

I. Información general en relación al dataset y las características :

1. Carga de las bibliotecas necesarias

```
In [ ]: pip install Kneed
```

```
Collecting Kneed
```

```
  Downloading kneed-0.7.0-py2.py3-none-any.whl (9.4 kB)
```

```
Requirement already satisfied: matplotlib in /usr/local/lib/python3.7/dist-packages (from Kneed) (3.2.2)
```

```
Requirement already satisfied: numpy>=1.14.2 in /usr/local/lib/python3.7/dist-packages (from Kneed) (1.19.5)
```

```
Requirement already satisfied: scipy in /usr/local/lib/python3.7/dist-packages (from Kneed) (1.4.1)
```

```
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.7/dist-packages (from matplotlib->Kneed) (1.3.2)
```

```
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.1 in /usr/local/lib/python3.7/dist-packages (from matplotlib->Kneed) (2.4.7)
```

```
Requirement already satisfied: cyclor>=0.10 in /usr/local/lib/python3.7/dist-packages (from matplotlib->Kneed) (0.10.0)
```

```
Requirement already satisfied: python-dateutil>=2.1 in /usr/local/lib/python3.7/dist-packages (from matplotlib->Kneed) (2.8.2)
```

```
Requirement already satisfied: six in /usr/local/lib/python3.7/dist-packages (from cyclor>=0.10->matplotlib->Kneed) (1.15.0)
```

```
Installing collected packages: Kneed
```

```
Successfully installed Kneed-0.7.0
```

```
In [ ]: # Importacion de bibliotecas

import numpy as np
import pandas as pd
from sklearn.datasets import make_blobs
import matplotlib.pyplot as plt
from matplotlib import style
import seaborn as sns
from sklearn.mixture import GaussianMixture #GMM
from sklearn.pipeline import make_pipeline
from sklearn.preprocessing import StandardScaler
import warnings
warnings.filterwarnings('ignore')
import argparse
from sklearn.preprocessing import MinMaxScaler
from sklearn.decomposition import PCA
from sklearn.cluster import KMeans
from kneed import KneeLocator
import plotly.graph_objects as go
from plotly.subplots import make_subplots
```

Importar el Dataset 5: Customer Analysis

```
In [ ]: data = pd.read_csv('/content/drive/MyDrive/Data Sets/marketing_campaign.csv',
sep="\t") # se verifica el separador el cual es este caso corresponde a "\t"
```

```
In [ ]: data.head(4) # se tienen 29 características y 2240 observaciones
```

Out[]:

	ID	Year_Birth	Education	Marital_Status	Income	Kidhome	Teenhome	Dt_Customer	Rec
0	5524	1957	Graduation	Single	58138.0	0	0	04-09-2012	
1	2174	1954	Graduation	Single	46344.0	1	1	08-03-2014	
2	4141	1965	Graduation	Together	71613.0	0	0	21-08-2013	
3	6182	1984	Graduation	Together	26646.0	1	0	10-02-2014	

```
In [ ]: data.isnull().sum() # se observa la presencia de datos nulos
```

```
Out[ ]: ID                                0
        Year_Birth                        0
        Education                          0
        Marital_Status                    0
        Income                            24
        Kidhome                           0
        Teenhome                          0
        Dt_Customer                       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
        Z_CostContact                     0
        Z_Revenue                         0
        Response                          0
        dtype: int64
```

```
In [ ]: # Validando los tipos de datos en el dataset
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2240 entries, 0 to 2239
Data columns (total 29 columns):
#   Column                                Non-Null Count  Dtype
---  -
0   ID                                     2240 non-null   int64
1   Year_Birth                           2240 non-null   int64
2   Education                             2240 non-null   object
3   Marital_Status                       2240 non-null   object
4   Income                               2216 non-null   float64
5   Kidhome                              2240 non-null   int64
6   Teenhome                             2240 non-null   int64
7   Dt_Customer                          2240 non-null   object
8   Recency                              2240 non-null   int64
9   MntWines                             2240 non-null   int64
10  MntFruits                            2240 non-null   int64
11  MntMeatProducts                      2240 non-null   int64
12  MntFishProducts                     2240 non-null   int64
13  MntSweetProducts                    2240 non-null   int64
14  MntGoldProds                        2240 non-null   int64
15  NumDealsPurchases                   2240 non-null   int64
16  NumWebPurchases                     2240 non-null   int64
17  NumCatalogPurchases                 2240 non-null   int64
18  NumStorePurchases                   2240 non-null   int64
19  NumWebVisitsMonth                   2240 non-null   int64
20  AcceptedCmp3                        2240 non-null   int64
21  AcceptedCmp4                        2240 non-null   int64
22  AcceptedCmp5                        2240 non-null   int64
23  AcceptedCmp1                        2240 non-null   int64
24  AcceptedCmp2                        2240 non-null   int64
25  Complain                             2240 non-null   int64
26  Z_CostContact                       2240 non-null   int64
27  Z_Revenue                           2240 non-null   int64
28  Response                             2240 non-null   int64
dtypes: float64(1), int64(25), object(3)
memory usage: 507.6+ KB
```

2. Descripción de las Características

Se tienen 29 características

Customer Personality Analysis

El análisis de la personalidad del cliente es un análisis detallado de los clientes ideales de una empresa. Ayuda a una empresa a comprender mejor a sus clientes y les facilita la modificación de productos de acuerdo con las necesidades, los comportamientos y las preocupaciones específicas de los diferentes tipos de clientes.

El análisis de la personalidad del cliente ayuda a una empresa a modificar su producto en función de sus clientes objetivo de diferentes tipos de segmentos de clientes.

Información referente a las personas o clientes

- ID: identificador , codigo de cliente, valor único
- Year_Birth: el año de nacimiento de los clientes.
- Education: Nivel educativo de los clientes
- Marital_Status: Estado civil de los clientes
- Income: Ingreso de los clientes expresado en forma anual *Kidhome: Numero de niños por hogar**Teenhome: Numero de adolescentes por hogar* *Dt_Customer: Fecha en que el cliente se vinculó con la empresa**Recency: Numero de días desde la última compra* **Complain: Indicador si el cliente se a quejado en los ultimos años. 1 indica que sí. 0 indica que no.*

Información sobre los productos de consumo

- MntWines: El monto que los clientes gastan en vinos o bebidas alcoholicas en los ultimos 2 años. *MntFruits: El monto que los clientes gastan en frutas en los ultimos 2 años**MntMeatProducts: El monto que los clientes gastan en productos carnicos en los ultimos 2 años* *MntFishProducts: El monto que los clientes gastan en pescado en los ultimos 2 años**MntSweetProducts: El monto que los clientes gastan en dulces o golocinas en los ultimos 2 años* **MntGoldProds: El monto que los clientes gastan en productos de oro, en los ultimos dos años.*

Información sobre características promocionales en el proceso de compra

- NumDealsPurchases: Number of purchases made with a discount. *AcceptedCmp1: 1 indica si el cliente aceptó la oferta en la primer campaña, 0 que no.**AcceptedCmp2: 1 indica si el cliente aceptó la oferta en la segunda campaña , 0 que no.* *AcceptedCmp3: 1 indica si el cliente aceptó la oferta en la tercer campaña , 0 que no.**AcceptedCmp4: 1 indica si el cliente aceptó la oferta en la cuarta campaña , 0 que no.* *AcceptedCmp5: 1 indica si el cliente aceptó la oferta en la quinta campaña , 0 que no.**Response: 1 ndica si el cliente aceptó la oferta en la ultima campaña, 0 que no.*

Medios de compra o canal de compra, en que el cliente elige para el proceso de compra

- NumWebPurchases: Cantidad de compras realizadas por medio de la pagina web.
- NumCatalogPurchases: Cantidad de compras realizadas por medio del catalogo. *NumStorePurchases: Cantidad de compras realizadas directamente en las tiendas.**NumWebVisitsMonth: Cantidad de visitas a la pagina de la empresa en el ultimo mes.*

A. Asegurarse de la Fiabilidad de los datos

Se requiere realizar limpia de dato

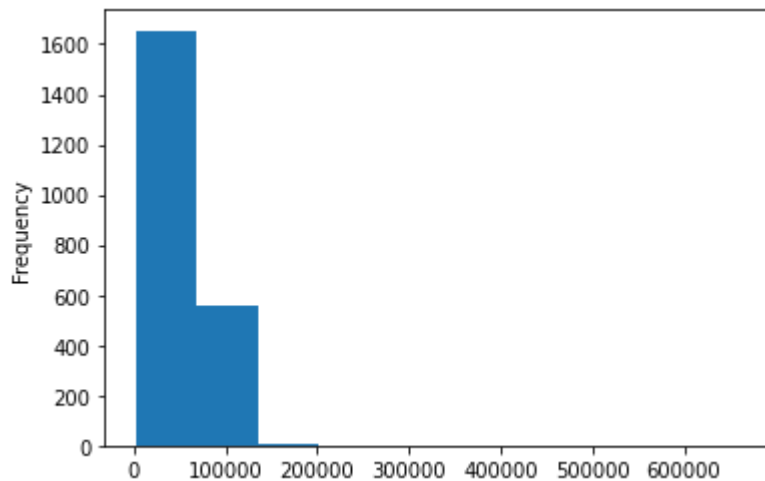
B. Aplicar procedimientos de limpiar e Imputación de datos

Limpieza de Datos

```
In [ ]: data['Income'].isnull().sum() # La característica que presenta nulos corresponde a La Income
```

Out[]: 24

```
In [ ]: data['Income'].plot.hist();  
sns.set(rc={'figure.figsize':(5,5)})  
plt.show() # con esto vemos la distribución de la característica, se observa que es del tipo distribución positiva, se debe imputar por la mediana
```



```
In [ ]: median = data['Income'].median() # se obtiene el valor de la mediana de la característica Income
```

```
In [ ]: data['Income'].fillna(median, inplace = True) # se procede a imputar por la mediana los valores nulos de la característica Income
```

```
In [ ]: data['Income'].isnull().sum() # se verifica que no queden nulos en la característica
```

Out[]: 0

```
In [ ]: # Validando los tipos de datos en el dataset
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2240 entries, 0 to 2239
Data columns (total 29 columns):
#   Column                                Non-Null Count  Dtype
---  ---                                -
0   ID                                    2240 non-null   int64
1   Year_Birth                           2240 non-null   int64
2   Education                            2240 non-null   object
3   Marital_Status                       2240 non-null   object
4   Income                               2240 non-null   float64
5   Kidhome                             2240 non-null   int64
6   Teenhome                             2240 non-null   int64
7   Dt_Customer                          2240 non-null   object
8   Recency                             2240 non-null   int64
9   MntWines                             2240 non-null   int64
10  MntFruits                             2240 non-null   int64
11  MntMeatProducts                       2240 non-null   int64
12  MntFishProducts                       2240 non-null   int64
13  MntSweetProducts                     2240 non-null   int64
14  MntGoldProds                         2240 non-null   int64
15  NumDealsPurchases                    2240 non-null   int64
16  NumWebPurchases                      2240 non-null   int64
17  NumCatalogPurchases                 2240 non-null   int64
18  NumStorePurchases                   2240 non-null   int64
19  NumWebVisitsMonth                   2240 non-null   int64
20  AcceptedCmp3                        2240 non-null   int64
21  AcceptedCmp4                        2240 non-null   int64
22  AcceptedCmp5                        2240 non-null   int64
23  AcceptedCmp1                        2240 non-null   int64
24  AcceptedCmp2                        2240 non-null   int64
25  Complain                             2240 non-null   int64
26  Z_CostContact                        2240 non-null   int64
27  Z_Revenue                           2240 non-null   int64
28  Response                             2240 non-null   int64
dtypes: float64(1), int64(25), object(3)
memory usage: 507.6+ KB
```

```
In [ ]: data.size # tenemos un tamaño de los datos de 64960
```

```
Out[ ]: 64960
```

