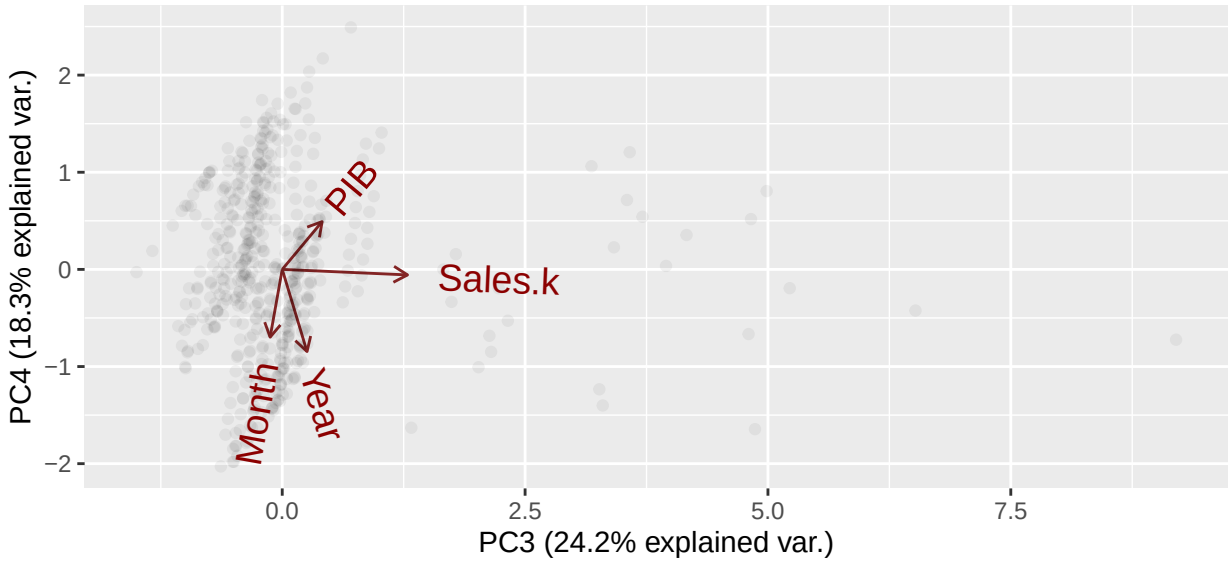
```
retail.pca.chart <- function(x, y){
  ggbiplot(retail.SalesYearMonth.pca, choices = x:y, obs.scale = 1, var.scale = 1,
           pc.biplot =TRUE, labels.size = 3, var.axes = TRUE,
           varname.size=5, alpha = 0.05)

}

par(mfrow= c(3,1))

#for(i in 1:3) {
#  print(retail.pca.chart(i,i+1))
#  }
p1 <- retail.pca.chart(1,2)
p2 <- retail.pca.chart(2,3)
p3 <- retail.pca.chart(3,4)
print(p1)
```


Month
PIB
Year
Sales.K
2
0
−2
−2.5
0.0
2.5
PC2 (25.4% explained var.)
PC1 (32.0% explained var.)

```
print(p2)
```




```
print(p3)
```



En los gráficos anteriores se puede ver lo siguiente.

- La primera componente es la que más contribuye a explicar el modelo en un 32 %.
- La segunda componente contribuye con un 25,4 %. Entre las dos componentes, el modelo queda definido en un 57,4 %.
- Todos los vectores tienen direcciones opuestas, y por su magnitud, las ventas son las que menos relevancia estadística tiene respecto a los vectores de temporalidad. Esto cambia en el último gráfico, en donde la componente de ventas tiene una magnitud mayor que el resto.

- Una mayor longitud de las fechas (su magnitud), nos expresa la importancia de la variable respecto al resto. Examinando las cuatro gráficas, no encontramos ninguna diferencia significativa.
- Como hemos visto anteriormente, con las 3 primeras componentes podemos explicar más del 80% del modelo.

