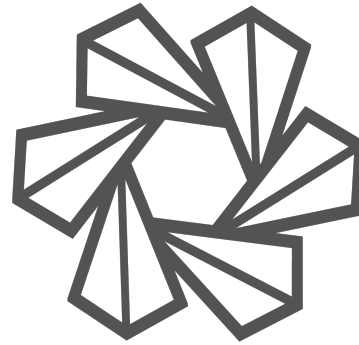
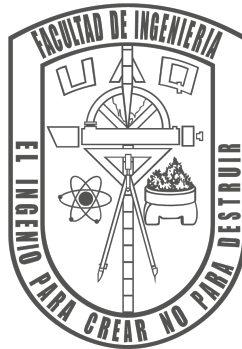
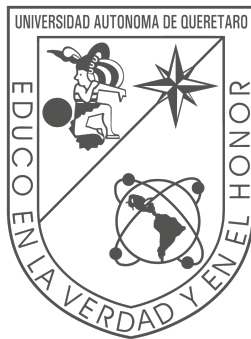


# Universidad Autónoma de Querétaro

Facultad de Ingeniería  
División de Investigación y Posgrado



## Práctica 1

# Manejo de Datos Tabulares y Visualización de Datos

Maestría en Ciencias en Inteligencia Artificial  
Optativa de especialidad IV - Machine Learning

Aldo Cervantes Marquez  
Expediente: 262775  
Profesor: Dr. Marco Antonio Aceves Fernández

Santiago de Querétaro, Querétaro, México  
Semestre 2023-1  
17 de Febrero de 2023

# Índice

<b>1. Objetivo</b>	<b>1</b>
<b>2. Introducción</b>	<b>1</b>
<b>3. Marco Teórico</b>	<b>1</b>
3.1. Instancias . . . . .	2
3.2. Atributos . . . . .	2
3.3. Tipos de datos . . . . .	2
3.4. Medidas Estadísticas . . . . .	2
<b>4. Materiales y Métodos</b>	<b>3</b>
4.1. Materiales . . . . .	3
4.1.1. Base de datos . . . . .	3
4.1.2. Librerías y entorno de desarrollo . . . . .	3
4.2. Metodología . . . . .	4
<b>5. Pseudocódigo</b>	<b>4</b>
<b>6. Resultados</b>	<b>5</b>
6.1. Librerías ggplot y altair . . . . .	8
<b>7. Conclusiones</b>	<b>9</b>
<b>Referencias</b>	<b>10</b>
<b>8. Código Documentado</b>	<b>11</b>

## 1. Objetivo

El principal objetivo de esta práctica es el de adquirir una base de datos y conocerla mediante su manejo de atributos e instancias, así como también la visualización de algunas características que la base de datos pueda proveer. Todo esto con la intención de poder observar en aplicaciones posteriores para poder filtrar y realizar un análisis más detallado de los datos que se adquieren mediante el lenguaje de programación Python.

La segunda parte de la práctica consiste en probar funciones mediante librerías especializadas en la visualización de datos, para poder contar con las herramientas necesarias en la realización y conocimiento de la práctica.

## 2. Introducción

La presente práctica consiste en dos partes, la primera es obtener una base de datos (*Dataset*) de algún proceso o sistema de interés. Para posteriormente analizarla y obtener algunos elementos importantes para conocer y poder visualizar la información de una manera más detallada y entendible.

Por lo que los principales componentes a obtener de la base de datos son:

- Número de atributos.
- Número de instancias.
- Mínimo.
- Máximo.
- Desviación estándar.
- Media/moda (según el caso).
- Datos atípicos.
- Datos faltantes.
- Tipo de distribución.

La segunda parte de la práctica consiste en realizar pruebas de dos librerías y hacer funciones de las mismas, las librerías son: **ggplot** y **Altair**.

## 3. Marco Teórico

El uso de las bases de datos tabulares son de alta importancia para el uso de algoritmos, y por lo tanto se dan a conocer algunos conceptos básicos que la comprenden, cabe destacar que todo lo de esta sección fue obtenido de [1].

### 3.1. Instancias

Se caracterizan por ser los valores de los atributos de la medición. Es decir, desde un punto de vista práctico, son los renglones (o registros) de la base de datos.

### 3.2. Atributos

Son los valores de las columnas y por lo tanto describen cada campo de la instancia en la que el atributo se refiere.

### 3.3. Tipos de datos

En las bases de datos principalmente existen los siguientes tipos de datos (véase Figura 1), los cuales son definidos formalmente como características en el sentido de que información posee el atributo.

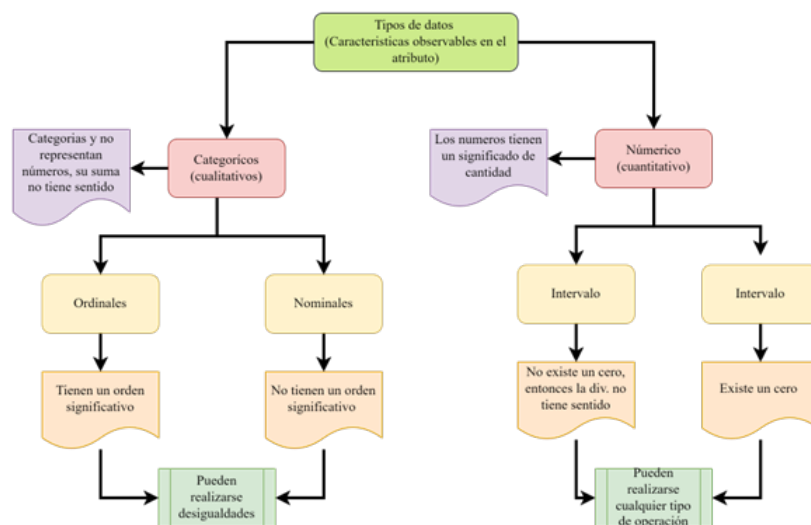


Figura 1: Tipos de características observables en un atributo.

Posteriormente, se tiene que los datos se pueden categorizar como continuos y discretos:

- Continuo: Tiene un número infinito de posibles valores. Son reales y/o de punto flotante
- Discreto: Tienen un número finito de posibles valores. Son enteros generalmente, como es el caso de los binarios.

### 3.4. Medidas Estadísticas

Para el desarrollo de la práctica será necesario conocer las formulas matemáticas de las medidas estadísticas para poder ser aplicados a la base de datos.

Media:

$$\mu = \frac{1}{n} \sum_{i=1}^n x_i \quad (1)$$

Moda:

$$\hat{m} = \text{valor que mas se repite} \quad (2)$$

Desviación estándar:

$$\sigma = \sqrt{\frac{\sum_{i=1}^n (x_i - \mu)^2}{n}} \quad (3)$$

## 4. Materiales y Métodos

### 4.1. Materiales

#### 4.1.1. Base de datos

La base de datos elegida fue obtenida de la página de [kaggle](#), la cual consta de datos de clientes de una tienda departamental durante los años de 2017-2020, enfocando la información a 4 pilares principales:

1. Perfil del cliente.
2. Preferencias de productos.
3. Éxito/fracaso de campañas publicitarias.
4. Frecuencia de consumos.

#### 4.1.2. Librerías y entorno de desarrollo

El análisis de los datos se llevará a cabo en el lenguaje de programación Python dentro del entorno de desarrollo de Jupyter Notebook.

Por parte de las librerías para visualización de datos se utilizarán matplotlib, seaborn, ggplot y altair.

Para instalar [ggplot](#) accediendo desde su página oficial se requiere de la siguiente [instrucción](#) en consola para obtener los paquetes de la librería (se instala la librería **plotnine** que contiene a ggplot, debido a que hay varias versiones y también es utilizada por el lenguaje R) [2]:

Mediante Pip (versión 3.4.0):

```
$ pip install plotnine
$ pip install 'plotnine[all]'
```

Mediante Anaconda (versión 3.4.0):

```
$ conda install -c conda-forge plotnine
```

Para instalar [Altair](#) accediendo desde su página oficial se instala mediante el comando en la terminal [3, 4]:

Mediante Pip (versión 4.2.2):

```
$ pip install altair vega_datasets
```

Mediante Anaconda (versión 4.2.2):

```
$ conda install -c conda-forge altair vega_datasets
```

Cabe destacar que se requieren los siguientes prerequisites (dependencies): Python 3.6 o mayor, [numpy](#), [entrypoints](#), [jsonschema](#), [Pandas](#) y [toolz](#).

## 4.2. Metodología

La metodología consiste básicamente en 2 partes: el manejo de los datos tabulares y aplicar la visualización con las librerías matplotlib, seaborn, ggplot y altair (véase Figura 2).

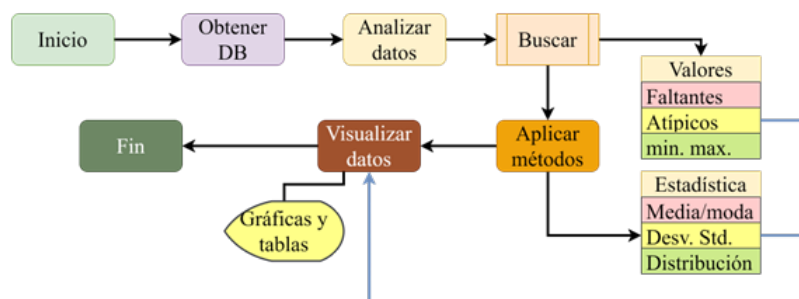


Figura 2: Metodología de la práctica.

## 5. Pseudocódigo

---

**Algoritmo 1** Pseudocódigo de manejo de datos tabulares y visualización.

---

```

Inicio
  DB → Obtener base de datos
  Filtrar(DB)
  Aplicar Análisis_datos:
    Buscar:
      Número (instancias, atributos)
      Faltantes, atípicos, mínimo, máximo
    Aplicar métodos_estadísticos:
      Media || moda, desviación estándar, Distribución

  Datos_resultantes → (Análisis_datos(DB), métodos_estadísticos(DB))

  Visualización datos(Datos_resultantes)
  Graficar (matplotlib, ggplot, altair, DB)
Fin
  
```

---

## 6. Resultados

La base datos fue analizada y se observó lo siguiente:

- Se cuenta originalmente con 39 atributos y 2205 instancias.

Sin embargo, fue posible cambiar algunos atributos de la tabla para reducir el total, mediante el uso de la categorización de las instancias.

Educación	Estado marital
0: sin educación	0: Anónim@
1: Básica	1: Divorciad@
2: Profesional	2: Casad@
3: Especialidad	3: Solter@
4: Maestría	4: Juntad@
5: Doctorado	5: Viud@
6: Desconocido	6: Desconocido

Posteriormente se eliminaron algunos atributos que no eran de utilidad como el ID del cliente. Para finalmente tener una versión con **28 atributos** y **2205 instancias**. En la Tabla 1 se muestra el análisis estadístico de la base de datos.

Tabla 1: Resultados de los atributos.

Categoría	Atributo	Etiqueta	Media	Desv.std.	[min,max]	Faltante	Atípicos	Dist.
Perfil del cliente	Ingreso	'Income'	51622	20713	[1730,113734]	0	0	Bimodal
	Hijos (niños)	'Kidhome'	0.44	0.53	[0,2]	0	0	Exp.
	Hijos (adolescentes)	'Teenhome'	0.5	0.54	[0,2]	0	0	Exp.
	Periodo de compra	'Recency'	49.009	49.9	[0,99]	0	0	Uniforme
Consumos del cliente	Venta dpto. vinos	'MntWines'	306	337	[0,1493]	0	34	Normal Izq
	Venta dpto. Frutas	'MntFruits'	26.4	39.8	[0,199]	0	245	Normal Izq
	Venta dpto. Carnes	'MntMeatProducts'	165.3	217	[0,1725]	0	170	Normal Izq
	Venta dpto. Pescado	'MntFishProducts'	37.75	54.82	[0,259]	0	222	Normal Izq
	Venta dpto. Dulces	'MntSweetProducts'	21.1	41.1	[0,262]	0	238	Normal Izq
	Venta dpto. Oro	'MntGoldProds'	44.05	51.73	[0,321]	0	204	Normal Izq
	Venta prod. reg.	'MntTotal'	562.76	575.93	[4,2491]	0	3	Normal Izq
	Venta total	'MntRegularProds'	606	601	[5,2525]	0	3	Normal Izq
Canales de venta	Venta en Oferta	'NumDealsPurchases'	2.31	1.88	[0,15]	0	82	Normal Izq
	Venta en Pag. Web	'NumWebPurchases'	4.1	2.73	[0,27]	0	3	Normal Izq
	Venta en Catálogo	'NumCatalogPurchases'	2.64	2.79	[0,28]	0	20	Normal Izq
	Venta en tienda	'NumStorePurchases'	5.82	3.24	[0,13]	0	0	Normal Izq
Marketing	Visitas a la Pagina web	'NumWebVisitsMonth'	5.33	2.41	[0,20]	0	8	Tiende Bimodal
P. cliente	Edad	'Age'	51.09	11.7	[24,80]	0	0	Tiende Bimodal
	Días como cliente	'Customer_Days'	2512.71	202.5	[2159,2858]	0	0	Uniforme
Marketing	Compró en Campaña 1	'AcceptedCmp1'	0	--	[0,1]	0	Binario	
	Compró en Campaña 2	'AcceptedCmp2'	0	--	[0,1]	0		
	Compró en Campaña 3	'AcceptedCmp3'	0	--	[0,1]	0		
	Compró en Campaña 4	'AcceptedCmp4'	0	--	[0,1]	0		
	Compró en Campaña 5	'AcceptedCmp5'	0	--	[0,1]	0		
	Compró en Campaña	'Response'	0	--	[0,1]	0		
	Tuvo queja	'Complain'	0	--	[0,1]	0		
P. cliente	Educación	'education'	2	--	[0,6]	0	Categorico	
	Estado civil	'marital_Status'	2	--	[0,6]	0		

A continuación se muestran las gráficas de mayor importancia, para observar todas, al final del documento se anexan.

Por parte de los ingresos (Información personal del cliente) se observa un comportamiento bimodal muy interesante y a pesar de no tener datos atípicos es posible observar como es que se remarcen clases económicas” (vease Figura 3).

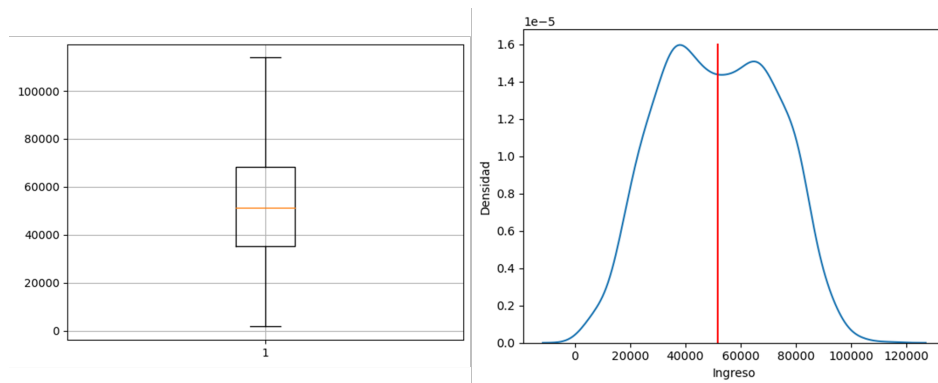


Figura 3: Boxplot y densidad de distribución de ingresos.