EDA - Analisis Exploratorio de Datos

Exploracion de las características, realizar boxplot, histogramas, Medidas de Tendencia Central

```
In [ ]: data1 = data.copy() # se realiza una copia del data set para el proceso del E
DA y para transformaciones futuras
```

1. Analisis Exploratorio de la Característica ID

Esta característica corresponde a un ID unico que se le asigna al cliente, no se le cálcula medidas de tendencia central, es un codigo identificador del cliente

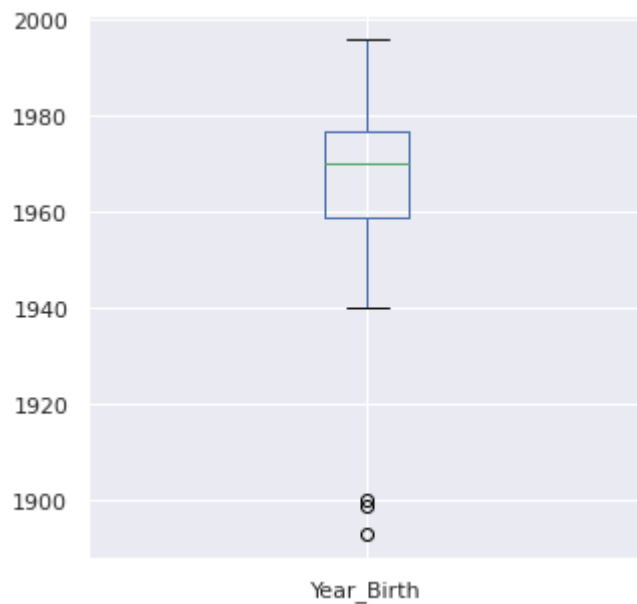
```
In [ ]: data1['ID'].duplicated().sum()    # se procede a verificar si existe algùn cod  
       igo duplicado, tal parece no hay datos duplicados en codigo del cliente
```

```
Out[ ]: 0
```

2. Analisis Exploratorio de la Característica **Year_Birth**

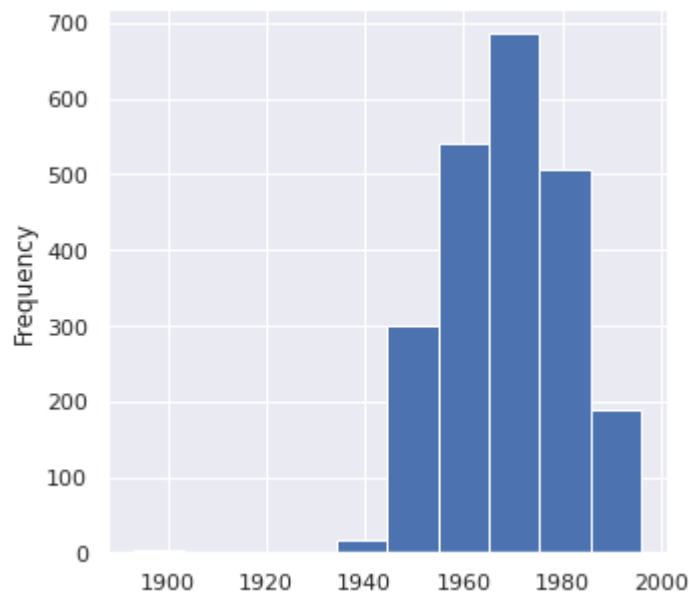
Corresponde al año de nacimiento de los clientes

```
In [ ]: data1['Year_Birth'].plot(kind='box')  
sns.set(rc={'figure.figsize':(5,5)})  
plt.show()
```



Histograma y Medidas de tendencia central y otros valores de relevancia en el análisis


```
In [ ]: data1['Year_Birth'].plot(kind='hist')
sns.set(rc={'figure.figsize':(5,5)})
plt.show()
```



```
In [ ]: data1['Year_Birth'].quantile(0.25)    # primer cuartil indica que un 25% de La
s personas presentan una fecha de nacimiento menor o igual a 1959
```

```
Out[ ]: 1959.0
```

```
In [ ]: data1['Year_Birth'].quantile(0.5)      # segundo cuartil indica que un 50% de La
s personas presentan una fecha de nacimiento menor o igual a 1970
```

```
Out[ ]: 1970.0
```

```
In [ ]: data1['Year_Birth'].quantile(0.75)     # tercer cuartil indica que un 75% de La
s personas presentan una fecha de nacimiento menor o igual a 1977
```

```
Out[ ]: 1977.0
```

```
In [ ]: data1['Year_Birth'].min()    # La fecha más antigua es 1893
```

```
Out[ ]: 1893
```

```
In [ ]: data1['Year_Birth'].max()    # La fecha mas reciente de nacimiento es 1996
```

```
Out[ ]: 1996
```

```
In [ ]: data1['Year_Birth'].mode()    # La fecha de nacimiento que más se repite es 19
76
```

```
Out[ ]: 0    1976
dtype: int64
```

```
In [ ]: data1['Year_Birth'].median() # La mediana indica que un 250% de las personas
      presentan una fecha de nacimiento menor o igual a 1970
```

```
Out[ ]: 1970.0
```

```
In [ ]: data1['Year_Birth'].mean() # el año promedio de nacimiento de los clientes es
      1968
```

```
Out[ ]: 1968.8058035714287
```

```
In [ ]: data1['Year_Birth'].std() # Los valores se alejan del promedio en 11.98
```

```
Out[ ]: 11.984069456885825
```

3. Analisis Exploratorio de la Característica **Education**

Presenta el nivel educativo de los clientes

Al ser una característica del tipo objeto no se le puede obtener el boxplot, mas adelante se procederá convertir en numérica.

```
In [ ]: data1['Education'].mode() # el nivel educativo que más está presente en los
      clientes es Graduation
```

```
Out[ ]: 0    Graduation
      dtype: object
```

```
In [ ]: data1['Education'].describe() # tenemos en la característica un total de 224
      0 bservaciones, cinco categorías distintas
      # 2n Cycle , Basic, Graduation, Master, PhD
```

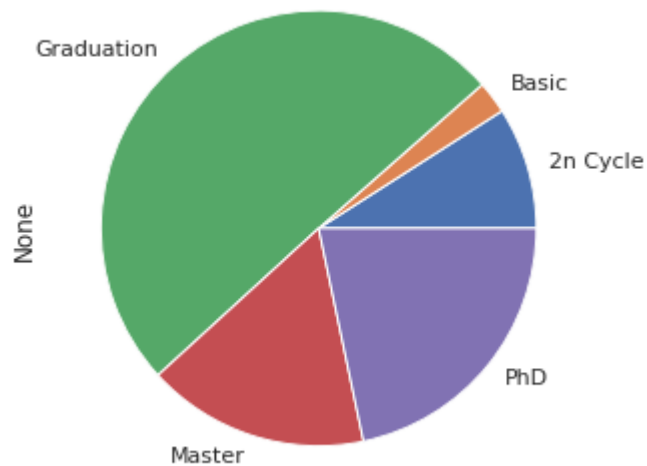
```
Out[ ]: count          2240
      unique           5
      top      Graduation
      freq          1127
      Name: Education, dtype: object
```

```
In [ ]: data1.groupby('Education').size() # Los niveles educativos de los clientes p
      redomida Graduation, seguido por PhD y en tercer lugar Master.
```

```
Out[ ]: Education
      2n Cycle      203
      Basic         54
      Graduation    1127
      Master        370
      PhD           486
      dtype: int64
```

```
In [ ]: data1.groupby('Education').size().plot(kind = 'pie') # se observa el peso de cada una de las niveles educativos de los clientes
```

```
Out[ ]: <matplotlib.axes._subplots.AxesSubplot at 0x7faaa5f3afd0>
```



4. Analisis Exploratorio de la Caracteristica **Marital_Status**

Indica el estado civil de los clientes, ejemplo casado, soltero o casado

```
In [ ]: data1.groupby('Marital_Status').size() # se observa la distribución del tipo de estatus marital de los clientes
```

```
Out[ ]: Marital_Status
Absurd      2
Alone        3
Divorced    232
Married     864
Single     480
Together    580
Widow        77
YOLO         2
dtype: int64
```

```
In [ ]: data1['Marital_Status'].mode() # el estado civil mas frecuente corresponde a Married = casados
```

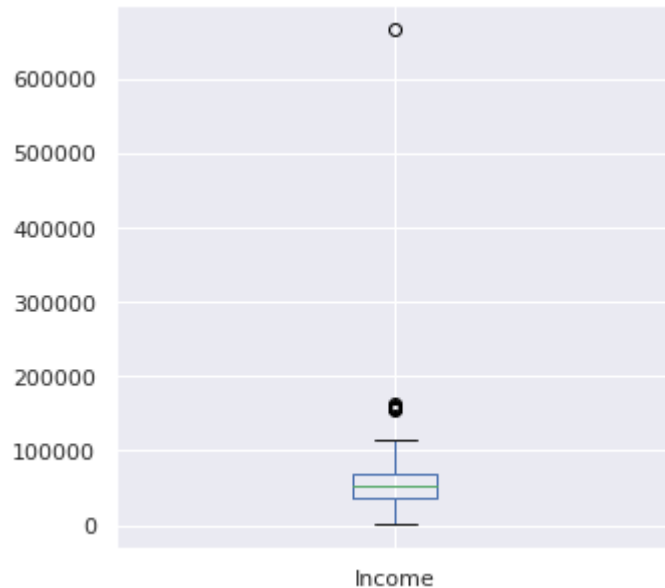
```
Out[ ]: 0    Married
dtype: object
```

```
In [ ]: data1['Marital_Status'].describe()
```

```
Out[ ]: count      2240
unique         8
top      Married
freq         864
Name: Marital_Status, dtype: object
```