Estudio SVD

El Singular Vector Descomposition (SVD) es una técnica matemática cuyo fin es el de transformar un conjunto de variables correlacionadas en otro conjunto sin correlación. Es utilizado como método de reducción de la dimensionalidad, indicando el número de dimensiones importantes que ayuden a simplificar el dataset, sin perder excesiva información.

La idea que subyace es que un set de vectores A, puede ser expresado en términos de la longitud de su proyección y de sus ejes ortogonales.

$$A = UDV'$$

En donde U es una matriz m x m, D es una matriz m x n llamada matriz diagonal, ya que todos sus valores son cero, menos la diagonal. Finalmente, V es una matriz n x n.

Parte del éxito de este algoritmo se basa en que la descomposición de la matriz de datos en sus valores singulares, gracias a la utilización de la computación paralela, tiene un ***performance* muy alto**. Por ello, este algoritmo no sólo se puede utilizar para descomponer las características principales del dataset, sino que también es empleado en sistemas de recomendación, procesamiento del lenguaje natural, en sistemas recomendadores y en compresión de imágenes.

```
set.seed(1234)

retail.svd <- (retail.tb.SalesByContryAndYearMonth[is.na(
  retail.tb.SalesByContryAndYearMonth$PIB)==F,
  c('Year', 'Month', 'Sale', 'PIB')])

dim(retail.svd)
```

```
## [1] 510   4
```

Nuestra matriz es del tamaño 510 x 4. Vamos a transformar esta matriz en 3, la matriz diagonal (d), y las matrices u y v. Conseguimos la transformación de las matrices a partir del dataset y mostramos su matriz diagonal.

```
A_svd <- svd(retail.svd)
A_svd$d
```

```
## [1] 5.499076e+06 1.053585e+06 1.704949e+04 8.006774e+01
```

Siendo V, la matriz de columnas ortonormales con el mismo espacio de columnas que la matriz original de nuestro dataset.

```
A_svd$v
```

```
##               [,1]          [,2]          [,3]          [,4]
## [1,] -1.902173e-03 -0.0386906376 -0.9992438774  3.330119e-03
## [2,] -7.074497e-06 -0.0001319005 -0.0033275002 -9.999945e-01
## [3,] -9.992298e-01  0.0392374793  0.0003828768  6.195851e-07
## [4,] -3.919321e-02 -0.9984805643  0.0387357011  3.084346e-06
```

Y finalmente, la matriz proyección, que como podemos comprobar, contiene la misma dimensionalidad que la matriz original.

```r
dim(A_svd$u)
```

```
## [1] 510   4
```

Enlaces Externo consultado:

http://thecooldata.com/es/2018/08/matrix-decomposition-image-compression-and-video-background-removal-using-r-part-1/

https://towardsdatascience.com/svd-8c2f72e264f

https://programmerclick.com/article/64901804965/#SVD_2

Conclusiones

A lo largo de este estudio, hemos trabajado bajo la idea de poder explotar datos provenientes del comercio electrónico. Se ha podido comprobar que es posible trabajar con R, en datasets que contienen un cierto volumen de datos. Posteriormente, nos hemos enfrentado al reto de adaptar y enriquecer los datos de nuestro dataset con otros provenientes del banco mundial.

El análisis exploratorio del dataset resultante, nos ha mostrado que UK es el principal país que provee ventas para el negocio. No obstante, el volumen de transacciones se extiende por más de 40 países. Hemos mostrado también, que el top 5 de países por volumen de ventas no supera el 4% del total. Este hecho apuntala que el negocio, salvo transacciones muy particulares, se encuentra muy repartido entre todos los países.

Se ha realizado un proceso de discretización, segmentando los datos según su volumen. Se ha optado por una solución matemática, para la determinación del número de clases. Aunque esta solución es válida a nivel académico, en un contexto de negocio, y a raíz de los resultados, se optaría por reclasificar manualmente el número de las clases, dando más importancia a las transacciones que anualmente se encuentran por debajo de los 100k.

Finalmente, se ha realizado dos procesos de reducción de la dimensionalidad, el PCA y el SVD. El no tener un dataset con muchas dimensiones, y la falta de correlación entre ellas ha hecho que el resultado no sea determinante para este caso en concreto.