Por parte de las ventas por departamento (consumo del cliente) se observa que hay muchos datos atípicos en las ventas que realizan, significando que hay clientes que gastan mucho en comparación con los demás y por lo tanto se ve reflejado (véase Figura 4). Por lo que en la gráfica de densidad se nota como no afecta mucho esos datos.

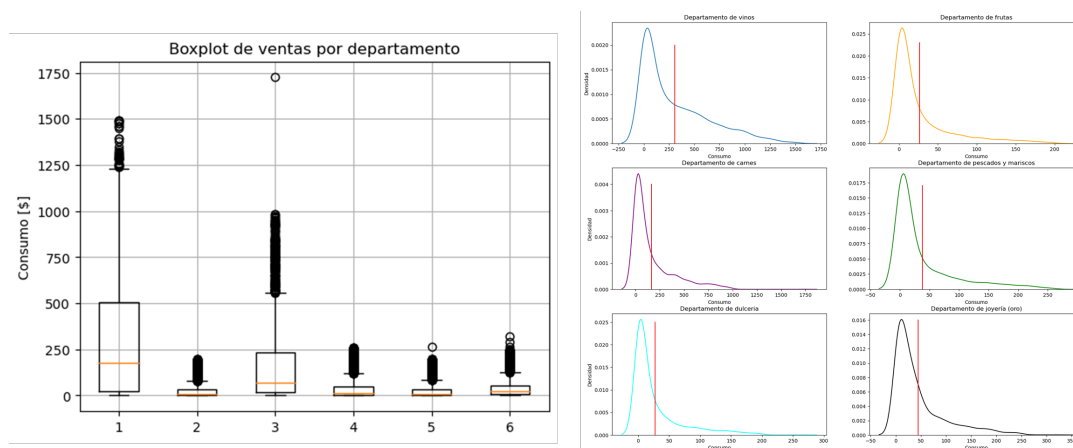


Figura 4: Boxplot y densidad de distribución en ventas por departamento.

donde:

1. Departamento de vinos.
2. Departamento de frutas.
3. Departamento de carnes.



4. Departamento de pescado y mariscos.
5. Departamento de dulcería.
6. Departamento de joyería (oro).

Por parte de las campañas publicitarias (marketing), se observó que no ha habido mucho éxito en las campañas pues ni un 10 % de los clientes compró a raíz de la campaña (véase Figura 6).

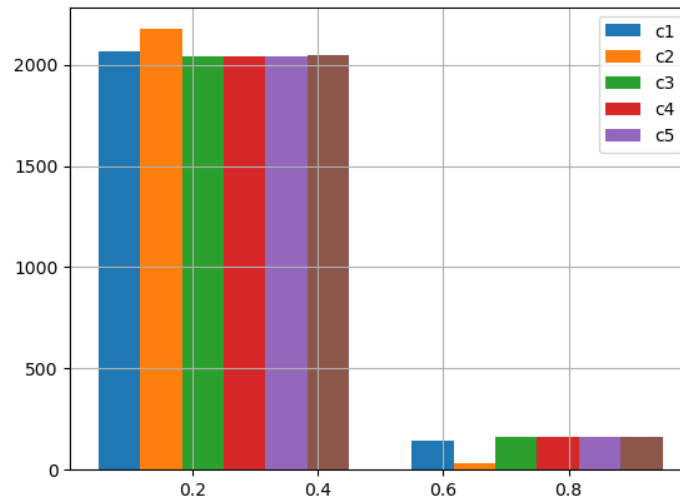


Figura 5: Histograma de ventas por campaña.

Finalmente en la categoría de canales de venta se observa que muy pocas personas compran muchos productos en general en la tienda, pues la distribución así lo muestra. Sin embargo aún así se vende más por tienda física.

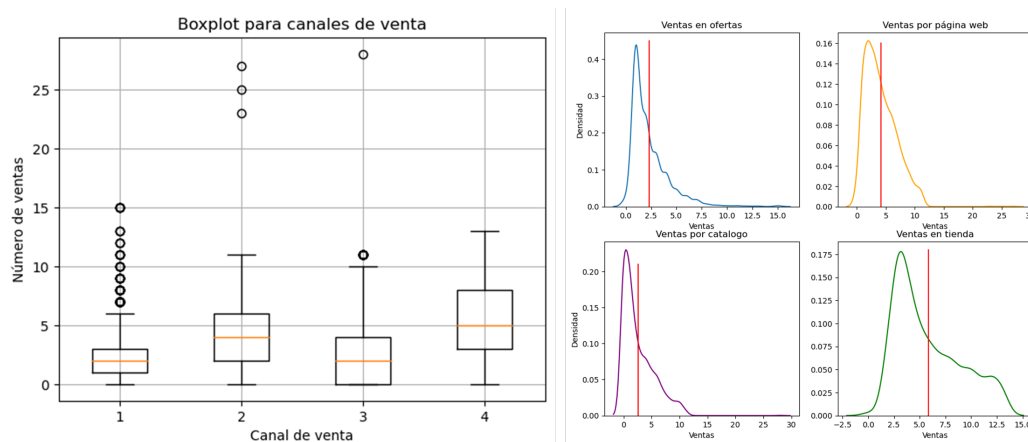
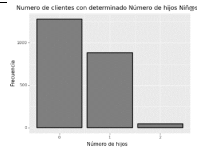
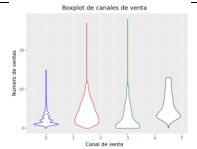
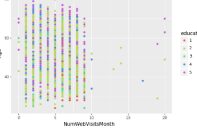
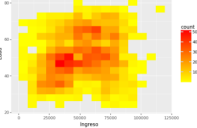
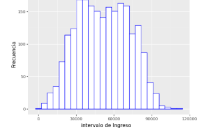
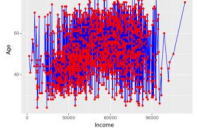
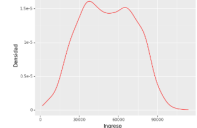
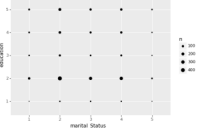
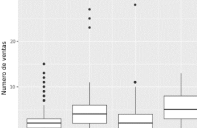
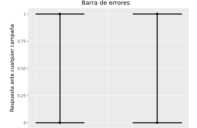


Figura 6: Boxplot y densidad de distribución por canales de venta.

## 6.1. Librerías ggplot y altair

A continuación, en la Tabla 2 se muestran algunas funciones de ggplot con sus respectivos comandos de funciones para poder ser utilizadas. La estructura de la sintaxis es mediante los signos + mediante una función `aes(dataframe)` la cual recopila los datos y permite concatenarlas con las funciones de cada gráfico y finalmente combinarlas para hacer gráficas diferentes a las conocidas, esta librería se destaca por su estética de los gráficos.

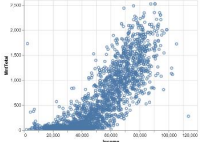
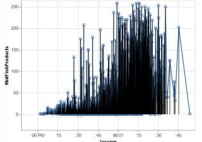
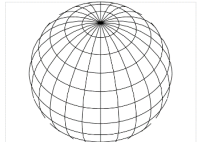
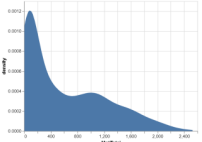
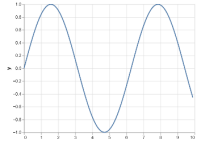
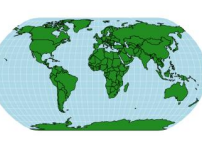
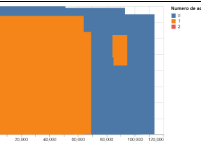
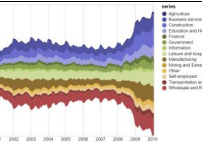
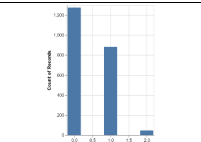
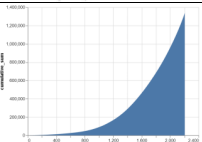
Tabla 2: Funciones de ggplot en python.

Ggplot					
Función (aes())	Breve descripción	Imagen	Función (aes())	Breve descripción	Imagen
<code>geom_bar()</code>	Gráfico de barras		<code>geom_violin()</code>	Gráfico de violín	
<code>geom_point()</code>	Diagrama de dispersión por clases		<code>geom_bin2d(bins=15)+ scale_fill_gradient()</code>	Mapa de calor	
<code>geom_hist()</code>	Histograma de los datos		<code>geom_line(color='blue')+ geom_point(color='red')</code>	Línea con punto	
<code>geom_density()</code>	Gráfico de distribución de densidad		<code>geom_count()+ scale_size_area(max_size=4)</code>	Gráfica de Puntos sobrepuestos	
<code>geom_boxplot()</code>	Boxplot		<code>geom_errorbar(x,ymin,ymax)</code>	Barras de error	

En la librería altair (véase Tabla 3 con las funciones), es posible generar gráficos de todo tipo, inclusive geográficos, lo cual le da una gran ventaja y competitividad con la librería geopandas, pero teniendo en cuenta que altair tiene más funciones específicas para estadística y visualización de los datos.

La sintaxis de esta librería se basa en el uso de objetos (uso de puntos para llamar a otras funciones), donde las que destacan son `encode()` siendo algo similar a `aes()` en ggplot. Sin embargo estas difieren en que altair tiene que llamar un `chart()` para poder crear la figura, algo que en ggplot no es necesario realizar.

Tabla 3: Funciones de altair en python.

Altair as alt					
Función (encode())	Breve descripción	Imagen	Función (encode())	Breve descripción	Imagen
<code>alt.Chart(db2).mark_point().encode(alt.X('Income'), alt.Y('MntTotal'))</code>	Diagrama de dispersión		<code>base.mark_line() + base.mark_point()</code>	Punto con línea	
<code>alt.Chart(data).mark_geoshape(stroke='black').project('orthographic', rotate=[0, -45, 0])</code>	Figuras 3d		<code>lt.Chart(db2).transform_density()</code>	Función de densidad	
<code>alt.Chart(data).transform_calculate(y='sin(datum.x)').mark_line().encode(x='x:Q', y='y:Q')</code>	Funciones continuas		<code>alt.layer(alt.Chart(sphere).mark_geoshape())</code>	Mapas	
<code>alt.Chart(db2).mark_rect().encode(alt.X('Income:Q', title='Income'), alt.Y('Age:Q', title='Age'), alt.Color('Teenhome:N', title='Numero de adolescentes'))</code>	Mapa de calor		<code>alt.Chart(source).mark_area().encode()</code>	Stream Graph Animadas	
<code>alt.Chart(db2).mark_bar(size=30).encode(x=('Kidhome:Q'), y='count()').properties(width=200)</code>	Barra de frecuencias		<code>alt.Chart(db2).transform_window()</code>	Gráfico Acumulativo	

## 7. Conclusiones

El manejo de datos tabulares son de gran importancia dentro del procesamiento y análisis de datos, en especial cuando estos datos serán utilizados para modelos basados en inteligencia artificial. Además de que la aplicación de las medidas estadísticas permiten tener un panorama general de los datos que se tienen, logrando poder observar si hay datos atípicos o aberrantes. También observar de una manera muy intuitiva el comportamiento de los datos y de manera gráfica. Como por ejemplo se observa que las ventas por oferta o rebaja son bajas ( $\mu = 2.31$ ) en comparación con las ventas de catalogo ( $\mu = 2.64$ ) y las campañas también son poco exitosas (5.98 %), lo que describe un mal manejo de las campañas en las rebajas. También observar que hay una distribución con tendencia a uniforme en los días que tienen los clientes afiliados a la tienda, lo que podría ayudar a que en un futuro se pueda enfocar de una mejor manera las campañas de marketing que se tienen actualmente. Por parte de las librerías es posible observar que se tiene mucha variedad de funciones que pueden ayudar a visualizar diferentes datos que son de interés de una manera estética y con una mejor presentación que la que se podría tener con una librería estándar. Esto será una herramienta que

servirá para futuras prácticas en caso de ocupar otras funciones más complejas de visualización.

Finalmente se puede concluir que esta práctica fue de mucha utilidad en el manejo de los datos, su pre-tratamiento y su visualización, lo que facilitó el entendimiento de los atributos en una base de datos relativamente grande.

## Referencias

- [1] M. Fernández, *Inteligencia Artificial para Programadores con Prisa*. Amazon Digital Services LLC - KDP Print US, 2021.
- [2] “Function reference • ggplot2.” <https://ggplot2.tidyverse.org/reference/#plot-basics>. (Accessed on 02/16/2023).
- [3] “Vega-altair: Declarative visualization in python — altair 4.2.2 documentation.” <https://altair-viz.github.io/>. (Accessed on 02/16/2023).
- [4] “Data visualization in python like in r’s ggplot2 — by dr. gregor scheithauer — towards data science.” <https://towardsdatascience.com/data-visualization-in-python-like-in-rs-ggplot2-bc62f8debbf5>. (Accessed on 02/16/2023).

# Práctica 1 manejo de datos tabulares

February 17, 2023

## 1 Práctica 1: Manejo de datos tabulares (parte 1)

### 1.1 Machine Learning

*Aldo Cervantes Marquez*

Esta práctica consiste en realizar el análisis de una base de datos de interés. Se deben realizar cierto procesamiento a los datos para poder visualizarlos y comprenderlos de mejor manera, por lo que se deberá obtener lo siguiente:

- Número de atributos.
- Número de instancias.
- mínimo, máximo, media.
- Desviación estándar.
- Datos atípicos.
- Datos Faltantes.
- Tipo de distribución.

```
[105]: import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

database=pd.read_csv("C:\\Users\\aldoa\\Machine Learning\\ifood_df.csv")
print(database.shape)
```

(2205, 39)

‘ifood\_df.csv’ es una base de datos obtenida de [Kaggle](#) que consiste en la información de 2205 clientes de una compañía XYZ con datos basados en:

1. Perfil del cliente.
2. Preferencias de productos.
3. Éxito/fracaso de campañas publicitarias.
4. Frecuencia de ventas.