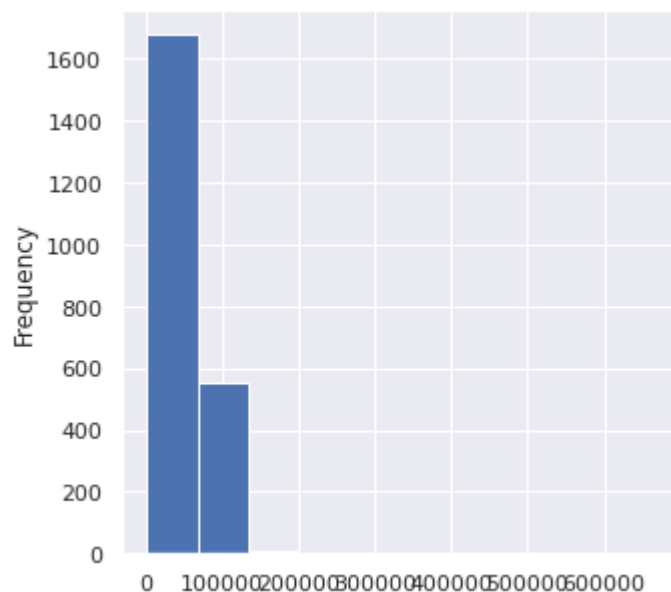
5. Analisis Exploratorio de la Característica **Income**

```
In [ ]: data1['Income'].plot(kind='box') # Income corresponde a los ingresos de los clientes
sns.set(rc={'figure.figsize':(5,5)})
plt.show()
```



Histograma y Medidas de tendencia central y otros valores de relevancia en el análisis

```
In [ ]: data1['Income'].plot(kind='hist') # distribucion del tipo positiva, acumulacion de datos al lado izquierdo
sns.set(rc={'figure.figsize':(5,5)})
plt.show()
```



```
In [ ]: data1['Income'].quantile(0.25)    # indica que un 25% de las personas presenta  
n un valor menor o igual de ingresos a 35 538
```

```
Out[ ]: 35538.75
```

```
In [ ]: data1['Income'].quantile(0.5)    # indica que un 50% de las personas presentan  
un valor menor o igual de ingresos a 51 381
```

```
Out[ ]: 51381.5
```

```
In [ ]: data1['Income'].quantile(0.75)    # indica que un 75% de las personas presenta  
n un valor menor o igual de 68 289
```

```
Out[ ]: 68289.75
```

```
In [ ]: data1['Income'].min()    # el ingreso mínimo es de 1730
```

```
Out[ ]: 1730.0
```

```
In [ ]: data1['Income'].max()    # el ingreso máximo es de 666 666
```

```
Out[ ]: 666666.0
```

```
In [ ]: data1['Income'].mode()    # el ingreso que más se repite es de 51 381
```

```
Out[ ]: 0    51381.5  
dtype: float64
```

```
In [ ]: data1['Income'].median()    # la mediana indica que un 50% de las personas pres  
entan un valor menor o igual de 51 381.5
```

```
Out[ ]: 51381.5
```

```
In [ ]: data1['Income'].mean()    # el ingreso promedio es de 52 237
```

```
Out[ ]: 52237.97544642857
```

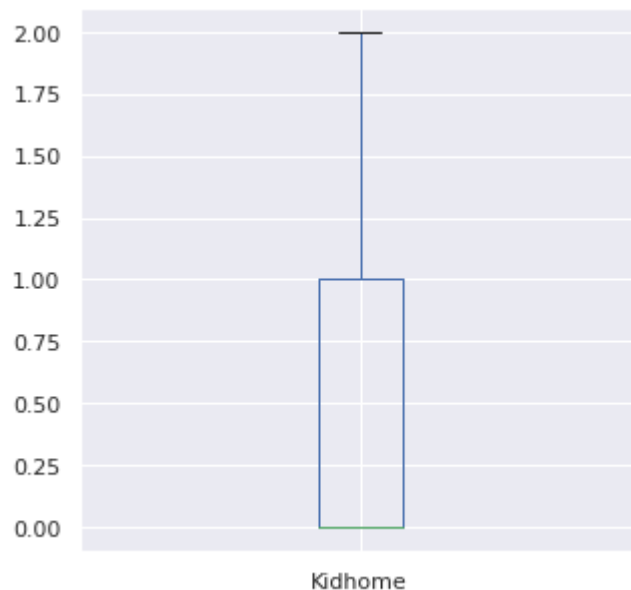
```
In [ ]: data1['Income'].std()    # los valores del ingreso se desvían del promedio en 2  
5 037
```

```
Out[ ]: 25037.955890621957
```

6. Análisis Exploratorio de la Característica **Kidhome**

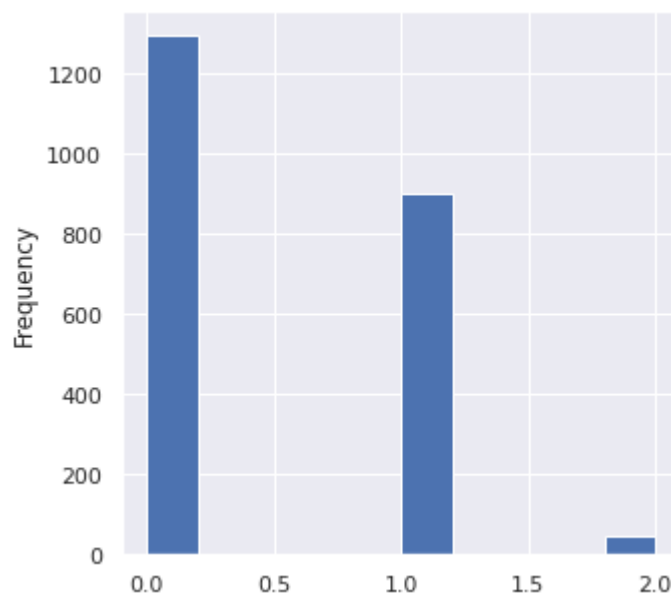
Indica la cantidad de niños por hogar de los clientes

```
In [ ]: data1['Kidhome'].plot(kind='box') # el vlaor minimo de niños por ho
gar corresponde a cero, el vlaor maximo corresponde a 2
sns.set(rc={'figure.figsize':(5,5)})
plt.show()
```



Histograma y Medidas de tendencia central y otros valores de relevancia en el análisis

```
In [ ]: data1['Kidhome'].plot(kind='hist') # distribucion con caracter
isticas del tipo positiva, valores acumulados a la izquierda
sns.set(rc={'figure.figsize':(5,5)})
plt.show()
```



```
In [ ]: data1['Kidhome'].quantile(0.25) # indica que un 25% de las personas presen
tan cero hijos
```

```
Out[ ]: 0.0
```

```
In [ ]: data1['Kidhome'].quantile(0.5)    # indica que un 50% de las personas presenta  
n cero hijos
```

```
Out[ ]: 0.0
```

```
In [ ]: data1['Kidhome'].quantile(0.75)  # indica que un 75% de las personas presentan  
uno o menos hijos
```

```
Out[ ]: 1.0
```

```
In [ ]: data1['Kidhome'].min()    # el valor mas bajo de hijos por hogar es cero
```

```
Out[ ]: 0
```

```
In [ ]: data1['Kidhome'].max()    # el valor mas alto de niños por hogar es de 2
```

```
Out[ ]: 2
```

```
In [ ]: data1['Kidhome'].mode()    # el valor que mas se repite es cero
```

```
Out[ ]: 0    0  
dtype: int64
```

```
In [ ]: data1['Kidhome'].median()    # La mediana indica que un % de las personas prese  
ntan cero hijos
```

```
Out[ ]: 0.0
```

```
In [ ]: data1['Kidhome'].mean()    # en promedio las familias presentan menos de un hi  
jo por hogar
```

```
Out[ ]: 0.44419642857142855
```

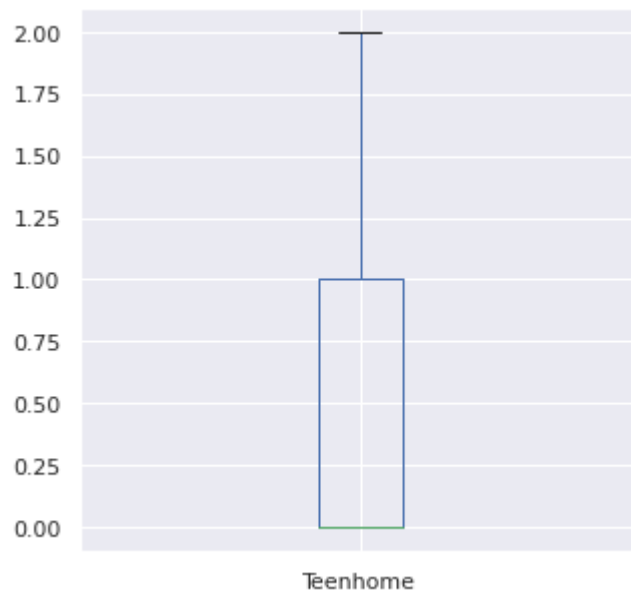
```
In [ ]: data1['Kidhome'].std()    # los valores se alejan del promedio en 0.53
```

```
Out[ ]: 0.5383980977345874
```

7. Analisis Exploratorio de la Caracteristica **Teenhome**

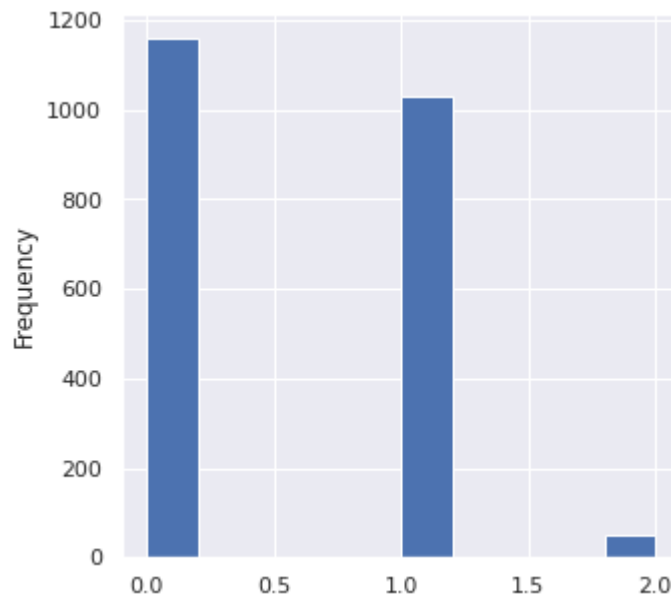
Cantidad de adolescentes por hogar de los clientes

```
In [ ]: data1['Teenhome'].plot(kind='box')    # el minimo de hijos adolescentes por h
ogar es de cero, el maximo es de cero
sns.set(rc={'figure.figsize':(5,5)})
plt.show()
```



Histograma y Medidas de tendencia central y otros valores de relevancia en el análisis

```
In [ ]: data1['Teenhome'].plot(kind='hist')
sns.set(rc={'figure.figsize':(5,5)})
plt.show()
```



```
In [ ]: data1['Teenhome'].quantile(0.25)    # indica que un 25% de las personas presen
tan un valor de cero adolescentes por hogar
```

```
Out[ ]: 0.0
```



```
In [ ]: data1['Teenhome'].quantile(0.5)  # indica que un 50% de las personas presenta  
n un valor de cero adolescentes por hogar
```

```
Out[ ]: 0.0
```

```
In [ ]: data1['Teenhome'].quantile(0.75)  # indica que un 75 % de las personas presen  
tan un valor menor o igual de a un hijo adolescente por hogar
```

```
Out[ ]: 1.0
```

```
In [ ]: data1['Teenhome'].min()  # el valor minimo es de cero
```

```
Out[ ]: 0
```

```
In [ ]: data1['Teenhome'].max()  # el valor maximo es de 2
```

```
Out[ ]: 2
```

```
In [ ]: data1['Teenhome'].mode()  # el valor mas repetido es de cero hijos
```

```
Out[ ]: 0    0  
dtype: int64
```

```
In [ ]: data1['Teenhome'].median()  # indica que un 50% de las personas presentan un  
valor de cero adolescentes por hogar
```

```
Out[ ]: 0.0
```

```
In [ ]: data1['Teenhome'].mean()  # el valor promedio de hijos es menor a uno por hoga  
r
```

```
Out[ ]: 0.50625
```

```
In [ ]: data1['Teenhome'].std()  # Los valores se desvian del promedio en 0.54
```

```
Out[ ]: 0.5445382307698755
```