Incluyendo el siguiente diccionario de nomenclaturas

entonces en el total de dinero gastado en los productos se tiene que:

$$MntWines + MntFruits + MntMeatProducts + MntFishProducts + MntSweetProducts = MntRegularProds$$

$$MntWines + MntFruits + MntMeatProducts + MntFishProducts + MntSweetProducts + MntGoldProds = Mn$$

### 1.1.1 Numero de atributos e instancias.

Al llamar la clase DataFrame podemos observar la estructura, por lo que contiene **2205 instancias** y **39 atributos** de manera cruda

## 1.2 Categorización de los datos

Sin embargo, es posible reducir el número de atributos mediante el uso de variables categoricas para el máximo nivel de estudios categorizando de la siguiente manera:

- Si no tiene educación = 0
- Si teiene basica = 1
- Si tiene estudios profesionales = 2
- Si tiene especialidad (2n\_cycle)= 3
- Si tiene maestría = 4
- Si tiene doctorado = 5
- Si es desconocido (dato faltante)= 6

De igual manera su estado civil:

- Si es anonima = 0
- Si es divorciad@ = 1
- Si es casad@ = 2
- Si es solter@ = 3
- Si esta juntad@ = 4
- Si es viud@ =5
- Si se desconoce (dato faltante) = 6

```
[106]: education=database[['education_Basic','education_Graduation','education_2n_
    ↳Cycle','education_Master','education_PhD']]

#print(education.value_counts())
education2=pd.DataFrame({'education'})
for a in range(len(education)):
    if education.iloc[a,0]==1: # Educación Básica
        education2.loc[a]=1
    elif education.iloc[a,1]==1: # Educacion profesional
        education2.loc[a]=2
    elif education.iloc[a,2]==1: # Especialidad
        education2.loc[a]=3
    elif education.iloc[a,3]==1: # Educacion maestría
        education2.loc[a]=4
    elif education.iloc[a,4]==1: # Educacion Doctorado
        education2.loc[a]=5
    elif education.iloc[a,:].isna(): # Si no se conoce
        education2.loc[a]=6
```

```

else:
    education2.loc[a]=0 # Si no tiene educación

#print(education2.value_counts())

```

[107]: marriage=database[['marital\_Divorced','marital\_Married','marital\_Single','marital\_Together','ma  
marriage2=pd.DataFrame({'marital\_Status'})

```

#print(marriage.value_counts())
for a in range(len(education)):
    if marriage.iloc[a,0]==1: # Divorciad@
        marriage2.loc[a]=1
    elif marriage.iloc[a,1]==1: # Casad@
        marriage2.loc[a]=2
    elif marriage.iloc[a,2]==1: # Solter@
        marriage2.loc[a]=3
    elif marriage.iloc[a,3]==1: # Juntad@
        marriage2.loc[a]=4
    elif marriage.iloc[a,4]==1: # Viud@
        marriage2.loc[a]=5
    elif marriage.iloc[a,:].isna(): # Si no se conoce
        marriage2.loc[a]=6
    else:
        marriage2.loc[a]=0 # Si no respondio

#print(marriage2.value_counts())

```

[108]: # Igualar las bases de datos para obtener otra y agregar las categorizaciones  
db2=database  
db2=db2.  
↳drop(['AcceptedCmpOverall','marital\_Divorced','marital\_Married','marital\_Single','marital\_Tog  
↳Cycle','education\_Master','education\_PhD'],axis=1)

```

#frames=[db2,marriage2,education2]
#db2=pd.concat(frames)
db2['marital_Status']=marriage2
db2['education']=education2
db2['MntTotal']=db2['MntWines']+ db2['MntFruits']+db2['MntMeatProducts']+
↳db2['MntFishProducts']+ db2['MntSweetProducts']+db2['MntGoldProds']
db2['MntRegularProds']=db2['MntWines']+ db2['MntFruits']+db2['MntMeatProducts']+
↳db2['MntFishProducts']+ db2['MntSweetProducts']
print('Los atributos son: ',db2.columns)
## Calcular cantidad de atributos e instancias
instancias,atributos=db2.shape
print('La cantidad de atributos es:',atributos)

print('La cantidad de instancias es:',instancias)

```

```
Los atributos son: Index(['Income', 'Kidhome', 'Teenhome', 'Recency',
'MntWines', 'MntFruits',
'MntMeatProducts', 'MntFishProducts', 'MntSweetProducts',
'MntGoldProds', 'NumDealsPurchases', 'NumWebPurchases',
'NumCatalogPurchases', 'NumStorePurchases', 'NumWebVisitsMonth',
'AcceptedCmp3', 'AcceptedCmp4', 'AcceptedCmp5', 'AcceptedCmp1',
'AcceptedCmp2', 'Complain', 'Response', 'Age', 'Customer_Days',
'MntTotal', 'MntRegularProds', 'marital_Status', 'education'],
dtype='object')
```

La cantidad de atributos es: 28

La cantidad de instancias es: 2205

☒ Atributos = 28

☒ Instancias = 2205

```
[109]: db2[['Income', 'Kidhome', 'Teenhome', 'Recency', 'MntWines', 'MntFruits',
'MntMeatProducts', 'MntFishProducts', 'MntSweetProducts',
'MntGoldProds', 'NumDealsPurchases', 'NumWebPurchases',
'NumCatalogPurchases', 'NumStorePurchases', 'NumWebVisitsMonth', 'Age',
'Customer_Days',
'MntTotal', 'MntRegularProds']].describe()
```

```
[109]:
```

	Income	Kidhome	Teenhome	Recency	MntWines \
count	2205.000000	2205.000000	2205.000000	2205.000000	2205.000000
mean	51622.094785	0.442177	0.506576	49.009070	306.164626
std	20713.063826	0.537132	0.544380	28.932111	337.493839
min	1730.000000	0.000000	0.000000	0.000000	0.000000
25%	35196.000000	0.000000	0.000000	24.000000	24.000000
50%	51287.000000	0.000000	0.000000	49.000000	178.000000
75%	68281.000000	1.000000	1.000000	74.000000	507.000000
max	113734.000000	2.000000	2.000000	99.000000	1493.000000

	MntFruits	MntMeatProducts	MntFishProducts	MntSweetProducts \
count	2205.000000	2205.000000	2205.000000	2205.000000
mean	26.403175	165.312018	37.756463	27.128345
std	39.784484	217.784507	54.824635	41.130468
min	0.000000	0.000000	0.000000	0.000000
25%	2.000000	16.000000	3.000000	1.000000
50%	8.000000	68.000000	12.000000	8.000000
75%	33.000000	232.000000	50.000000	34.000000
max	199.000000	1725.000000	259.000000	262.000000

	MntGoldProds	NumDealsPurchases	NumWebPurchases	NumCatalogPurchases \
count	2205.000000	2205.000000	2205.000000	2205.000000
mean	44.057143	2.318367	4.100680	2.645351
std	51.736211	1.886107	2.737424	2.798647
min	0.000000	0.000000	0.000000	0.000000
25%	9.000000	1.000000	2.000000	0.000000



50%	25.000000	2.000000	4.000000	2.000000
75%	56.000000	3.000000	6.000000	4.000000
max	321.000000	15.000000	27.000000	28.000000

	NumStorePurchases	NumWebVisitsMonth	Age	Customer_Days \
count	2205.000000	2205.000000	2205.000000	2205.000000
mean	5.823583	5.336961	51.095692	2512.718367
std	3.241796	2.413535	11.705801	202.563647
min	0.000000	0.000000	24.000000	2159.000000
25%	3.000000	3.000000	43.000000	2339.000000
50%	5.000000	6.000000	50.000000	2515.000000
75%	8.000000	7.000000	61.000000	2688.000000
max	13.000000	20.000000	80.000000	2858.000000

	MntTotal	MntRegularProds
count	2205.000000	2205.000000
mean	606.821769	562.764626
std	601.675284	575.936911
min	5.000000	4.000000
25%	69.000000	56.000000
50%	397.000000	343.000000
75%	1047.000000	964.000000
max	2525.000000	2491.000000

```
[110]: ## Moda
print("** Campañas y respuesta **")
print(db2[['AcceptedCmp1']].mode())
print(db2[['AcceptedCmp2']].mode())
print(db2[['AcceptedCmp3']].mode())
print(db2[['AcceptedCmp4']].mode())
print(db2[['AcceptedCmp5']].mode())
print(db2[['Response']].mode())
print("** Quejas **")
print(db2[['Complain']].mode())
print("** Educación **")
print(db2[['education']].mode())
print("** Estado civil **")
print(db2[['marital_Status']].mode())
```

```
** Campañas y respuesta **
AcceptedCmp1
0          0
AcceptedCmp2
0          0
AcceptedCmp3
0          0
AcceptedCmp4
0          0
```

```

    AcceptedCmp5
0          0
    Response
0          0
** Quejas **
    Complain
0          0
** Educación **
    education
0          2
** Estado civil **
    marital_Status
0          2

```

```

[162]: lab=['AcceptedCmp1','AcceptedCmp2','AcceptedCmp3','AcceptedCmp4','AcceptedCmp5']
temp=0
for xx in range(len(lab)):
    aa=db2[lab[xx]].value_counts()
    temp=temp+aa[1]/(2205)

print('El porcentaje de exito de las campañas es de:',(temp/(len(lab))*100),'%')

```

El porcentaje de exito de las campañas es de: 5.986394557823129 %

A continuación se muestran las métricas estadísticas principales.

- ☒ Media
- ☒ Desviación estándar
- ☒ Máximo
- ☒ Mínimo

```

[111]: db2.isna().sum()

```

```

[111]: Income          0
Kidhome              0
Teenhome            0
Recency             0
MntWines            0
MntFruits           0
MntMeatProducts     0
MntFishProducts     0
MntSweetProducts    0
MntGoldProds        0
NumDealsPurchases   0
NumWebPurchases     0
NumCatalogPurchases 0
NumStorePurchases   0
NumWebVisitsMonth   0
AcceptedCmp3        0

```

```

AcceptedCmp4      0
AcceptedCmp5      0
AcceptedCmp1      0
AcceptedCmp2      0
Complain          0
Response          0
Age              0
Customer_Days     0
MntTotal          0
MntRegularProds   0
marital_Status    0
education         0
dtype: int64

```

Como se observa no hay datos faltantes - [x] **Datos faltantes = 0**

### 1.3 Datos atípicos

```

[112]: # Obtención de boxplot para gastos por departamento
bp1=plt.boxplot(db2[['MntWines', 'MntFruits',
                    'MntMeatProducts', 'MntFishProducts', 'MntSweetProducts',
                    'MntGoldProds']])
plt.title('Boxplot de ventas por departamento')
plt.ylabel('Consumo [$]')
plt.grid()

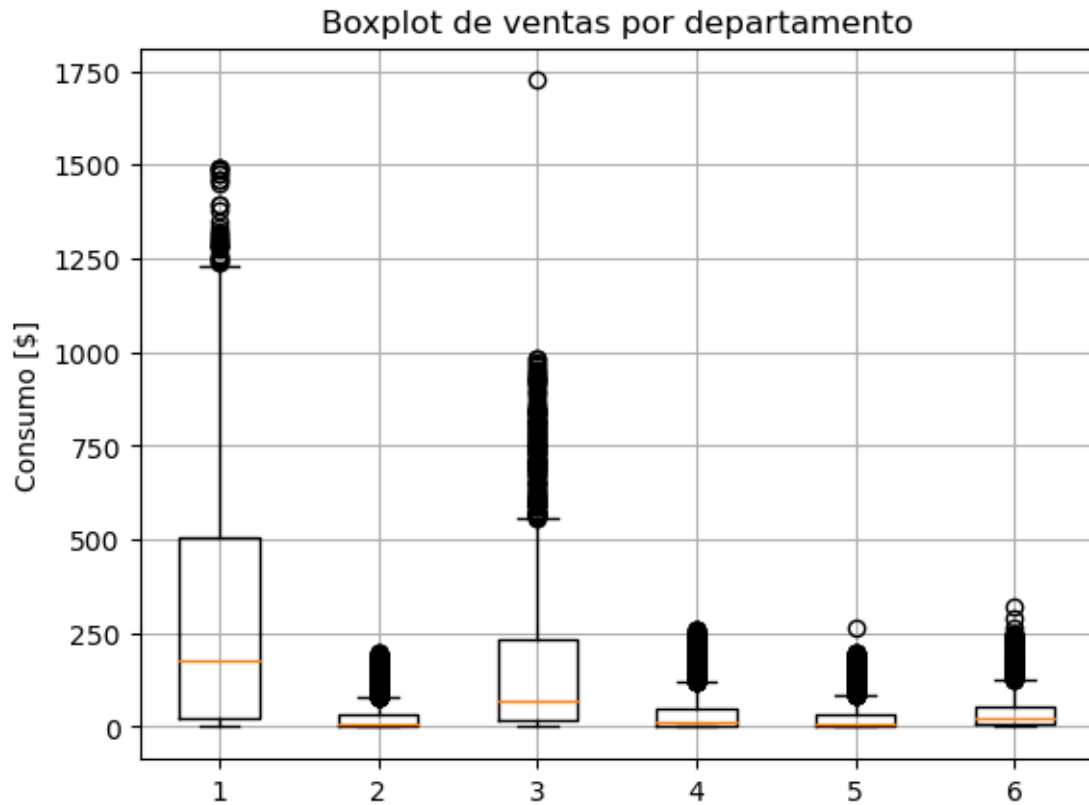
labels=['MntWines', 'MntFruits',
        'MntMeatProducts', 'MntFishProducts', 'MntSweetProducts',
        'MntGoldProds']
outliers1=[item.get_ydata() for item in bp1['fliers']]
for lon in range(len(outliers1)):
    print('Cantidad de valores atípicos de ',labels[lon],':',outliers1[lon].
    ↪shape)

```

```

Cantidad de valores atípicos de MntWines : (34,)
Cantidad de valores atípicos de MntFruits : (245,)
Cantidad de valores atípicos de MntMeatProducts : (170,)
Cantidad de valores atípicos de MntFishProducts : (222,)
Cantidad de valores atípicos de MntSweetProducts : (238,)
Cantidad de valores atípicos de MntGoldProds : (204,)

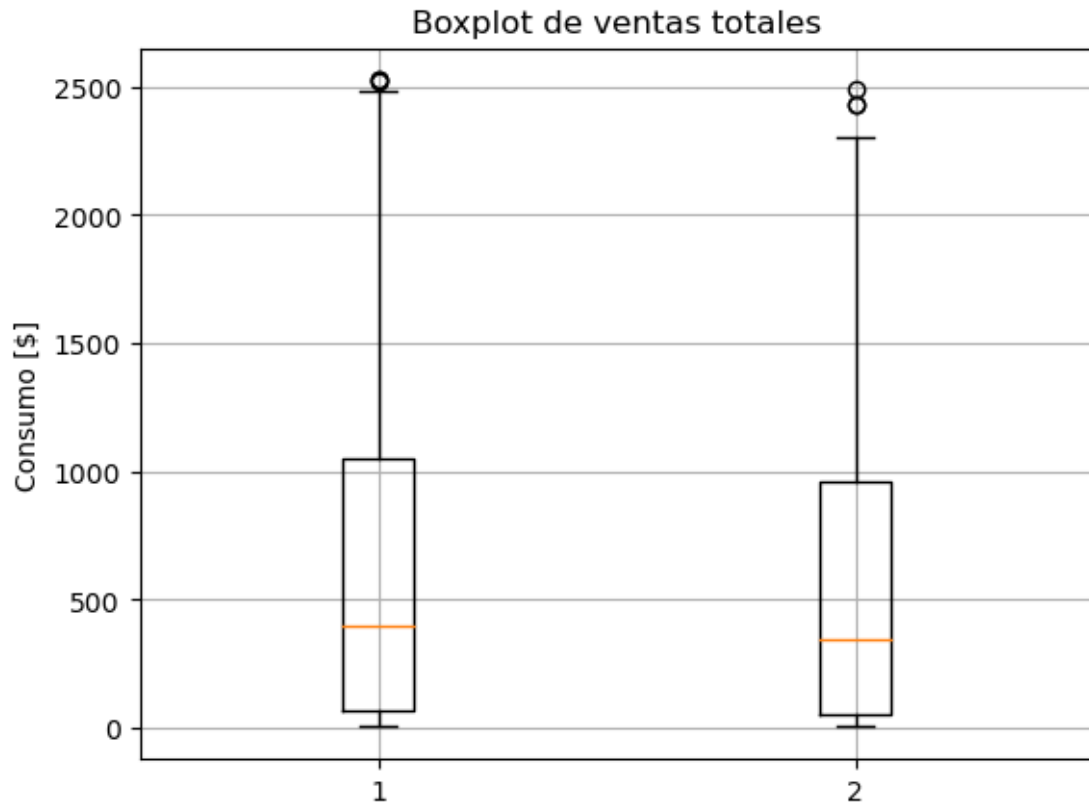
```



```
[113]: # Datos atípicos en totales
bp1=plt.boxplot(db2[['MntTotal', 'MntRegularProds']])
plt.title('Boxplot de ventas totales')
plt.ylabel('Consumo [$]')
plt.grid()

labels=['MntTotal', 'MntRegularProds']
outliers1=[item.get_ydata() for item in bp1['fliers']]
for lon in range(len(outliers1)):
    print('Cantidad de valores atípicos de ',labels[lon],':',outliers1[lon].
    ↳shape)
```