8. Analisis Exploratorio de la Caracteristica **Dt_Customer**

Fecha de inscripción del cliente en la empresa.

```
In [ ]: data1.groupby('Dt_Customer').size()
```

```
Out[ ]: Dt_Customer
01-01-2013    4
01-01-2014    3
01-02-2013    3
01-02-2014    1
01-03-2013    3
..
31-08-2012   12
31-08-2013    8
31-10-2012    5
31-12-2012    1
31-12-2013    3
Length: 663, dtype: int64
```

```
In [ ]: data1['Dt_Customer'] = pd.to_datetime(data1.Dt_Customer)
```

```
In [ ]: data1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2240 entries, 0 to 2239
Data columns (total 29 columns):
#   Column                Non-Null Count  Dtype
---  -
0   ID                    2240 non-null   int64
1   Year_Birth            2240 non-null   int64
2   Education             2240 non-null   object
3   Marital_Status        2240 non-null   object
4   Income                2240 non-null   float64
5   Kidhome               2240 non-null   int64
6   Teenhome              2240 non-null   int64
7   Dt_Customer           2240 non-null   datetime64[ns]
8   Recency               2240 non-null   int64
9   MntWines              2240 non-null   int64
10  MntFruits             2240 non-null   int64
11  MntMeatProducts       2240 non-null   int64
12  MntFishProducts       2240 non-null   int64
13  MntSweetProducts      2240 non-null   int64
14  MntGoldProds          2240 non-null   int64
15  NumDealsPurchases     2240 non-null   int64
16  NumWebPurchases       2240 non-null   int64
17  NumCatalogPurchases  2240 non-null   int64
18  NumStorePurchases     2240 non-null   int64
19  NumWebVisitsMonth     2240 non-null   int64
20  AcceptedCmp3          2240 non-null   int64
21  AcceptedCmp4          2240 non-null   int64
22  AcceptedCmp5          2240 non-null   int64
23  AcceptedCmp1          2240 non-null   int64
24  AcceptedCmp2          2240 non-null   int64
25  Complain              2240 non-null   int64
26  Z_CostContact         2240 non-null   int64
27  Z_Revenue             2240 non-null   int64
28  Response              2240 non-null   int64
dtypes: datetime64[ns](1), float64(1), int64(25), object(2)
memory usage: 507.6+ KB
```

Medidas de tendencia central y otros valores de relevancia en el análisis

```
In [ ]: data1['Dt_Customer'].quantile(0.25)      # indica que un 25% de las personas pr
esentan una fecha menor o igual de inscripcion con la empresa al 2016-01-09
```

```
Out[ ]: Timestamp('2013-01-19 18:00:00')
```

```
In [ ]: data1['Dt_Customer'].quantile(0.50)      # indica que un 50 % de las personas
presentan un valor menor o igual de la fecha de inscripcion al 2013-07-11
```

```
Out[ ]: Timestamp('2013-07-11 00:00:00')
```

```
In [ ]: data1['Dt_Customer'].quantile(0.75)      # indica que un 75% de las personas p  
resentan un valor menor o igual de la fecha de inscripcion al 2013-12-30
```

```
Out[ ]: Timestamp('2013-12-30 06:00:00')
```

```
In [ ]: data1['Dt_Customer'].min()      # la fecha minima a 2012-01-08
```

```
Out[ ]: Timestamp('2012-01-08 00:00:00')
```

```
In [ ]: data1['Dt_Customer'].max()      # la fecha maxima 2014.12-06
```

```
Out[ ]: Timestamp('2014-12-06 00:00:00')
```

```
In [ ]: data1['Dt_Customer'].mode()      # la fecha que mas se repite es 2012-08-31
```

```
Out[ ]: 0    2012-08-31  
dtype: datetime64[ns]
```

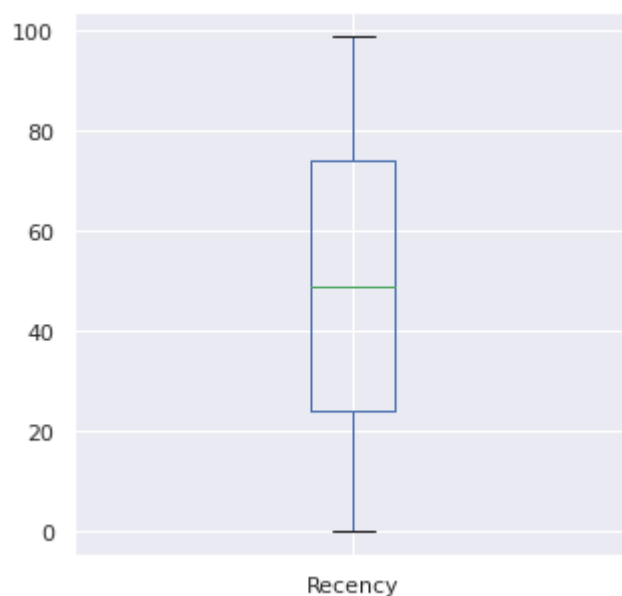
```
In [ ]: data1['Dt_Customer'].mean()
```

```
Out[ ]: Timestamp('2013-07-11 22:57:38.571432192')
```

9. Analisis Exploratorio de la Caracteristica **Recency**

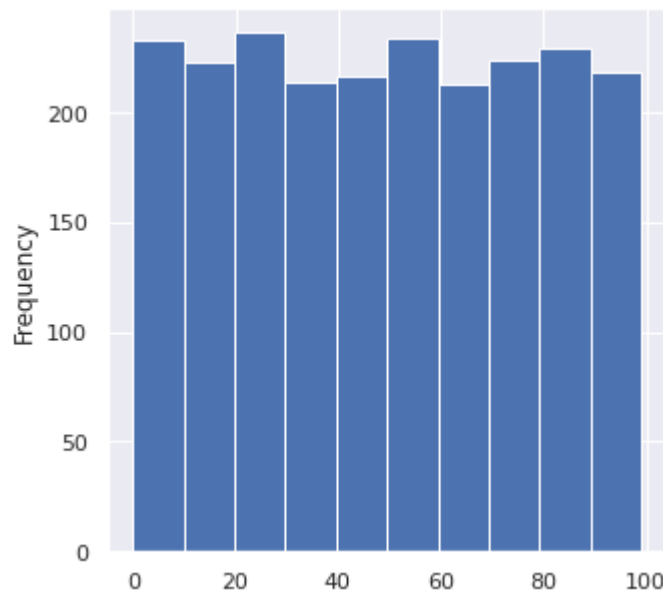
Número de días desde la última compra del cliente.

```
In [ ]: data1['Recency'].plot(kind='box')      # La menor cantidad de dias  
que una persona presenta desde su ultima compra es de cero, y el maximo de dia  
s 99 .  
sns.set(rc={'figure.figsize':(5,5)})  
plt.show()
```



Histograma y Medidas de tendencia central y otros valores de relevancia en el análisis

```
In [ ]: data1['Recency'].plot(kind='hist')
sns.set(rc={'figure.figsize':(5,5)})
plt.show()
```



```
In [ ]: data1['Recency'].quantile(0.25)    # indica que un 25% de las personas presen
tan un valor menor o igual de 24 días desde la última compra
```

```
Out[ ]: 24.0
```

```
In [ ]: data1['Recency'].quantile(0.5)     # indica que un 50% de las personas presenta
n un valor menor o igual de 49 días desde la última compra
```

```
Out[ ]: 49.0
```

```
In [ ]: data1['Recency'].quantile(0.75)    # indica que un 75% de las personas presenta
n un valor menor o igual de 74 días desde la última compra
```

```
Out[ ]: 74.0
```

```
In [ ]: data1['Recency'].min()             # el valor mínimo es cero
```

```
Out[ ]: 0
```

```
In [ ]: data1['Recency'].max()             # el valor máximo es de 99 días
```

```
Out[ ]: 99
```

```
In [ ]: data1['Recency'].mode()            # el valor que más se repite es 56 días
```

```
Out[ ]: 0    56
dtype: int64
```

```
In [ ]: data1['Recency'].median()          # indica que un 50% de las personas presentan un v
alor menor o igual de 49 días desde la última compra
```

```
Out[ ]: 49.0
```

```
In [ ]: data1['Recency'].mean()    # el valor promedio de dias desde la ultima compra
```

```
Out[ ]: 49.109375
```

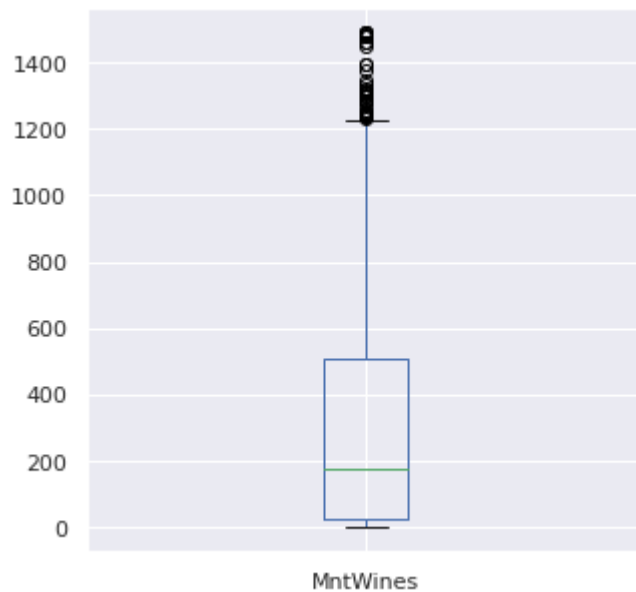
```
In [ ]: data1['Recency'].std()    # Los valores se alejan del promedio en 28.96
```

```
Out[ ]: 28.962452808378206
```