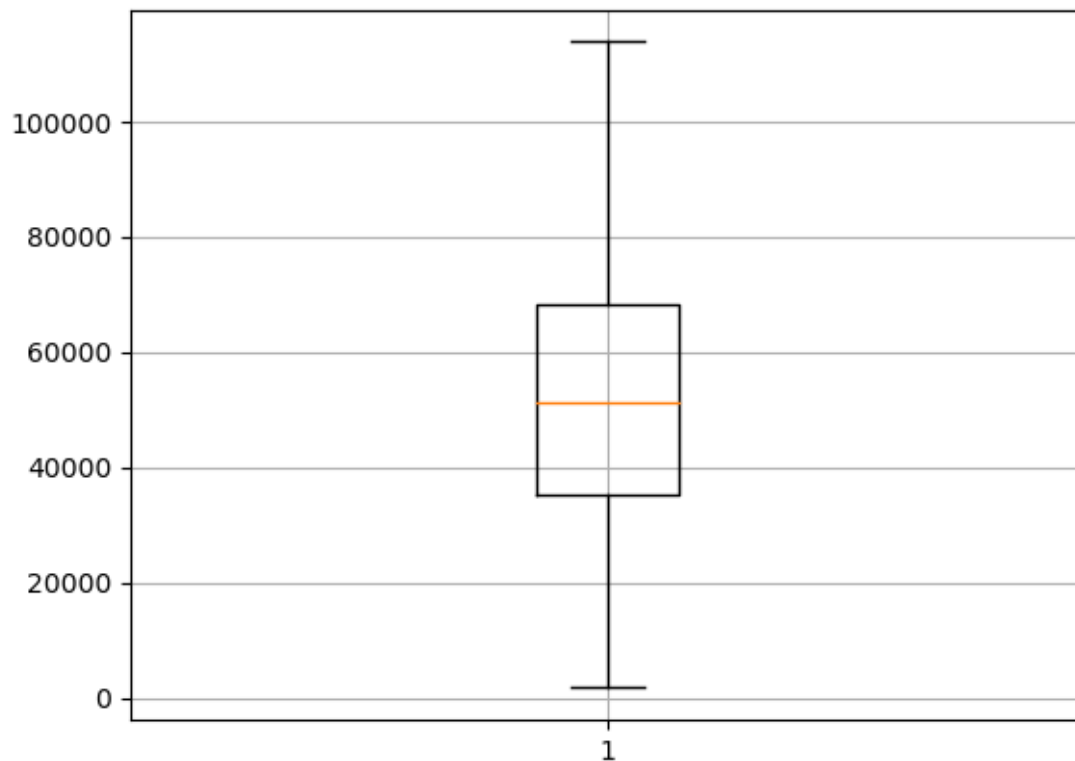
```
Cantidad de valores atípicos de MntTotal : (3,)
Cantidad de valores atípicos de MntRegularProds : (3,)
```



```
[114]: # Datos de boxplot para ingresos de la familia
bp1=plt.boxplot(db2['Income'])
plt.grid()

outliers1=[item.get_ydata() for item in bp1['fliers']]
for lon in range(len(outliers1)):
    print('Cantidad de visitas a la pagina web por mes',outliers1[lon].shape)
```

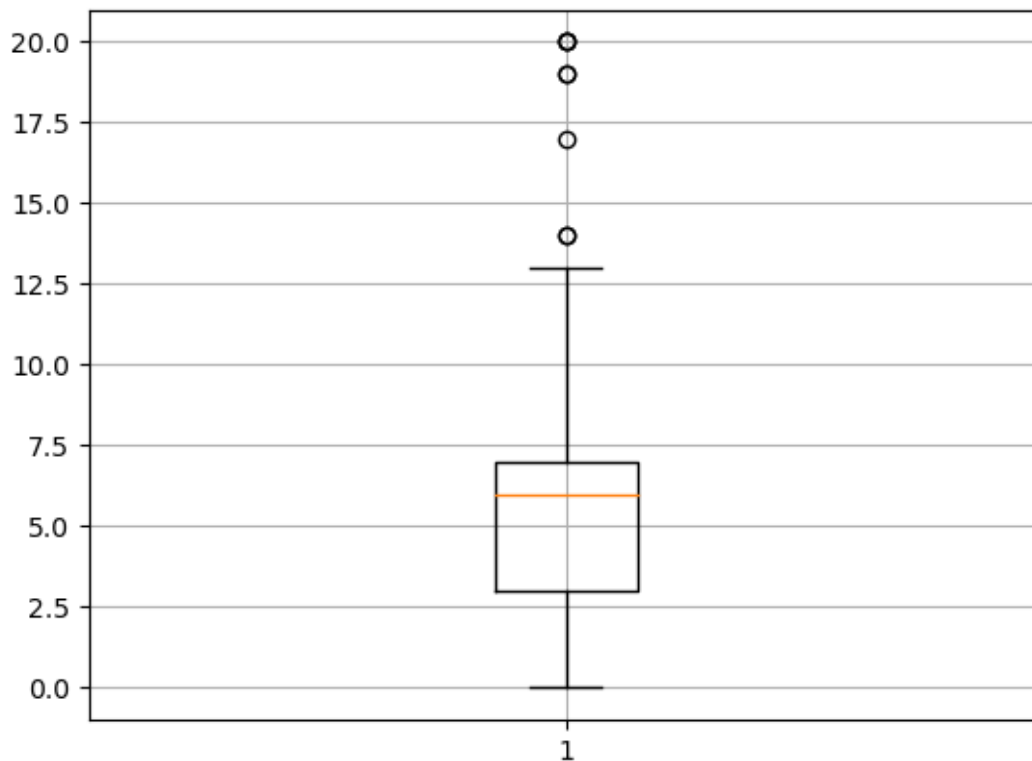
Cantidad de visitas a la pagina web por mes (0,)



```
[115]: bp1=plt.boxplot(db2['NumWebVisitsMonth'])
plt.grid()

outliers1=[item.get_ydata() for item in bp1['fliers']]
for lon in range(len(outliers1)):
    print('Cantidad de visitas a la pagina web por mes',outliers1[lon].shape)
```

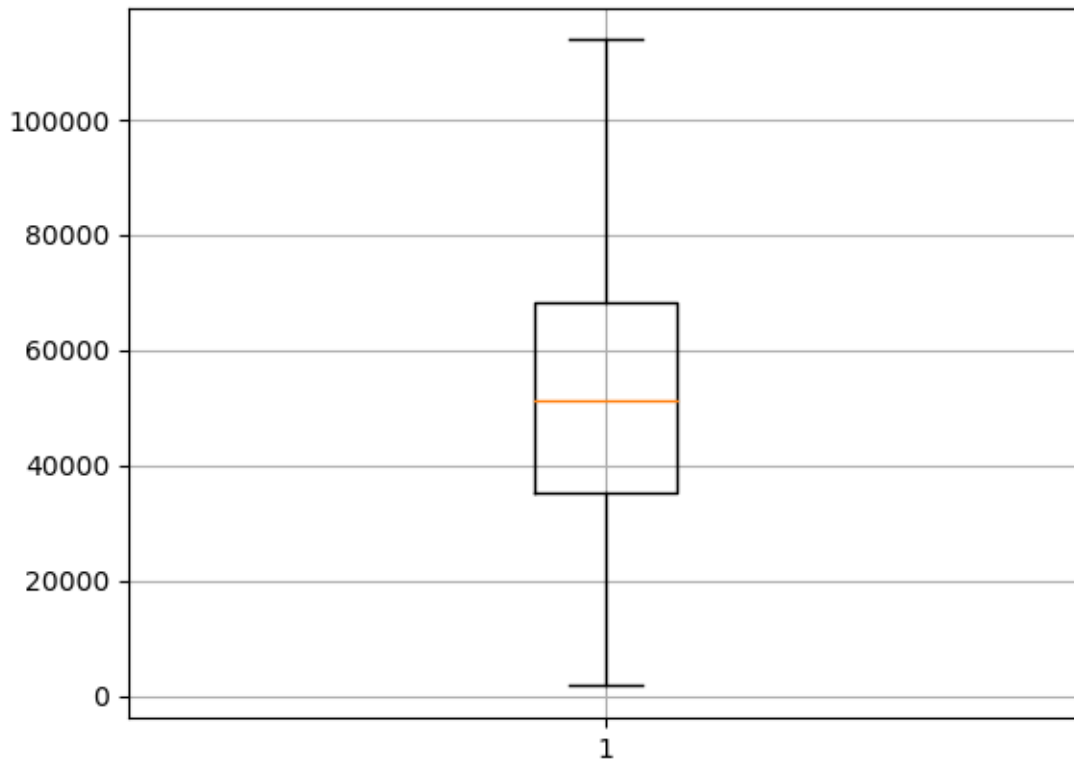
Cantidad de visitas a la pagina web por mes (8,)



```
[116]: # Boxplot para visitas web
bp1=plt.boxplot(db2['Income'])
plt.grid()

outliers1=[item.get_ydata() for item in bp1['fliers']]
for lon in range(len(outliers1)):
    print('Cantidad de datos atipicos en ingresos',outliers1[lon].shape)
```

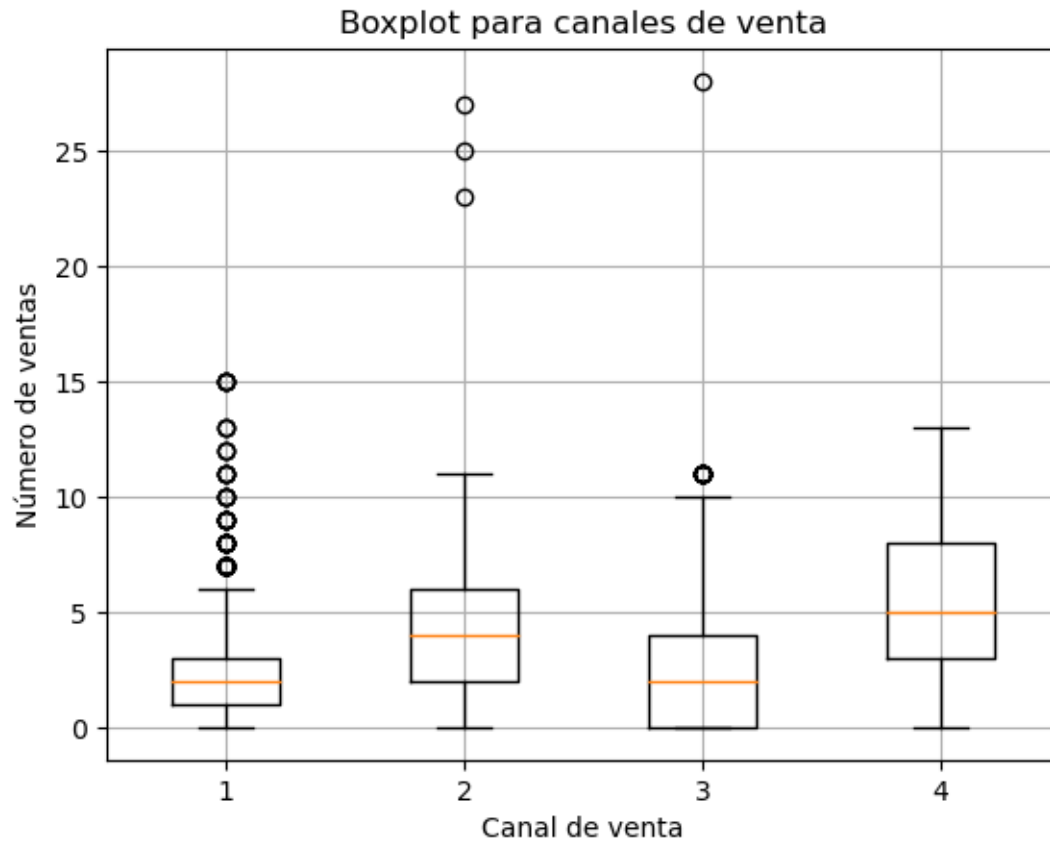
Cantidad de datos atipicos en ingresos (0,)



```
[117]: # Datos de boxplot para canales de venta
bp1=plt.boxplot(db2[['NumDealsPurchases',
    ↳ 'NumWebPurchases', 'NumCatalogPurchases', 'NumStorePurchases']])
plt.title('Boxplot para canales de venta')
plt.ylabel('Número de ventas')
plt.xlabel('Canal de venta')
plt.grid()
labels=['NumDealsPurchases', 'NumWebPurchases', 'NumCatalogPurchases',
    ↳ 'NumStorePurchases']
outliers1=[item.get_ydata() for item in bp1['fliers']]
for lon in range(len(outliers1)):
    print('Cantidad de valores atípicos de ', labels[lon], ': ', outliers1[lon].
    ↳ shape)
```

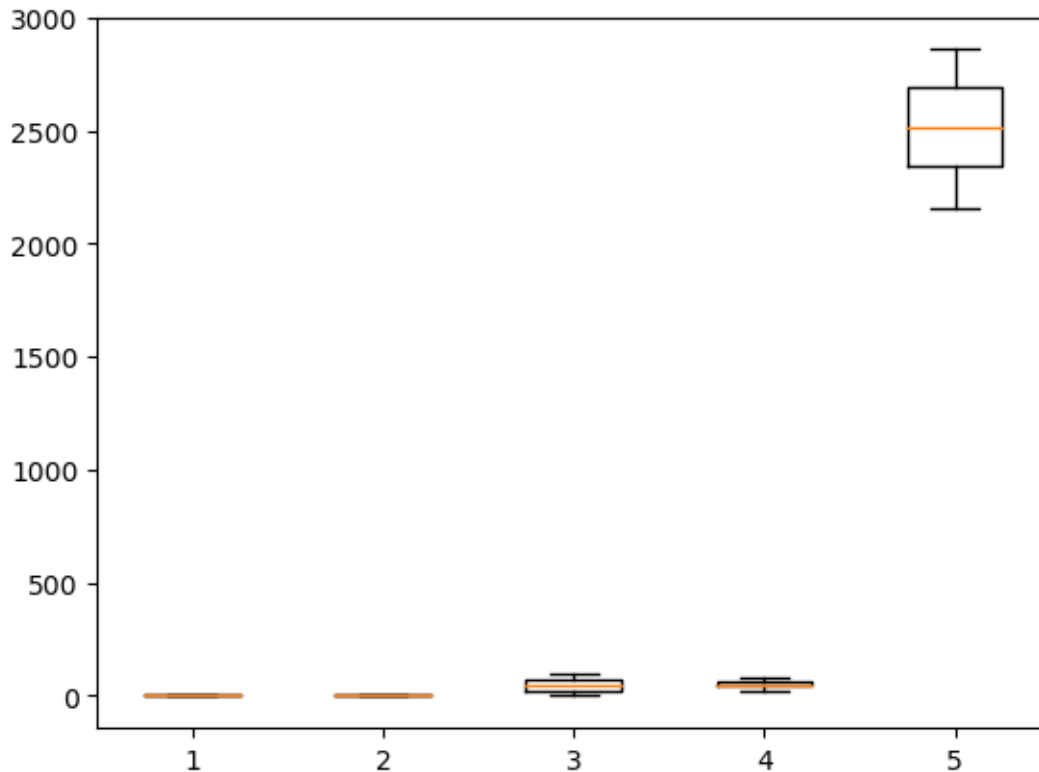
```
Cantidad de valores atípicos de NumDealsPurchases : (82,)
Cantidad de valores atípicos de NumWebPurchases : (3,)
Cantidad de valores atípicos de NumCatalogPurchases : (20,)
Cantidad de valores atípicos de NumStorePurchases : (0,)
```





```
[118]: # Datos atipicos por Perfil del cliente
aa=['Kidhome','Teenhome','Recency', 'Age', 'Customer_Days']
bp1=plt.boxplot(db2[aa])
outliers1=[item.get_ydata() for item in bp1['fliers']]
for lon in range(len(outliers1)):
    print('Cantidad de valores atípicos de ',aa[lon],':',outliers1[lon].shape)
```

```
Cantidad de valores atípicos de Kidhome : (0,)
Cantidad de valores atípicos de Teenhome : (0,)
Cantidad de valores atípicos de Recency : (0,)
Cantidad de valores atípicos de Age : (0,)
Cantidad de valores atípicos de Customer_Days : (0,)
```



## 1.4 Tipo de distribución

```
[119]: # Para valores ventas por departamento
# Density Transform
import seaborn as sns
fig, axes = plt.subplots(3, 2, figsize=(17, 15))
fig.suptitle('Distribución de ventas por departamento')
axes[0, 0].set_title('Departamento de vinos')
axes[0, 1].set_title('Departamento de frutas')
axes[1, 0].set_title('Departamento de carnes')
axes[1, 1].set_title('Departamento de pescados y mariscos')
axes[2, 0].set_title('Departamento de dulcería')
axes[2, 1].set_title('Departamento de joyería (oro)')

a1 = ['MntWines', 'MntFruits',
      'MntMeatProducts', 'MntFishProducts', 'MntSweetProducts',
      'MntGoldProds']

sns.kdeplot(ax=axes[0, 0], data=db2['MntWines'])
axes[0, 0].plot([db2[a1[0]].mean(), db2[a1[0]].mean()], [0, 0.0020], color='red')
```

```
axes[0,0].set_xlabel('Consumo')
axes[0,0].set_ylabel('Densidad')
#axes[0,0].set_xlim([0,db2[a1[0]].max()])

sns.kdeplot(ax=axes[0,1],data=db2['MntFruits'],color='orange')
axes[0,1].plot([db2[a1[1]].mean(),db2[a1[1]].mean()], [0,0.023],color='red')
axes[0,1].set_xlabel('Consumo')
axes[0,1].set_ylabel('')

sns.kdeplot(ax=axes[1,0],data=db2['MntMeatProducts'],color='purple')
axes[1,0].plot([db2[a1[2]].mean(),db2[a1[2]].mean()], [0,0.0040],color='red')
axes[1,0].set_xlabel('Consumo')
axes[1,0].set_ylabel('Densidad')

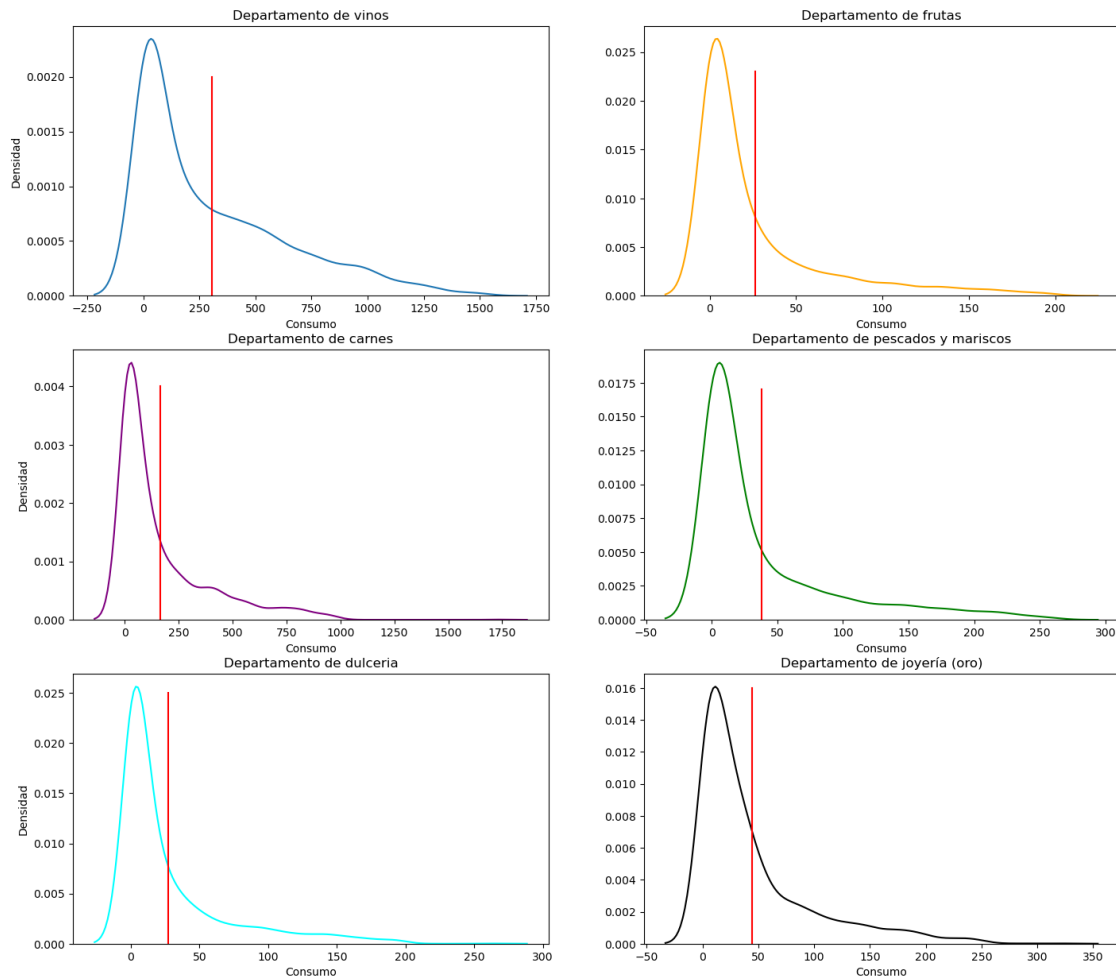
sns.kdeplot(ax=axes[1,1],data=db2['MntFishProducts'],color='green')
axes[1,1].plot([db2[a1[3]].mean(),db2[a1[3]].mean()], [0,0.0170],color='red')
axes[1,1].set_xlabel('Consumo')
axes[1,1].set_ylabel('')

sns.kdeplot(ax=axes[2,0],data=db2['MntSweetProducts'],color='cyan')
axes[2,0].plot([db2[a1[4]].mean(),db2[a1[4]].mean()], [0,0.0250],color='red')
axes[2,0].set_xlabel('Consumo')
axes[2,0].set_ylabel('Densidad')

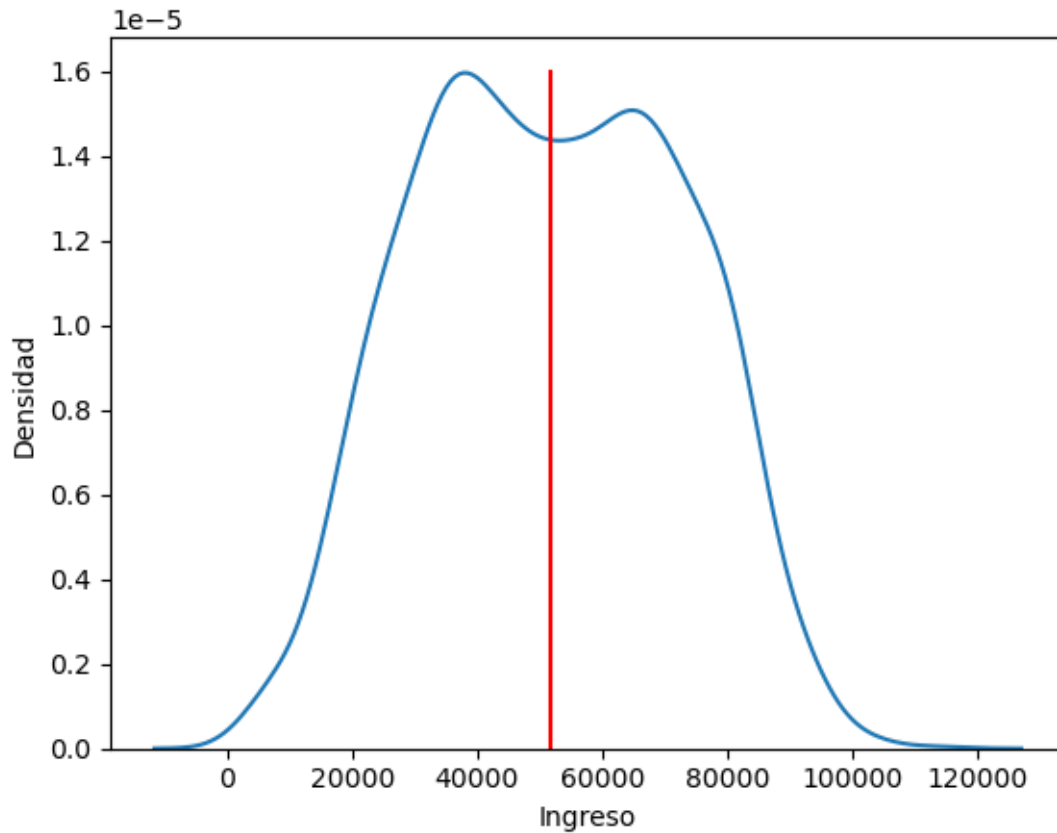
sns.kdeplot(ax=axes[2,1],data=db2['MntGoldProds'],color='black')
axes[2,1].plot([db2[a1[5]].mean(),db2[a1[5]].mean()], [0,0.0160],color='red')
axes[2,1].set_xlabel('Consumo')
axes[2,1].set_ylabel('')
#sns.kdeplot(ax=axes[0,1],db2['MntWines'])
```

[119]: Text(0, 0.5, '')

Distribución de ventas por departamento



```
[120]: # Distribución de ingresos
#fig,axes=plt.subplots(1,1)
#fig.suptitle('Distribución de Ingresos')
ax= sns.kdeplot(db2['Income'])
#axes[0].set_xlabel('Ingreso')
ax.set(xlabel='Ingreso',ylabel='Densidad')
ax.plot([db2['Income'].mean(),db2['Income'].mean()], [0,0.0000160],color='red')
plt.show()
```



```
[121]: # Distribución de los canales de venta
a2=['NumDealsPurchases', 'NumWebPurchases', 'NumCatalogPurchases',
    ↪ 'NumStorePurchases']

fig, axes = plt.subplots(2, 2, figsize=(10, 9))
fig.suptitle('Ventas por canales de venta')
axes[0, 0].set_title('Ventas en ofertas')
axes[0, 1].set_title('Ventas por página web')
axes[1, 0].set_title('Ventas por catalogo')
axes[1, 1].set_title('Ventas en tienda')

sns.kdeplot(ax=axes[0, 0], data=db2[a2[0]])
axes[0, 0].plot([db2[a2[0]].mean(), db2[a2[0]].mean()], [0, 0.45], color='red')
axes[0, 0].set_xlabel('Ventas')
axes[0, 0].set_ylabel('Densidad')

sns.kdeplot(ax=axes[0, 1], data=db2[a2[1]], color='orange')
```

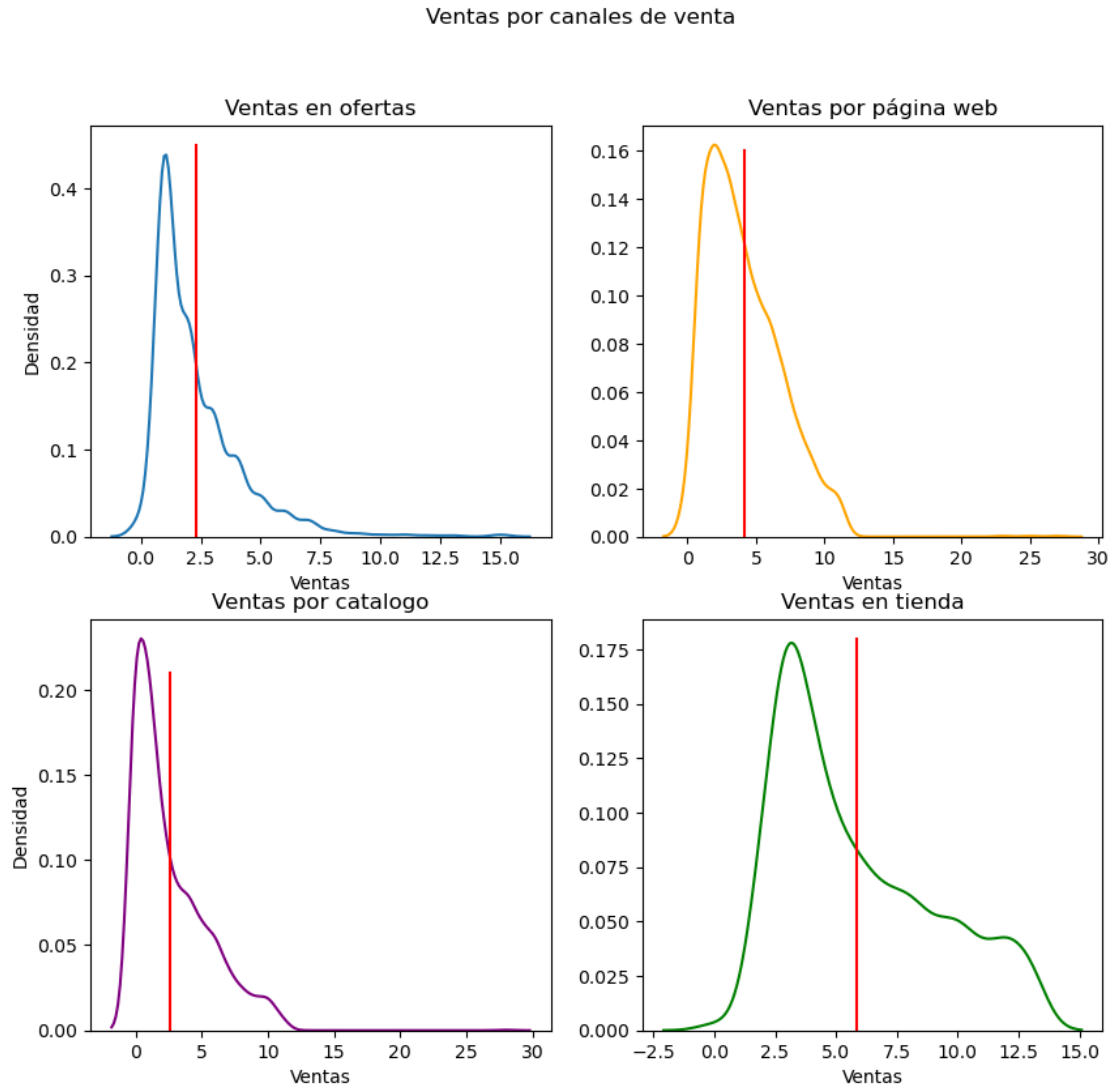
```
axes[0,1].plot([db2[a2[1]].mean(),db2[a2[1]].mean()], [0,0.16],color='red')
axes[0,1].set_xlabel('Ventas')
axes[0,1].set_ylabel('')

sns.kdeplot(ax=axes[1,0],data=db2[a2[2]],color='purple')
axes[1,0].plot([db2[a2[2]].mean(),db2[a2[2]].mean()], [0,0.21],color='red')
axes[1,0].set_xlabel('Ventas')
axes[1,0].set_ylabel('Densidad')

sns.kdeplot(ax=axes[1,1],data=db2[a2[3]],color='green')
axes[1,1].plot([db2[a2[3]].mean(),db2[a2[3]].mean()], [0,0.18],color='red')
axes[1,1].set_xlabel('Ventas')
axes[1,1].set_ylabel('')

#sns.kdeplot(ax=axes[0,1],db2['MntWines'])
```