10. Analisis Exploratorio de la Caracteristica **MntWines**

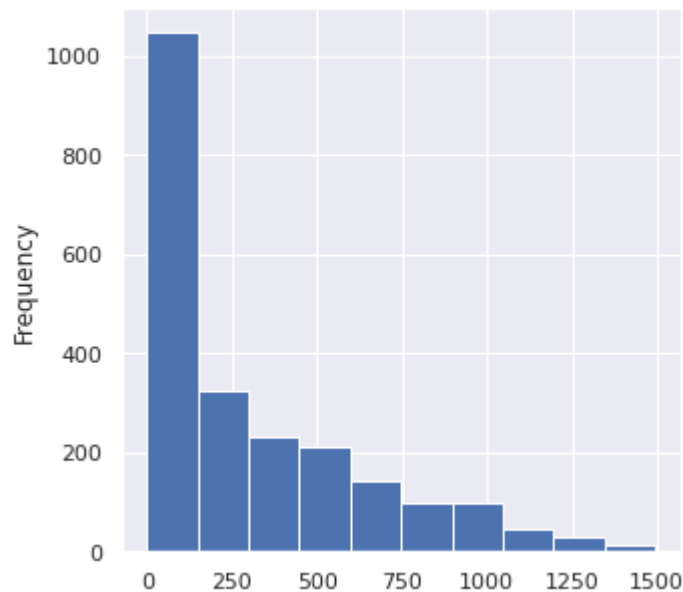
Cantidad gastada en vino en los últimos 2 años

```
In [ ]: data1['MntWines'].plot(kind='box')    # se observan valores atipicos , el valor minimo de cero , el valor maximo de 1493, La mediana de 173
sns.set(rc={'figure.figsize':(5,5)})
plt.show()
```



Histograma y Medidas de tendencia central y otros valores de relevancia en el análisis

```
In [ ]: data1['MntWines'].plot(kind='hist')      # distribucion con comportamiento tip
o positiva
sns.set(rc={'figure.figsize':(5,5)})
plt.show()
```



```
In [ ]: data1['MntWines'].quantile(0.25)  # indica que un 25% de las personas present
an un valor menor o igual de compra de bebidas de 23.75 en los últimos 2 años
```

Out[]: 23.75

```
In [ ]: data1['MntWines'].quantile(0.5)    # indica que un 50% de las personas presenta
n un valor menor o igual a 173.5 de compras en bebidas
```

Out[]: 173.5

```
In [ ]: data1['MntWines'].quantile(0.75)   # indica que un 75% de las personas present
an un valor menor o igual a 504.25 en bebidas en los últimos dos años
```

Out[]: 504.25

```
In [ ]: data1['MntWines'].min()            # el valor mínimo es de cero
```

Out[]: 0

```
In [ ]: data1['MntWines'].max()            # el valor máximo es de 1493
```

Out[]: 1493

```
In [ ]: data1['MntWines'].mode()          # el valor que más se repite es 2
```

Out[]: 0 2
dtype: int64

```
In [ ]: data1['MntWines'].median() # indica que un 50% de las personas presentan un  
valor menor o igual a 173
```

```
Out[ ]: 173.5
```

```
In [ ]: data1['MntWines'].mean() # el valor promedio de compras en productos de bebida  
as es de 303.93
```

```
Out[ ]: 303.9357142857143
```

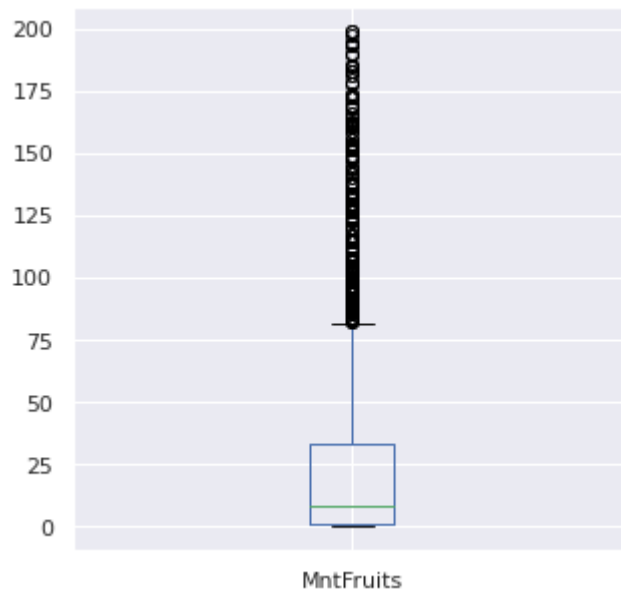
```
In [ ]: data1['MntWines'].std() # Los valores se desvian del promedio en 336.59
```

```
Out[ ]: 336.5973926053715
```

11. Analisis Exploratorio de la Caracteristica **MntFruits**

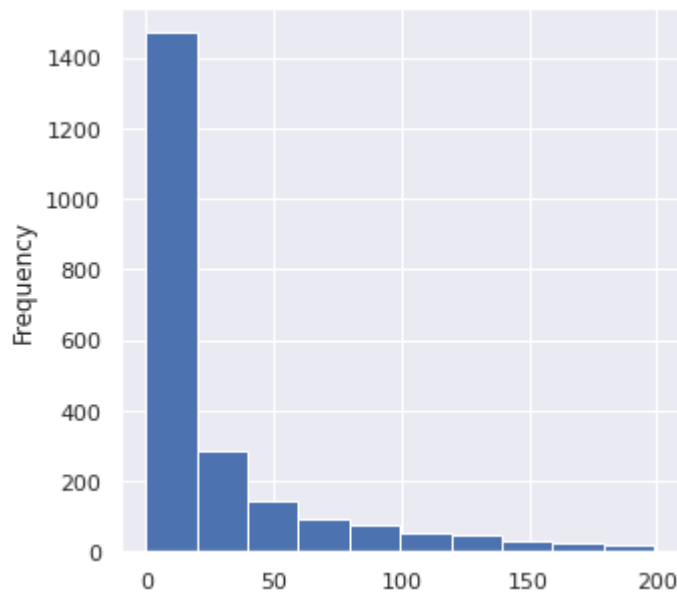
Cantidad gastada en frutas en los últimos 2 años.

```
In [ ]: data1['MntFruits'].plot(kind='box') # se observan valores atipicos, valor  
inimo cero , valor maximo de 199, mediana de 8  
sns.set(rc={'figure.figsize':(5,5)})  
plt.show()
```



Histograma y Medidas de tendencia central y otros valores de relevancia en el análisis


```
In [ ]: data1['MntFruits'].plot(kind='hist') # distribucion con comportamiento positivo, acumulacion de la datos a la izquierda.  
sns.set(rc={'figure.figsize':(5,5)})  
plt.show()
```



```
In [ ]: data1['MntFruits'].quantile(0.25) # indica que un 25% de las personas presentan un valor menor o igual 1 en el consumo de frutas
```

```
Out[ ]: 1.0
```

```
In [ ]: data1['MntFruits'].quantile(0.5) # indica que un 50% de las personas presentan un valor menor o igual 8 en el consumo de frutas
```

```
Out[ ]: 8.0
```

```
In [ ]: data1['MntFruits'].quantile(0.75) # indica que un 75% de las personas presentan un valor menor o igual 33 en el consumo de frutas
```

```
Out[ ]: 33.0
```

```
In [ ]: data1['MntFruits'].min() # el valor minimo es de cero
```

```
Out[ ]: 0
```

```
In [ ]: data1['MntFruits'].max() # el valor maximo es de 199 lo que una persona a gastado en los ultimos dos años en frutas
```

```
Out[ ]: 199
```

```
In [ ]: data1['MntFruits'].mode() # el valor mas repetido de lo que las personas gastan en frutas es de cero
```

```
Out[ ]: 0 0  
dtype: int64
```

```
In [ ]: data1['MntFruits'].median() # indica que un 50% de las personas presentan un valor menor o igual 8 en lo que gastan las personas en frutas
```

```
Out[ ]: 8.0
```

```
In [ ]: data1['MntFruits'].mean() # el valor promedio de lo que las personas gastan en frutas en los dos últimos 2 años en frutas es de 26.30
```

```
Out[ ]: 26.302232142857143
```

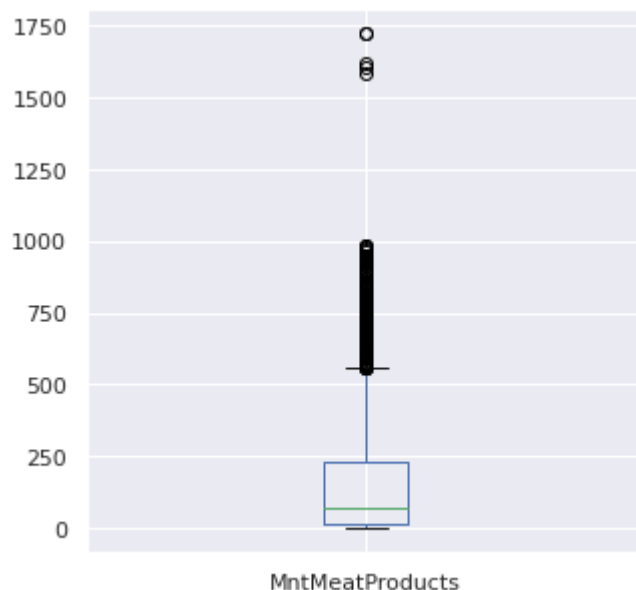
```
In [ ]: data1['MntFruits'].std() # los valores se pueden desviar del promedio en 39.77
```

```
Out[ ]: 39.77343376457866
```

12. Analisis Exploratorio de la Característica **MntMeatProducts**

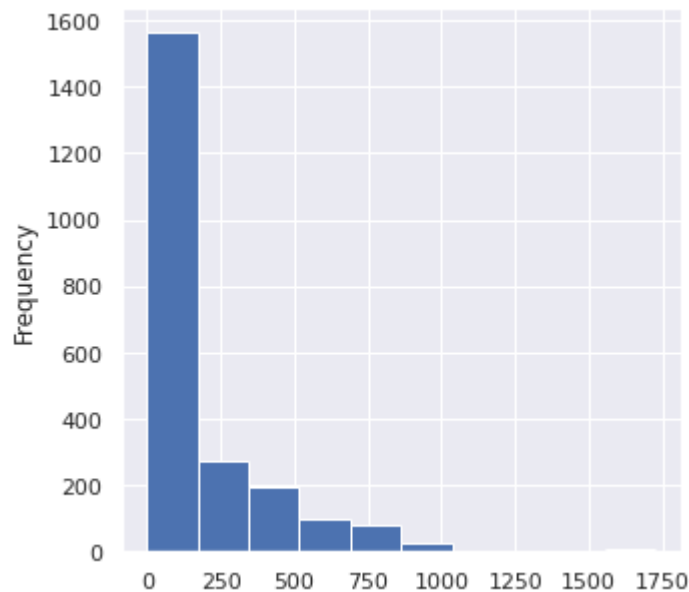
Cantidad gastada en carne en los últimos 2 años.

```
In [ ]: data1['MntMeatProducts'].plot(kind='box') # se observan valores atipicos
sns.set(rc={'figure.figsize':(5,5)})
plt.show()
```



Histograma y Medidas de tendencia central y otros valores de relevancia en el análisis

```
In [ ]: data1['MntMeatProducts'].plot(kind='hist')    # distribucion con comportamient  
o tipo positivo  
sns.set(rc={'figure.figsize':(5,5)})  
plt.show()
```



```
In [ ]: data1['MntMeatProducts'].quantile(0.25)    # indica que un 25% de las personas  
presentan un valor menor o igual 16 en lo que han gastado en productos carnic  
os los ultimos 2 años
```

Out[]: 16.0

```
In [ ]: data1['MntMeatProducts'].quantile(0.5)    # indica que un 50% de las personas p  
resentan un valor menor o igual a 67 en lo que las personas han gastado en los  
ultimos 2 años en productos carnicos
```