[121]: Text(0, 0.5, '')



```
[122]: # Distribución por perfil del cliente

a2=['Recency', 'Age', 'Customer_Days', 'NumWebVisitsMonth']

fig,axes=plt.subplots(1,4,figsize=(15,3))

#fig.suptitle('Perfil del cliente')
axes[0].set_title('Dias entre consumos')
axes[1].set_title('Edad')
axes[2].set_title('Dias como cliente')
axes[3].set_title('Visitas mensuales')

sns.kdeplot(ax=axes[0],data=db2[a2[0]])
```

```

axes[0].plot([db2[a2[0]].mean(),db2[a2[0]].mean()], [0,0.011],color='red')

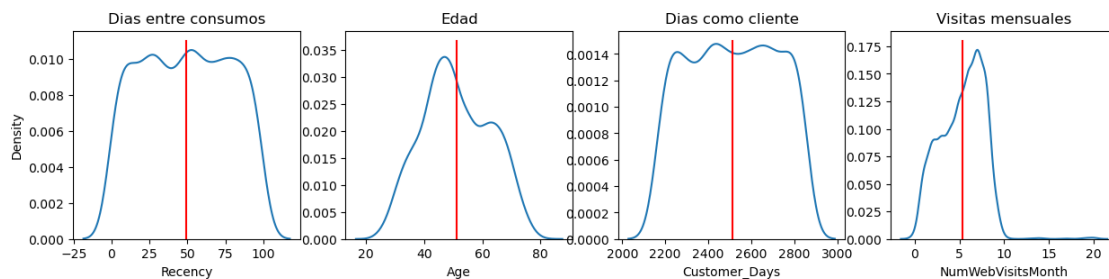
sns.kdeplot(ax=axes[1],data=db2[a2[1]])
axes[1].plot([db2[a2[1]].mean(),db2[a2[1]].mean()], [0,0.0367],color='red')
axes[1].set_ylabel('')

sns.kdeplot(ax=axes[2],data=db2[a2[2]])
axes[2].plot([db2[a2[2]].mean(),db2[a2[2]].mean()], [0,0.0015],color='red')
axes[2].set_ylabel('')

sns.kdeplot(ax=axes[3],data=db2[a2[3]])
axes[3].plot([db2[a2[3]].mean(),db2[a2[3]].mean()], [0,0.18],color='red')
axes[3].set_ylabel('')

```

[122]: Text(0, 0.5, '')



```

[123]: # Distribución de totales
a2=['MntTotal', 'MntRegularProds']

fig,axes=plt.subplots(1,2,figsize=(15,3))

#fig.suptitle('Perfil del cliente')
axes[0].set_title('Consumo de productos regulares')
axes[1].set_title('Consumo de productos totales')

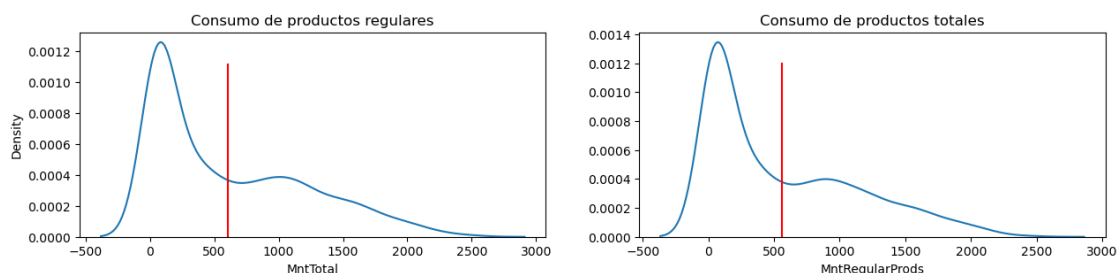
sns.kdeplot(ax=axes[0],data=db2[a2[0]])
axes[0].plot([db2[a2[0]].mean(),db2[a2[0]].mean()], [0,0.00111],color='red')

sns.kdeplot(ax=axes[1],data=db2[a2[1]])
axes[1].plot([db2[a2[1]].mean(),db2[a2[1]].mean()], [0,0.0012],color='red')
axes[1].set_ylabel('')

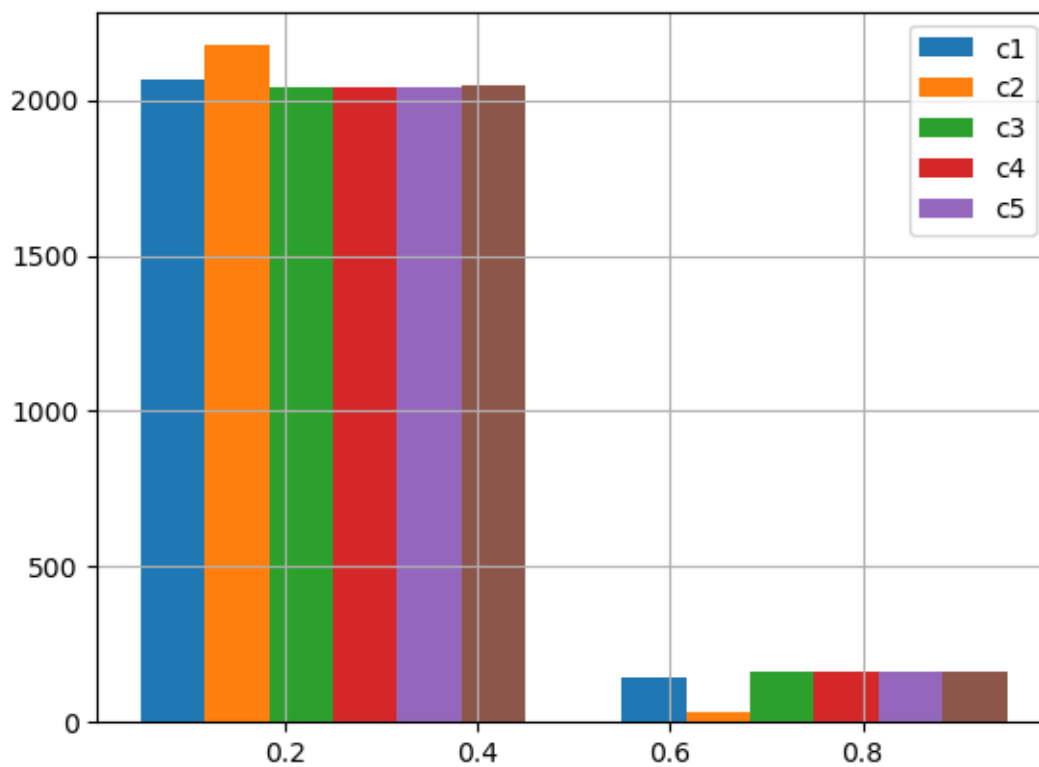
```

[123]: Text(0, 0.5, '')





```
[154]: # Histogramas de campañas
plt.hist(db2[['AcceptedCmp1', 'AcceptedCmp2', 'AcceptedCmp3',
            'AcceptedCmp4', 'AcceptedCmp4', 'AcceptedCmp5']], bins=2, histtype='bar',
        label=['c1', 'c2', 'c3', 'c4', 'c5'])
plt.legend(prop={'size': 10})
plt.grid()
```



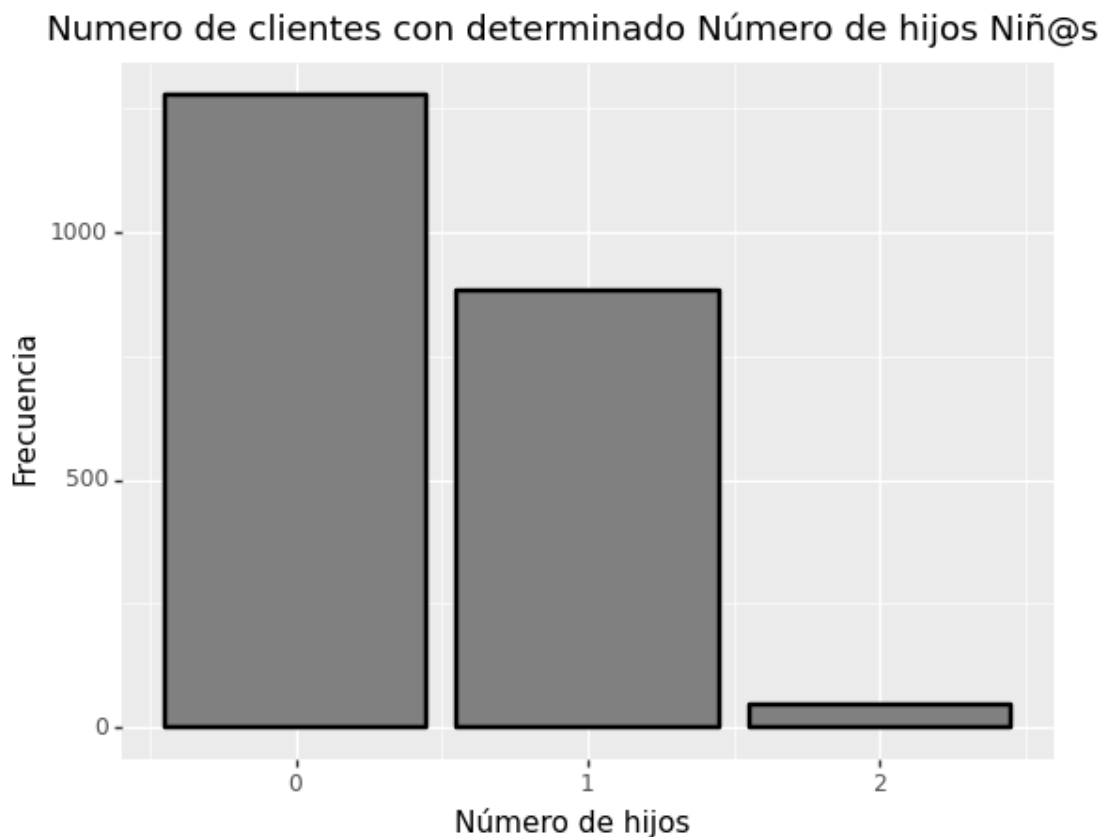
```
[ ]:
```

## 2 Librería ggplot (Parte 2)

Visitar la página oficial [ggplot](#) y con ayuda de [TWDSC](#) se puede descargar mediante la librería de **Plotnine**.

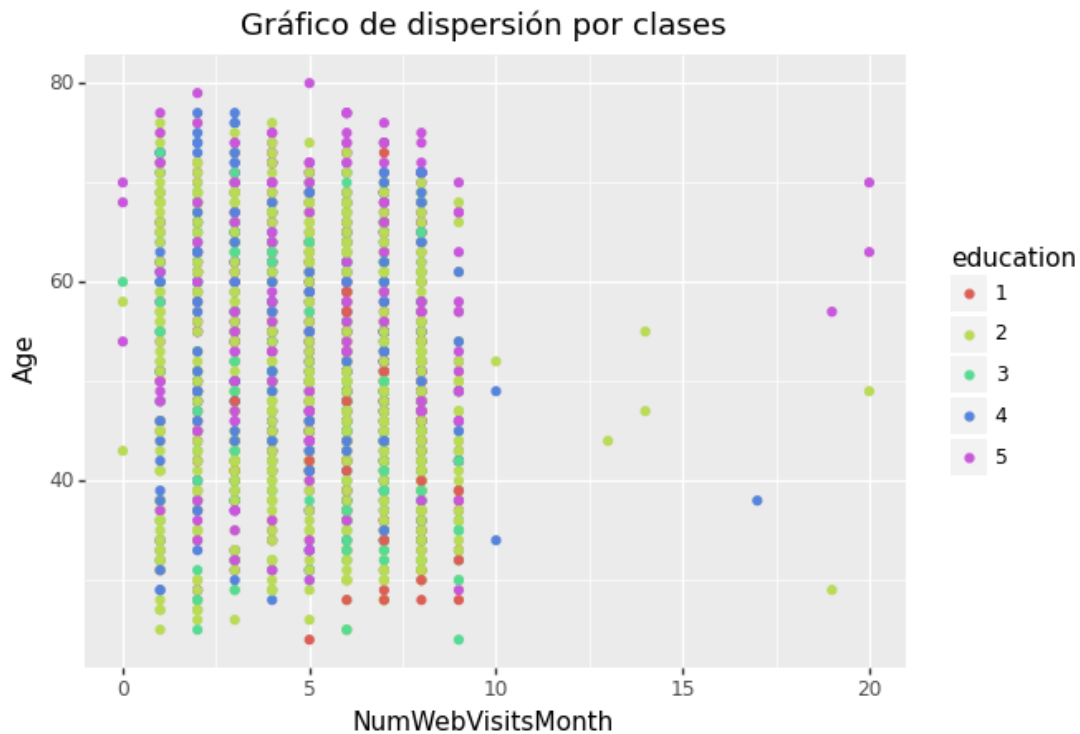
Tiene una alta correlación con el lenguaje R por lo que hay varias funciones que son realizadas de otra manera o no se pueden realizar, la versión es le 3.4.0

```
[124]: from plotnine import *
# 1 histograma
(ggplot(db2)+aes(x='Kidhome')+geom_bar(size=1,fill='gray',color='black')+labs(title='Numero_
↳de clientes con determinado Número de hijos Niñ@s'
,x='Número de_
↳hijos',y='Frecuencia'))
```



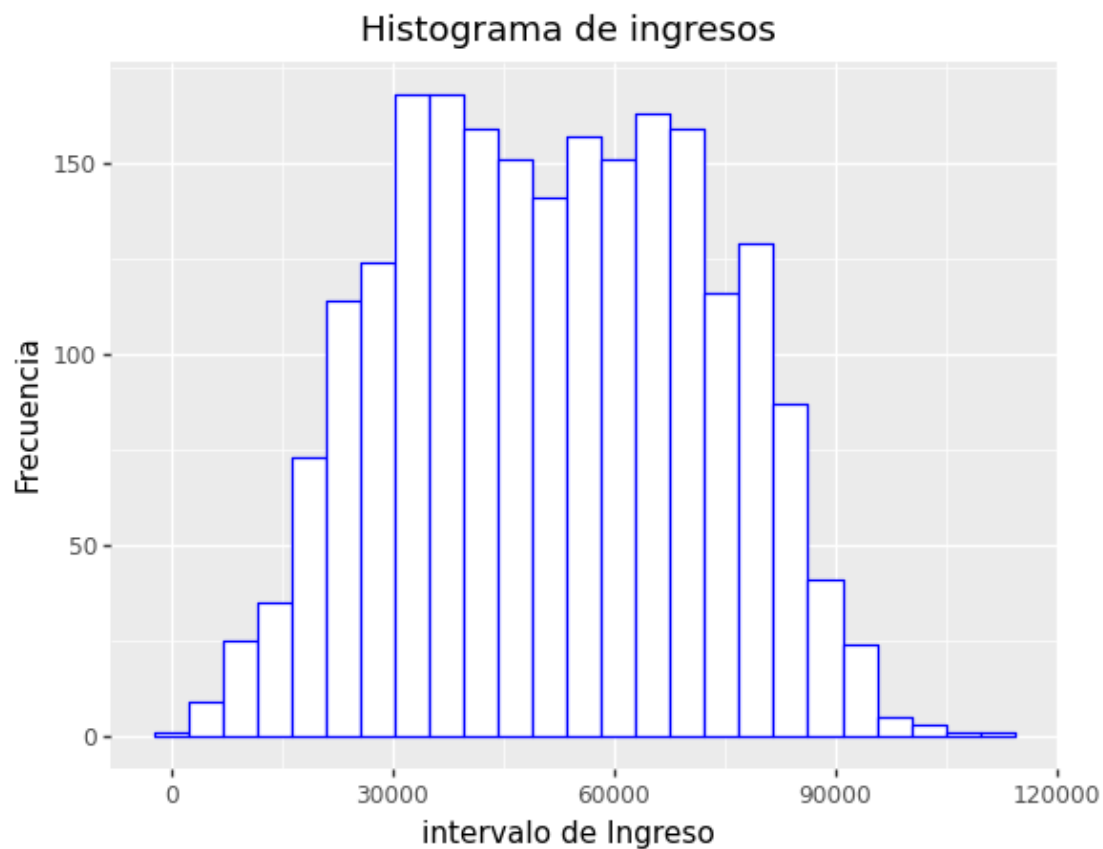
```
[124]: <ggplot: (174013262583)>
```

```
[125]: # 2 Dispersion por clases
(ggplot(db2)+aes(x='NumWebVisitsMonth',y='Age',color='education')+geom_point()+labs(title='Gráf
↳de dispersión por clases'))
```



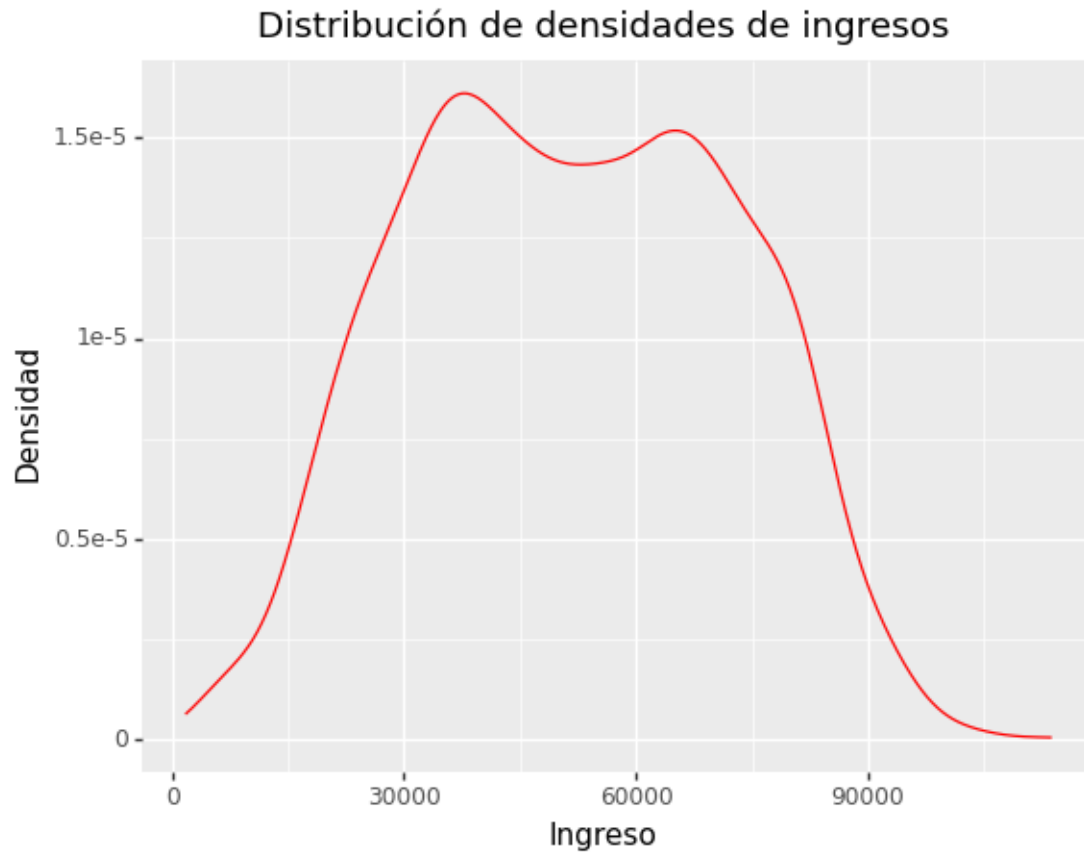
[125]: <ggplot: (174013294165)>

```
[126]: # 3 Histogram
(ggplot(db2)+geom_histogram(aes(x='Income'),bins=25,fill='white',color='blue')+
 labs(title='Histograma de ingresos',x='intervalo de Ingreso',y='Frecuencia'))
```



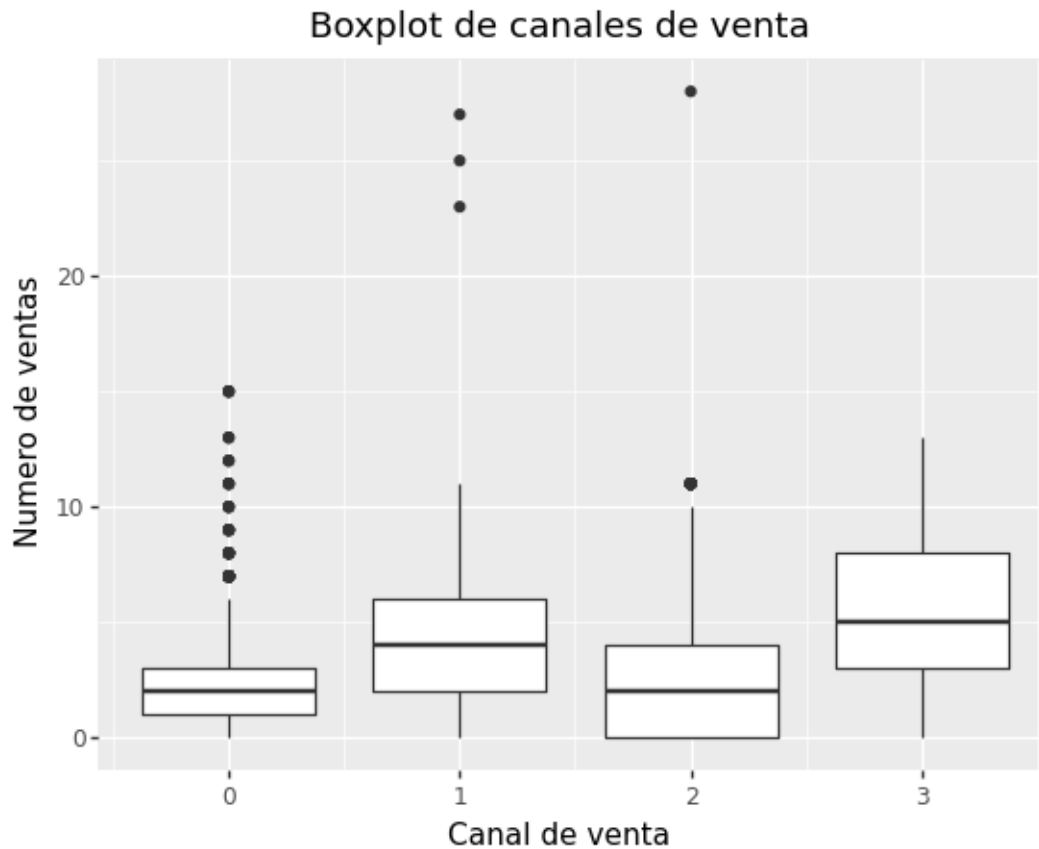
[126]: <ggplot: (174013293081)>

```
[127]: # 4 Density Curve
(ggplot(db2)+ geom_density(aes(x='Income'),color='Red')+
 labs(title='Distribución de densidades de ingresos',x='Ingreso',y='Densidad'))
```



[127]: <ggplot: (174013315578)>

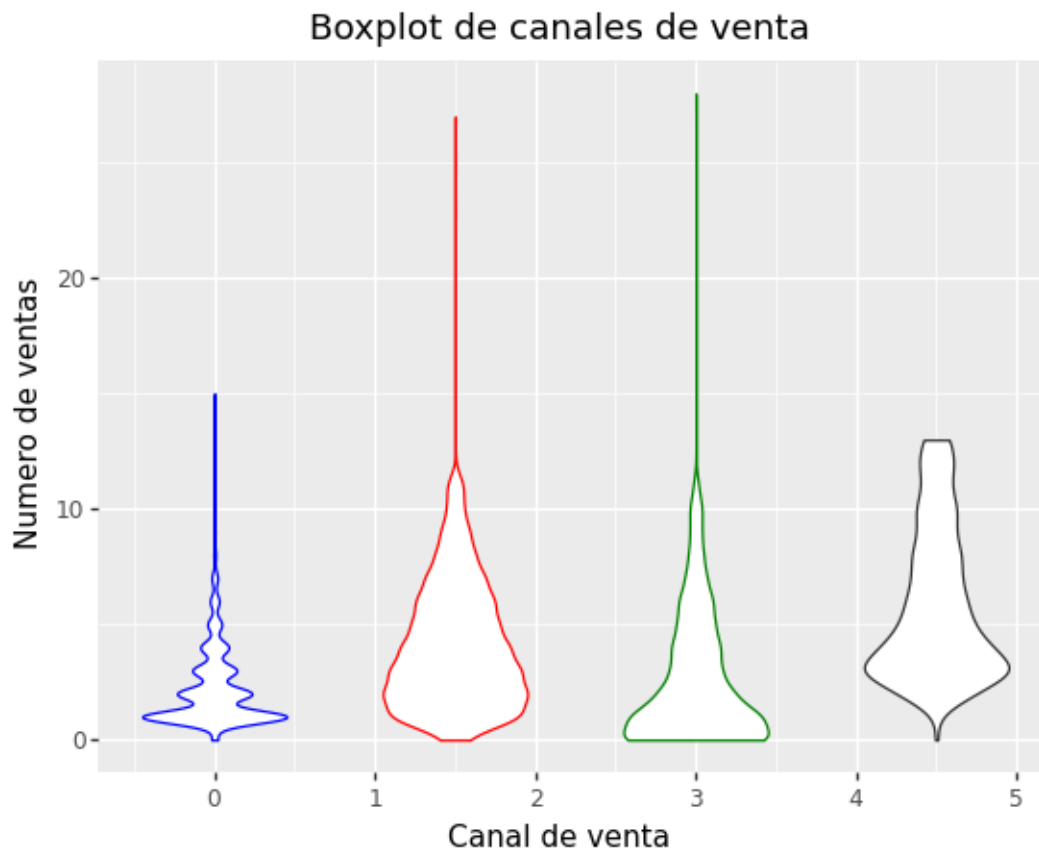
```
[128]: # 5 Boxplot
(ggplot(db2)+geom_boxplot(aes(x=0,y='NumDealsPurchases'))+
 geom_boxplot(aes(x=1,y='NumWebPurchases'))+
 geom_boxplot(aes(x=2,y='NumCatalogPurchases'))+
 geom_boxplot(aes(x=3,y='NumStorePurchases'))
+labs(title='Boxplot de canales de venta',x='Canal de venta',y='Numero de
↪ventas'))
# 'NumDealsPurchases', 'NumWebPurchases', 'NumCatalogPurchases',
↪'NumStorePurchases'
```



[128]: <ggplot: (174011471674)>

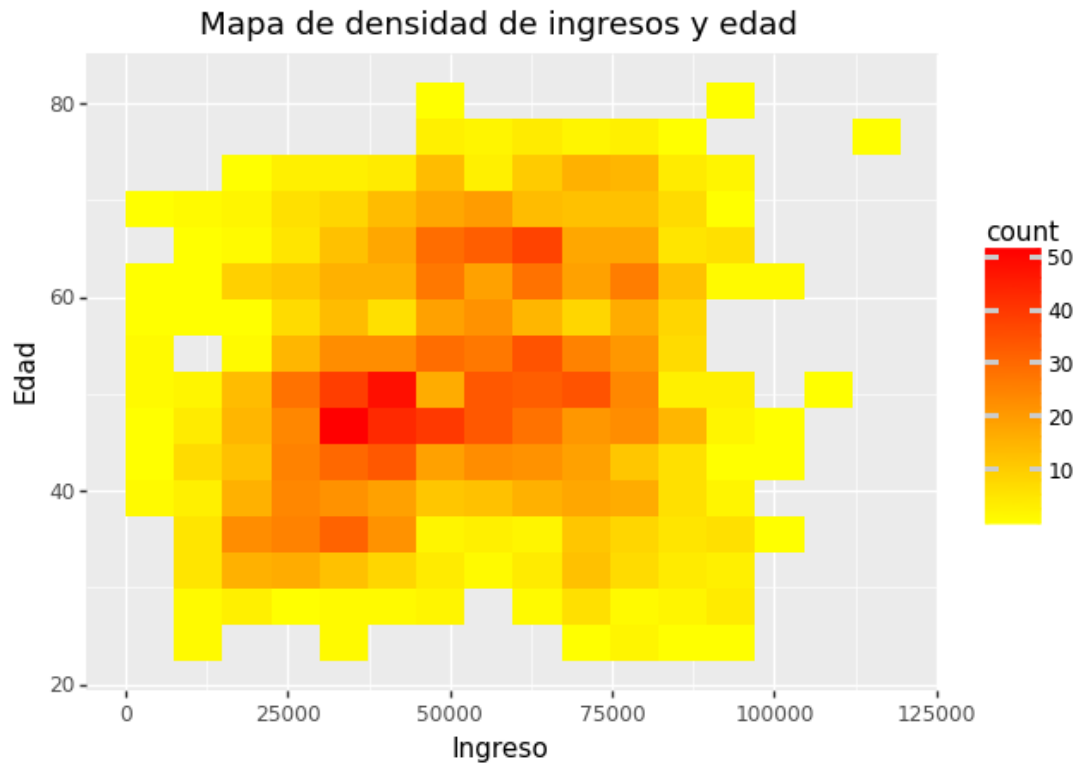
[129]: # 6 Violin

```
(ggplot(db2)+geom_violin(aes(x=0,y='NumDealsPurchases'),color='blue')+
  stat_ydensity(aes(x=1.5,y='NumWebPurchases'),color='red')+
  geom_violin(aes(x=3,y='NumCatalogPurchases'),color='green')+
  geom_violin(aes(x=4.5,y='NumStorePurchases'))
+labs(title='Boxplot de canales de venta',x='Canal de venta',y='Numero de_
↪ventas'))
```