Out[]: 67.0

```
In [ ]: data1['MntMeatProducts'].quantile(0.75)    # indica que un 75% de las personas  
presentan un valor menor o igual 232 en lo que las personas han gastado en lo  
s ultimos dos años en productos carnicos
```

Out[]: 232.0

```
In [ ]: data1['MntMeatProducts'].min()    # el valor mas bajo es de cero
```

Out[]: 0

```
In [ ]: data1['MntMeatProducts'].max()    # el valor mas alto es de 1725
```

Out[]: 1725

```
In [ ]: data1['MntMeatProducts'].median()    # indica que un 50% de las personas presen  
tan un valor menor o igual a 67
```

Out[]: 67.0

```
In [ ]: data1['MntMeatProducts'].mean() # el monto promedio que las personas han gastado en productos carnicos los ultimos 2 años es de 166.95
```

```
Out[ ]: 166.95
```

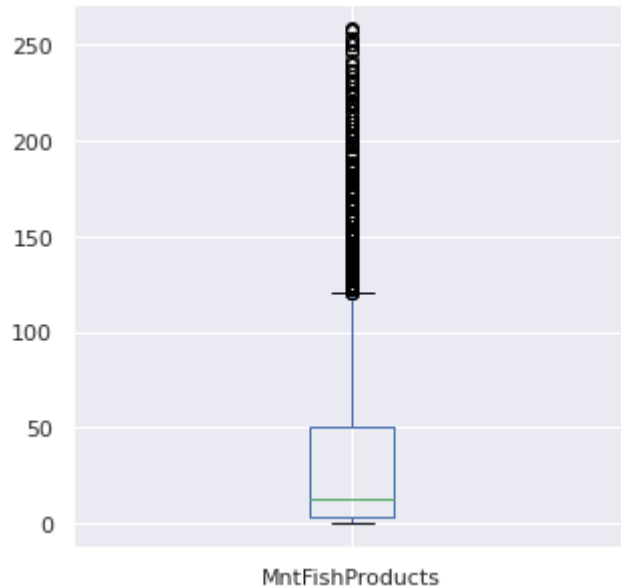
```
In [ ]: data1['MntMeatProducts'].std() # los valores se desvian del promedio en 225.71
```

```
Out[ ]: 225.71537251175434
```

13. Analisis Exploratorio de la Caracteristica **MntFishProducts**

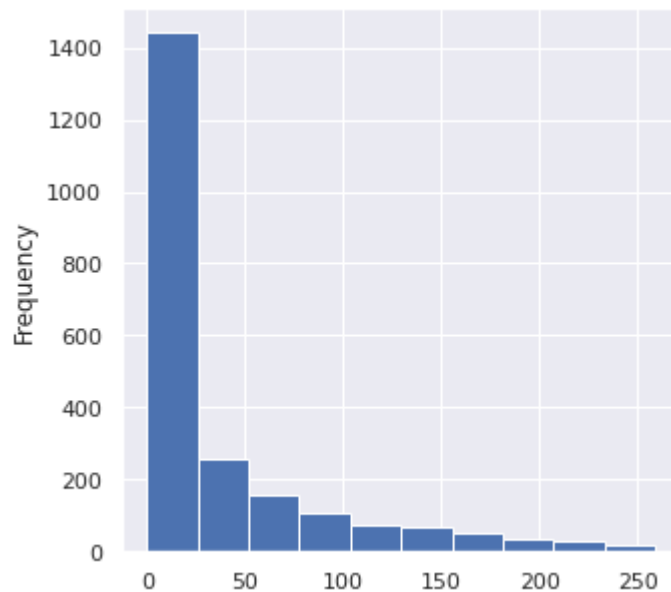
Cantidad gastada en pescado en los últimos 2 años.

```
In [ ]: data1['MntFishProducts'].plot(kind='box')
sns.set(rc={'figure.figsize':(5,5)})
plt.show()
```



Histograma y Medidas de tendencia central y otros valores de relevancia en el análisis

```
In [ ]: data1['MntFishProducts'].plot(kind='hist') # comportamiento de los datos con distribucion del tipo positiva
sns.set(rc={'figure.figsize':(5,5)})
plt.show()
```



```
In [ ]: data1['MntFishProducts'].quantile(0.25) # indica que un 25% de las personas
presentan un valor menor o igual a 3 en el lo que gastan en pescado
```

Out[]: 3.0

```
In [ ]: data1['MntFishProducts'].quantile(0.5) # indica que un 50% de las personas p
resentan un valor menor o igual a 12 en lo que gastan en pescado
```

Out[]: 12.0

```
In [ ]: data1['MntFishProducts'].quantile(0.75) # indica que un 75% de las personas
presentan un valor menor o igual a 50 en lo que gastan en pescado
```

Out[]: 50.0

```
In [ ]: data1['MntFishProducts'].min() # el valor mas bajo es de cero
```

Out[]: 0

```
In [ ]: data1['MntFishProducts'].max() # el valor mas alto de 259
```

Out[]: 259

```
In [ ]: data1['MntFishProducts'].mode() # el valor que mas se repite es d ecero
```

Out[]: 0 0
dtype: int64

```
In [ ]: data1['MntFishProducts'].median() # indica que un 50 % de las personas pres  
entan un valor menor o igual a 12 en lo que las personas gastan en pescado
```

```
Out[ ]: 12.0
```

```
In [ ]: data1['MntFishProducts'].mean() # En promedio lo que las personas han gastad  
o los últimos dos años en pescado es de 37.52
```

```
Out[ ]: 37.52544642857143
```

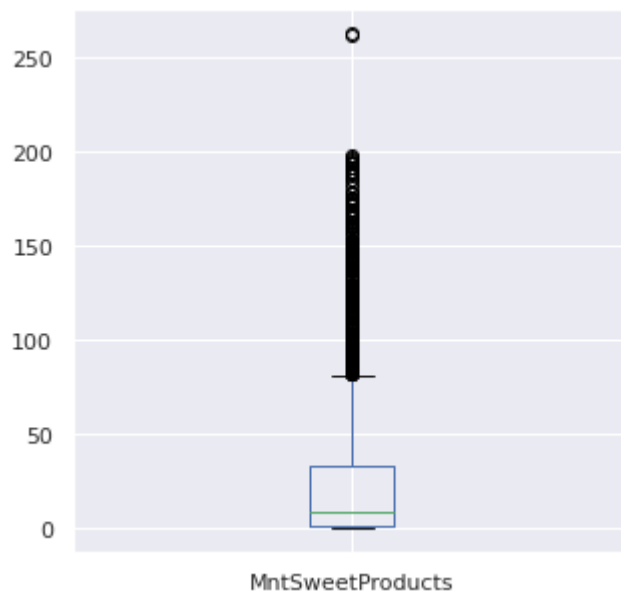
```
In [ ]: data1['MntFishProducts'].std() # Los valores se alejan del promedio en 54.62
```

```
Out[ ]: 54.62897940287769
```

14. Analisis Exploratorio de la Caracteristica **MntSweetProducts**

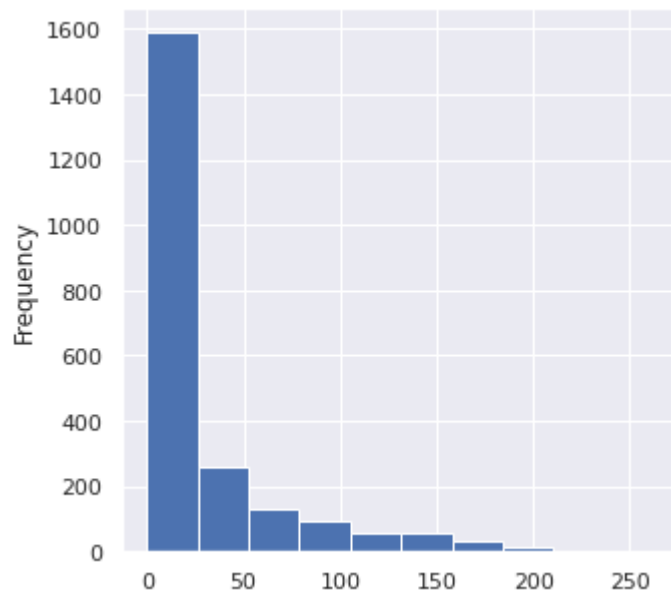
Cantidad gastada en dulces en los últimos 2 años.

```
In [ ]: data1['MntSweetProducts'].plot(kind='box') # se observan valores atipicos  
sns.set(rc={'figure.figsize':(5,5)})  
plt.show()
```



Histograma y Medidas de tendencia central y otros valores de relevancia en el análisis

```
In [ ]: data1['MntSweetProducts'].plot(kind='hist')    # distribucion del tipo positiv
sns.set(rc={'figure.figsize':(5,5)})
plt.show()
```



```
In [ ]: data1['MntSweetProducts'].quantile(0.25)    # indica que un 25% de las persona
s presentan un valor menor o igual a 1 lo que las personas han gastado en dulc
es los ultimos 2 años
```

Out[]: 1.0

```
In [ ]: data1['MntSweetProducts'].quantile(0.5)    # indica que un 50% de las personas
presentan un valor menor o igual a 8 lo que las personas han gastado en dulc
es los ultimos 2 años
```

Out[]: 8.0

```
In [ ]: data1['MntSweetProducts'].quantile(0.75)    # indica que un 75% de las personas
presentan un valor menor o igual a 33 lo que las personas han gastado en dulc
es los ultimos 2 años
```

Out[]: 33.0

```
In [ ]: data1['MntSweetProducts'].min()    # el valor minimo es de 0
```

Out[]: 0

```
In [ ]: data1['MntSweetProducts'].max()    # el valor maximo es de 263
```

Out[]: 263

```
In [ ]: data1['MntSweetProducts'].mode()    # iel valor que mas se repite es de 0
```

Out[]: 0 0
dtype: int64

```
In [ ]: data1['MntSweetProducts'].median() # indica que un 50% de las personas presentan un valor menor o igual a 8
```

```
Out[ ]: 8.0
```

```
In [ ]: data1['MntSweetProducts'].mean() # el promedio de gasto en dulces es de 27.062
```

```
Out[ ]: 27.06294642857143
```

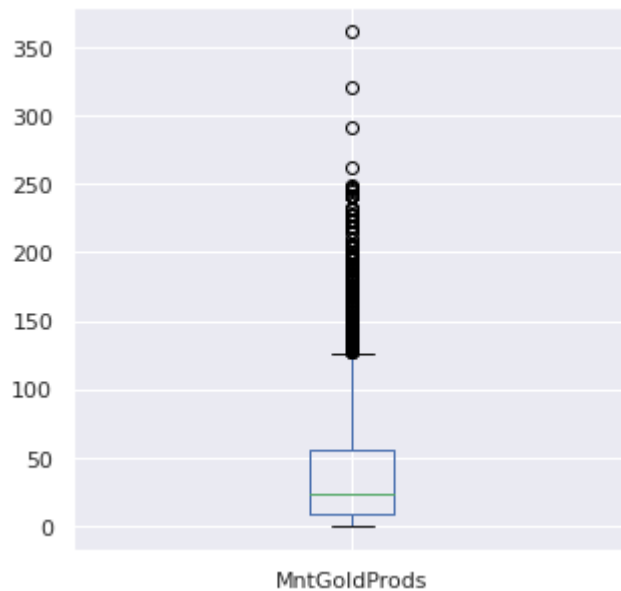
```
In [ ]: data1['MntSweetProducts'].std() # Los valores se desvian en 41.28 del promedio
```

```
Out[ ]: 41.28049848785481
```

15. Analisis Exploratorio de la Caracteristica **MntGoldProds**

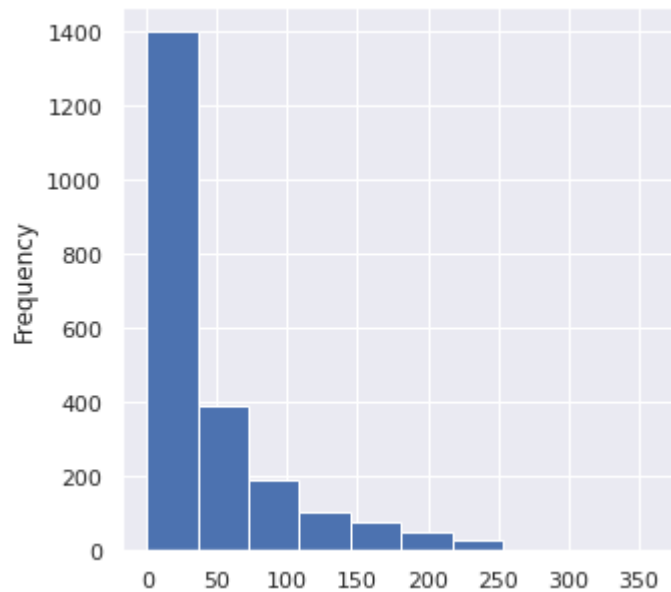
Cantidad gastada en oro en los últimos 2 años.

```
In [ ]: data1['MntGoldProds'].plot(kind='box') # se observan valores atipicos en la caracteristica
sns.set(rc={'figure.figsize':(5,5)})
plt.show()
```



Histograma y Medidas de tendencia central y otros valores de relevancia en el análisis


```
In [ ]: data1['MntGoldProds'].plot(kind='hist')    # distribucion de tipo positiva
sns.set(rc={'figure.figsize':(5,5)})
plt.show()
```



```
In [ ]: data1['MntGoldProds'].quantile(0.25)    # indica que un 25% de las personas pr
esentan un valor menor o igual a 9
```

```
Out[ ]: 9.0
```

```
In [ ]: data1['MntGoldProds'].quantile(0.5)    # indica que un 50% de las personas prese
ntan un valor menor o igual a 24
```

```
Out[ ]: 24.0
```

```
In [ ]: data1['MntGoldProds'].quantile(0.75)    # indica que un 75% de las personas pres
entan un valor menor o igual a 56 Lo que han gastado en oro los ultimos 2 años
```

```
Out[ ]: 56.0
```

```
In [ ]: data1['MntGoldProds'].min()    # valor minimo es de cero
```

```
Out[ ]: 0
```

```
In [ ]: data1['MntGoldProds'].max()    # el valor mas alto es de 362
```

```
Out[ ]: 362
```

```
In [ ]: data1['MntGoldProds'].mode()    # el valor mas repetido es de 1
```

```
Out[ ]: 0    1
dtype: int64
```

```
In [ ]: data1['MntGoldProds'].median() # indica que un 50% de las personas presentan un valor menor o igual a 24
```

```
Out[ ]: 24.0
```

```
In [ ]: data1['MntGoldProds'].mean() # el promedio de lo que las personas gastan en productos de otro es 44.02
```

```
Out[ ]: 44.021875
```

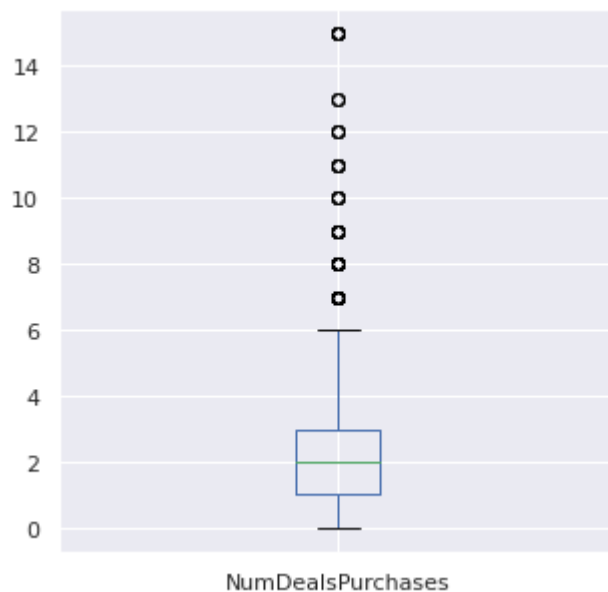
```
In [ ]: data1['MntGoldProds'].std() # los valores se desvian en 52.16 del promedio
```

```
Out[ ]: 52.1674389149972
```

16. Analisis Exploratorio de la Caracteristica **NumDealsPurchases**

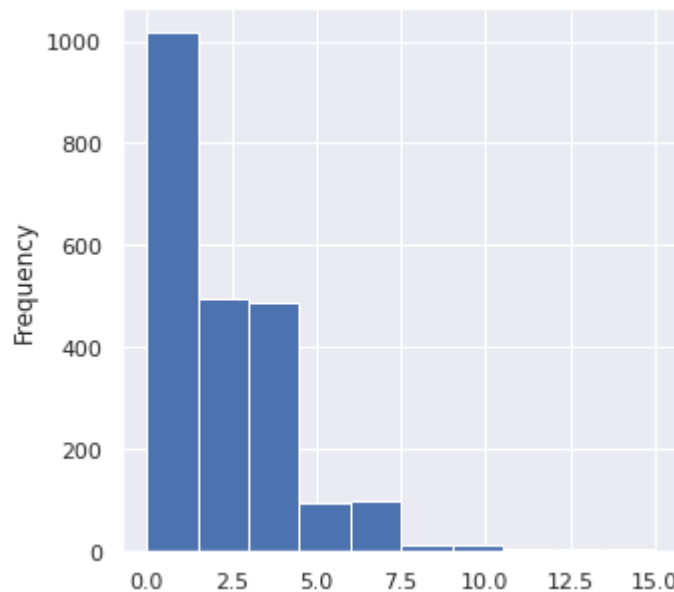
Número de compras realizadas con descuento.

```
In [ ]: data1['NumDealsPurchases'].plot(kind='box')  
sns.set(rc={'figure.figsize':(5,5)})  
plt.show()
```



Histograma y Medidas de tendencia central y otros valores de relevancia en el análisis

```
In [ ]: data1['NumDealsPurchases'].plot(kind='hist')    # distribucion del tipo positi  
va  
sns.set(rc={'figure.figsize':(5,5)})  
plt.show()
```



```
In [ ]: data1['NumDealsPurchases'].quantile(0.25)    # indica que un 25% de las persona  
s presentan un valor menor o igual una compra
```

```
Out[ ]: 1.0
```

```
In [ ]: data1['NumDealsPurchases'].quantile(0.5)    # indica que un 50% de las personas  
presentan un valor menor o igual de 2 compras
```

```
Out[ ]: 2.0
```

```
In [ ]: data1['NumDealsPurchases'].quantile(0.75)    # indica que un 75% de las personas  
presentan un valor menor o igual de 3 compras
```

```
Out[ ]: 3.0
```

```
In [ ]: data1['NumDealsPurchases'].min()    # el valor que mas se repite es 0
```

```
Out[ ]: 0
```

```
In [ ]: data1['NumDealsPurchases'].max()    # el valor maximo es de 15
```

```
Out[ ]: 15
```

```
In [ ]: data1['NumDealsPurchases'].mode()    # el valro que mas se repite es 1
```

```
Out[ ]: 0    1  
dtype: int64
```

```
In [ ]: data1['NumDealsPurchases'].median() # indica que un 50% de las personas presentan un valor menor o igual a 2 la cantidad de compras realizadas con descuento
```

```
Out[ ]: 2.0
```

```
In [ ]: data1['NumDealsPurchases'].mean() # el promedio de las compras realizadas con descuento es de 2.32
```

```
Out[ ]: 2.325
```

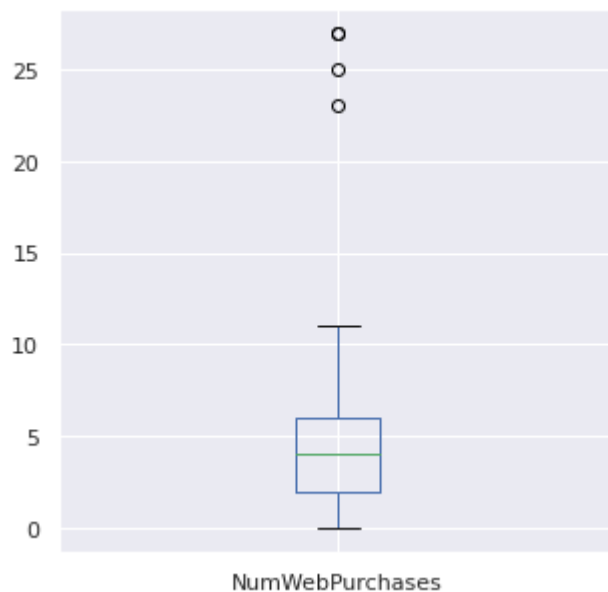
```
In [ ]: data1['NumDealsPurchases'].std() # los valores se desvían en 1.93
```

```
Out[ ]: 1.9322375008559614
```

17. Analisis Exploratorio de la Característica **NumWebPurchases**

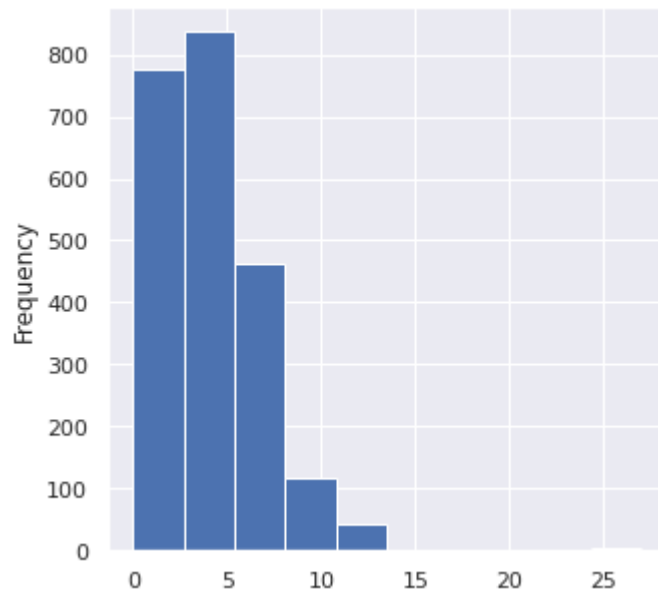
Número de compras realizadas a través del sitio web de la empresa.

```
In [ ]: data1['NumWebPurchases'].plot(kind='box')
sns.set(rc={'figure.figsize':(5,5)})
plt.show()
```