[129]: <ggplot: (174001030289)>

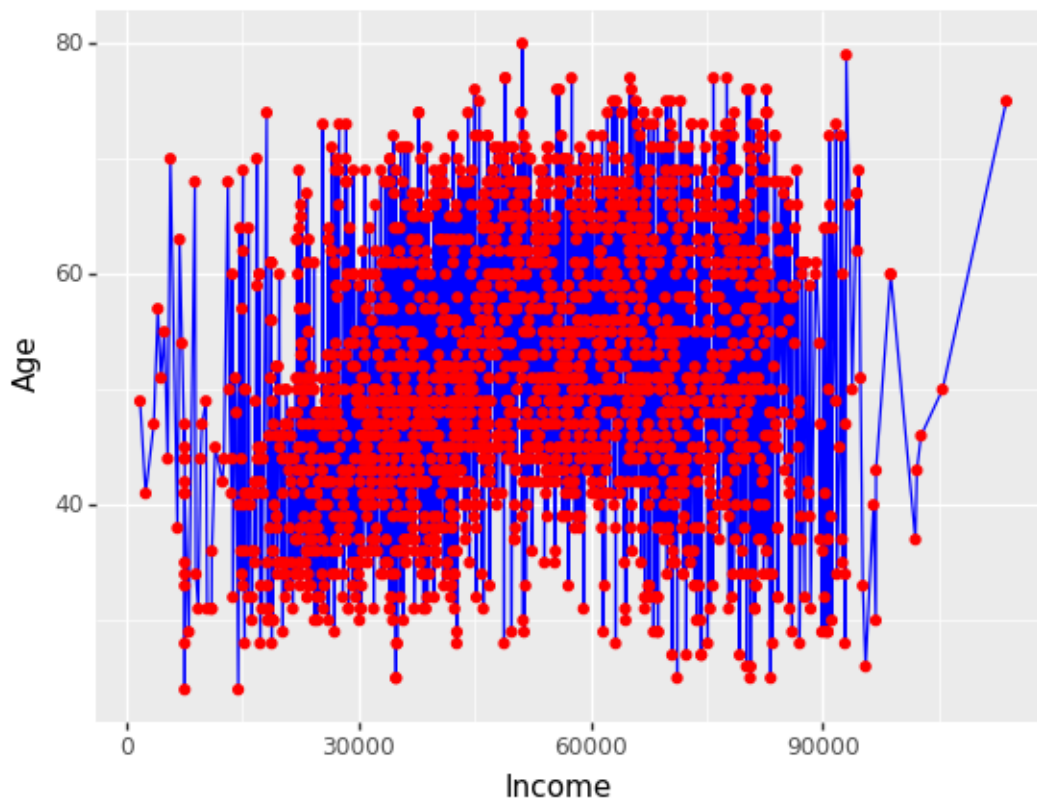
```
[130]: # 7 Density Map
(ggplot(db2)+aes(x='Income',y='Age')+geom_bin2d(bins=15)+scale_fill_gradient(high="red", low = "yellow")+
labs(title='Mapa de densidad de ingresos y edad',x='Ingreso',y='Edad'))
```



```
[130]: <ggplot: (174011186426)>
```

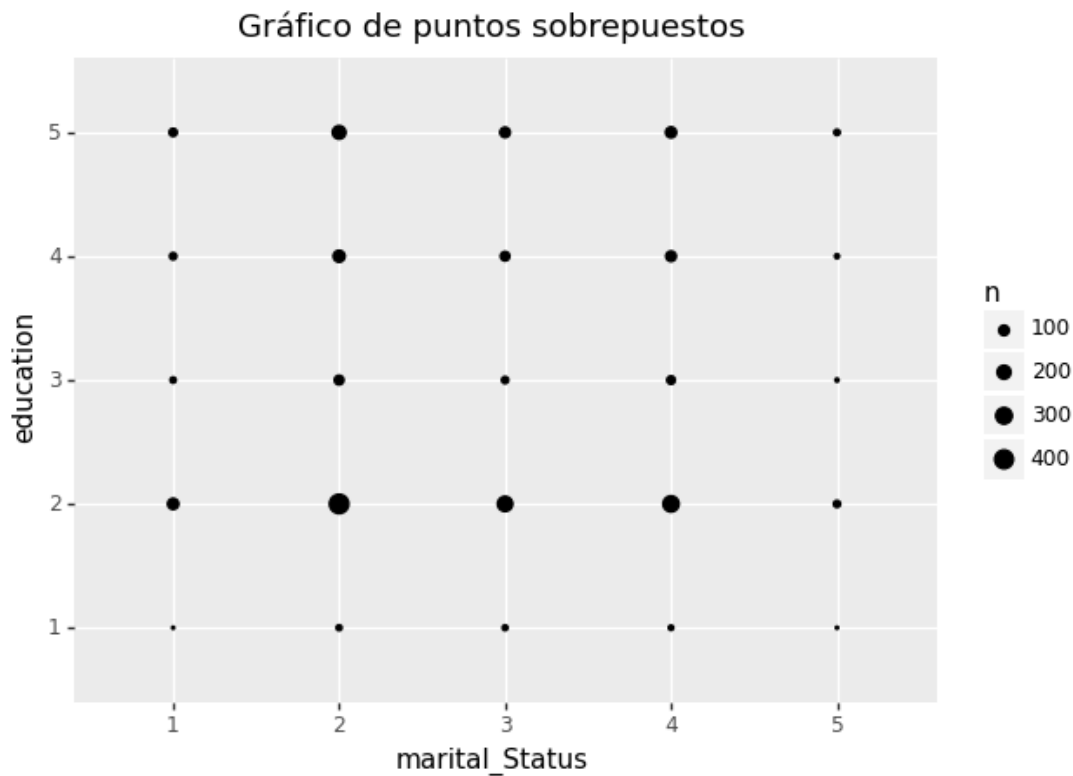
```
[131]: # 8 line  
(ggplot(db2)+aes(x='Income',y='Age')+geom_line(color='blue')+geom_point(color='red'))
```





```
[131]: <ggplot: (174011668658)>
```

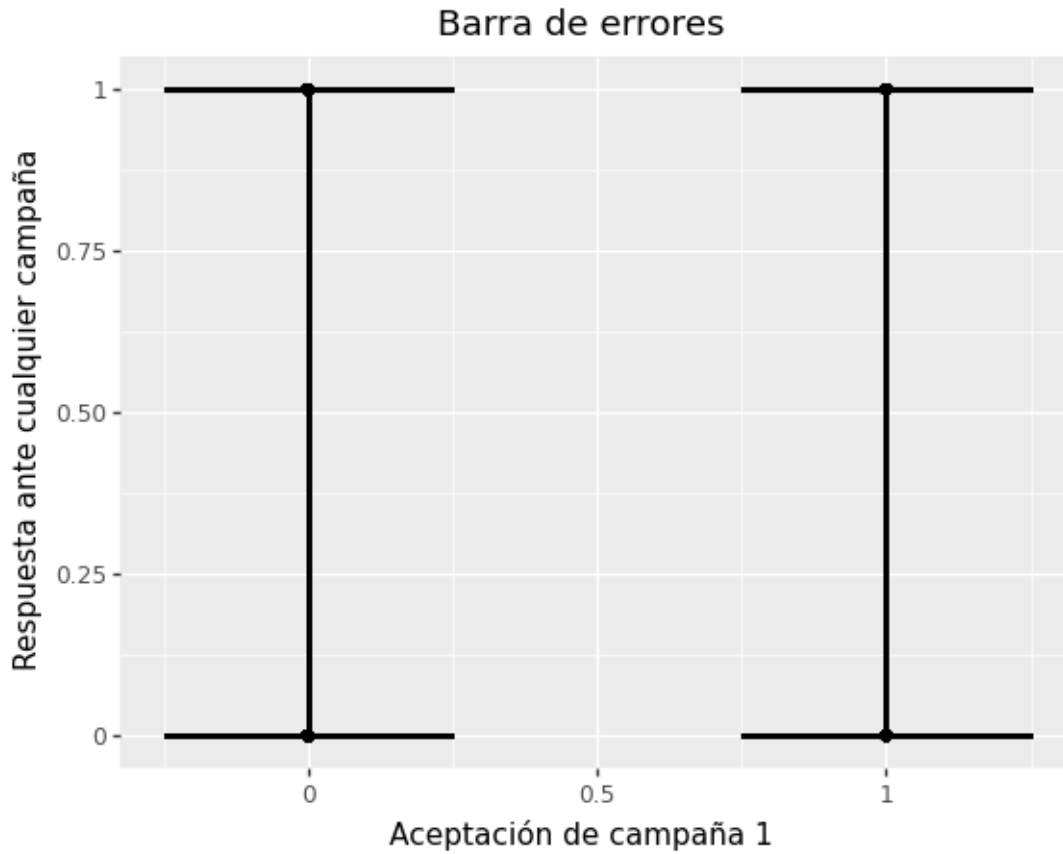
```
[132]: # 9 Count Overlapping points  
(ggplot(db2)+aes(x='marital_Status',y='education')+geom_count()+scale_size_area(max_size=4)+  
  labs(title='Gráfico de puntos sobrepuestos'))
```



[132]: <ggplot: (174011533809)>

[133]: #10 error bar

```
(ggplot(db2)+geom_point(aes(x='AcceptedCmp1',y='Response'))+geom_errorbar(aes(x='AcceptedCmp1',
+labs(title='Barra de errores',x='Aceptación de campaña 1',y='Respuesta ante_
↳cualquier campaña'))
```



[133]: <ggplot: (174010814461)>

### 3 Librería Altair (Parte 2)

Descarga mediane su pagina oficial: [Altair](#)

Instalación mediante pip:

```
pip install altair vega_datasets
```

Instalación para Anaconda:

```
conda install -c conda-forge altair vega_datasets
```

Ambas en la version 4.2.2 Se requieren los siguientes requisitos mandatorios para poderlo instalar (dependencies):

Python 3.6 o mayor

numpy

entrypoints

jsonschema

Pandas

Toolz

Teniendo algunas funciones como:

attachment:image.png

```
[134]: import altair as alt
from vega_datasets import data
## 1 Scatter plot
#iris=data.iris()
alt.Chart(db2).mark_point().encode(alt.X('Income'),alt.Y('MntTotal'),
                                   #alt.OpacityValue(1),alt.Color('Income')
                                   ).interactive()
#print(type(db2[['Income', 'NumWebPurchases']]))
```

[134]: alt.Chart(...)

```
[135]: # 2 3d Shapes
data = alt.graticule(step=[15, 15])

alt.Chart(data).mark_geoshape(stroke='black').project(
    'orthographic',
    rotate=[0, -45, 0]
)
```

[135]: alt.Chart(...)

```
[136]: # 3 time series con arrays
data = alt.sequence(0, 10, 0.1, as_='x')

alt.Chart(data).transform_calculate(
    y='sin(datum.x)'
).mark_line().encode(
    x='x:Q',
    y='y:Q',
)
```

[136]: alt.Chart(...)

```
[137]: # 4 Matriz de valores
from vega_datasets import data
```

```
#temps = data.seattle_temps()
#temps = temps[temps.date < '2010-01-15']
alt.Chart(db2).mark_rect().encode(
    alt.X('Income:Q', title='Income'),
    alt.Y('Age:Q', title='Age'),
    alt.Color('Teenhome:N', title='Numero de adolescentes')
)
```

[137]: alt.Chart(...)

```
[138]: # 5 Bar frequency
#data = pd.DataFrame({'name': ['a', 'b'], 'value': [4, 10]})

alt.Chart(db2).mark_bar(size=30).encode(
    x=('Kidhome:Q'),
    y='count()'
).properties(width=200)
```

[138]: alt.Chart(...)

```
[139]: # 6 Gráficas pulso con linea
from altair.expr import datum

stocks = data.stocks.url

base = alt.Chart(db2).encode(
    x='Income:T',
    y='MntFishProducts:Q'
    #color='symbol:N'
)

base.mark_line() + base.mark_point()

alt.layer(
    base.mark_line(),
    base.mark_point(),
    base.mark_rule()
).interactive()
```

[139]: alt.LayerChart(...)

```
[140]: import altair as alt
from vega_datasets import data
# Density Transform
alt.Chart(db2).transform_density(
    'MntTotal',
    as_=['MntTotal', 'density'],
```

```

).mark_area().encode(
    x="MntTotal:Q",
    y='density:Q',
)

```

[140]: alt.Chart(...)

```

[141]: # 8 Geo plots
from vega_datasets import data

# Data generators for the background
sphere = alt.sphere()
graticule = alt.graticule()

# Source of land data
source = alt.topo_feature(data.world_110m.url, 'countries')

# Layering and configuring the components
alt.layer(
    alt.Chart(sphere).mark_geoshape(fill='lightblue'),
    alt.Chart(graticule).mark_geoshape(stroke='white', strokeWidth=0.5),
    alt.Chart(source).mark_geoshape(fill='ForestGreen', stroke='black')
).project(
    'naturalEarth1'
).properties(width=600, height=400).configure_view(stroke=None)

```

[141]: alt.LayerChart(...)

```

[142]: # 9 Stream Graphs
import altair as alt
from vega_datasets import data

source = data.unemployment_across_industries.url

alt.Chart(source).mark_area().encode(
    alt.X('yearmonth(date):T',
        axis=alt.Axis(format='%Y', domain=False, tickSize=0)
    ),
    alt.Y('sum(count):Q', stack='center', axis=None),
    alt.Color('series:N',
        scale=alt.Scale(scheme='category20b')
    )
).interactive()

```

[142]: alt.Chart(...)

```
[143]: #10 Cumulative count chart

alt.Chart(db2).transform_window(
    cumulative_count="count()",
    cumulative_sum="sum(MntTotal)",
    sort=[{"field": "MntTotal"}],
).mark_area().encode(
    y="cumulative_sum:Q",
    x="cumulative_count:Q"
)
```

```
[143]: alt.Chart(...)
```

```
[ ]:
```

```
[ ]:
```

```
[ ]:
```