Histograma y Medidas de tendencia central y otros valores de relevancia en el análisis

```
In [ ]: data1['NumWebPurchases'].plot(kind='hist') # distribucion del tipo positiva
sns.set(rc={'figure.figsize':(5,5)})
plt.show()
```



```
In [ ]: data1['NumWebPurchases'].quantile(0.25) # indica que un 25% de las personas
presentan un valor menor o igual a 2 las compras por la web
```

```
Out[ ]: 2.0
```

```
In [ ]: data1['NumWebPurchases'].quantile(0.5) # indica que un 50% de las personas pr
esentan un valor menor o igual a 4 las compras por la web
```

```
Out[ ]: 4.0
```

```
In [ ]: data1['NumWebPurchases'].quantile(0.75) # indica que un 75% de las personas
presentan un valor menor o igual a 6 las compras por la web
```

```
Out[ ]: 6.0
```

```
In [ ]: data1['NumWebPurchases'].min() # el valor minimo es de cero
```

```
Out[ ]: 0
```

```
In [ ]: data1['NumWebPurchases'].max() # el valor maximo de compras por la web es de
27
```

```
Out[ ]: 27
```

```
In [ ]: data1['NumWebPurchases'].mode() # el valor que mas se repite es 2
```

```
Out[ ]: 0    2
dtype: int64
```

```
In [ ]: data1['NumWebPurchases'].median() # indica que un 50% de las personas presen  
tan un valor menor o igual a 4
```

```
Out[ ]: 4.0
```

```
In [ ]: data1['NumWebPurchases'].mean() # el valor promedio de compras realizadas por  
la web es de 4
```

```
Out[ ]: 4.084821428571429
```

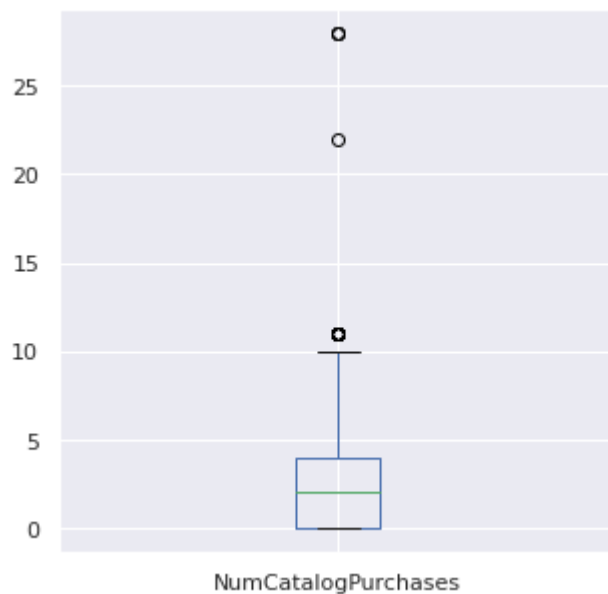
```
In [ ]: data1['NumWebPurchases'].std() # Los valores se alejan del promedio en 2.77
```

```
Out[ ]: 2.7787141473881087
```

18. Analisis Exploratorio de la Caracteristica **NumCatalogPurchases**

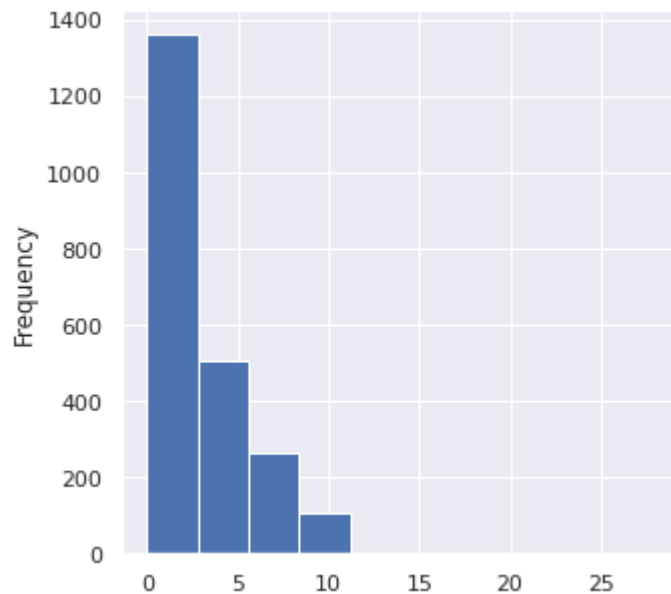
Número de compras realizadas mediante catálogo.

```
In [ ]: data1['NumCatalogPurchases'].plot(kind='box')  
sns.set(rc={'figure.figsize':(5,5)})  
plt.show()
```



Histograma y Medidas de tendencia central y otros valores de relevancia en el análisis

```
In [ ]: data1['NumCatalogPurchases'].plot(kind='hist')    #la distribucion es de tipo
        sns.set(rc={'figure.figsize':(5,5)})              #positiva, concentracion de los datos hacia la izquierda del grafico
        plt.show()
```



```
In [ ]: data1['NumCatalogPurchases'].quantile(0.25)    # indica que un 25% de las perso
        #nas presentan un valor menor o igual a 0 compras por catalogo
```

```
Out[ ]: 0.0
```

```
In [ ]: data1['NumCatalogPurchases'].quantile(0.5)     # indica que un 50% de las person
        #as presentan un valor menor o igual a 2 de compras por catalogo
```

```
Out[ ]: 2.0
```

```
In [ ]: data1['NumCatalogPurchases'].quantile(0.75)    # indica que un 75% de las perso
        #nas presentan un valor menor o igual a 4 de compras por catalogo
```

```
Out[ ]: 4.0
```

```
In [ ]: data1['NumCatalogPurchases'].min()             # el valor minimo es cero
```

```
Out[ ]: 0
```

```
In [ ]: data1['NumCatalogPurchases'].max()             # el valor maximo es de 28
```

```
Out[ ]: 28
```

```
In [ ]: data1['NumCatalogPurchases'].mode()           # el valor que mas se repite es cero
```

```
Out[ ]: 0      0
        dtype: int64
```

```
In [ ]: data1['NumCatalogPurchases'].median() # indica que un 50% de las personas p  
resentan un valor menor o igual a 2
```

```
Out[ ]: 2.0
```

```
In [ ]: data1['NumCatalogPurchases'].mean() # el valor promedio de compras por catalog  
o corresponde a 2.66
```

```
Out[ ]: 2.6620535714285714
```

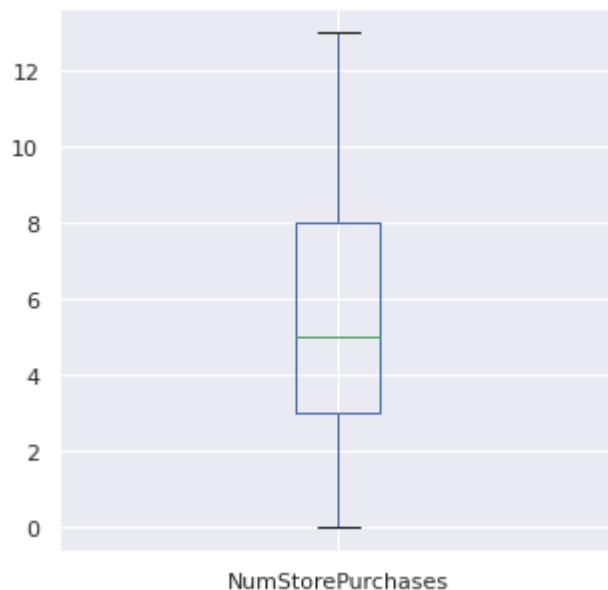
```
In [ ]: data1['NumCatalogPurchases'].std() # los valores se alejan del promedio en  
2.92
```

```
Out[ ]: 2.9231006555397197
```

19. Analisis Exploratorio de la Caracteristica **NumStorePurchases**

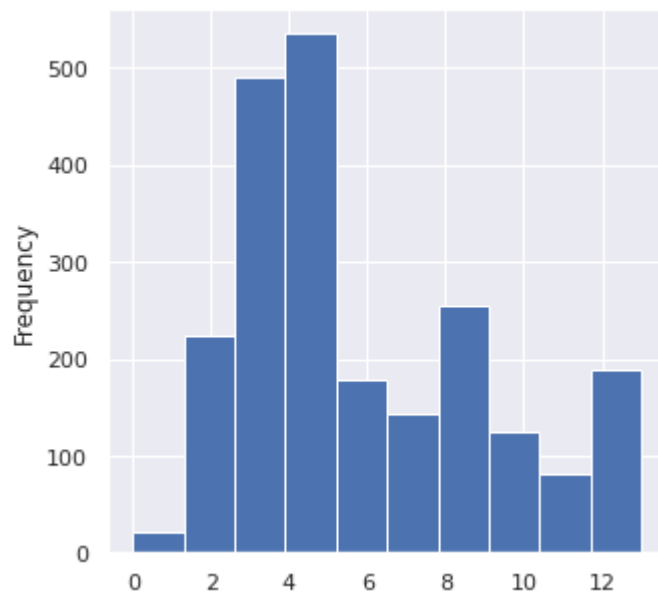
Número de compras realizadas directamente en tiendas.

```
In [ ]: data1['NumStorePurchases'].plot(kind='box') # no se observan valores atip  
icos, el valor minimo es cero, valor maximo 13. La mediana corresponde a 5  
sns.set(rc={'figure.figsize':(5,5)})  
plt.show()
```



Histograma y Medidas de tendencia central y otros valores de relevancia en el análisis


```
In [ ]: data1['NumStorePurchases'].plot(kind='hist')
sns.set(rc={'figure.figsize':(5,5)})
plt.show()
```



```
In [ ]: data1['NumStorePurchases'].quantile(0.25)  # indica que un 25% de las personas
s presentan un valor menor o igual a 3
```

```
Out[ ]: 3.0
```

```
In [ ]: data1['NumStorePurchases'].quantile(0.5)  # indica que un 50% de las personas
presentan un valor menor o igual a 5
```

```
Out[ ]: 5.0
```

```
In [ ]: data1['NumStorePurchases'].quantile(0.75)  # indica que un 75% de las personas
s presentan un valor menor o igual a 8
```

```
Out[ ]: 8.0
```

```
In [ ]: data1['NumStorePurchases'].min()  # el valor minimo es cero
```

```
Out[ ]: 0
```

```
In [ ]: data1['NumStorePurchases'].max()  # el valor maximo es 13
```

```
Out[ ]: 13
```

```
In [ ]: data1['NumStorePurchases'].mode()  # el valor que mas se repite es 3
```

```
Out[ ]: 0    3
dtype: int64
```

```
In [ ]: data1['NumStorePurchases'].median() # indica que un 50% de las personas pres  
entan un valor menor o igual a 5
```

```
Out[ ]: 5.0
```

```
In [ ]: data1['NumStorePurchases'].mean() # el valor promedio de compras realizadas e  
n las tiendas es de 5.79
```

```
Out[ ]: 5.790178571428571
```

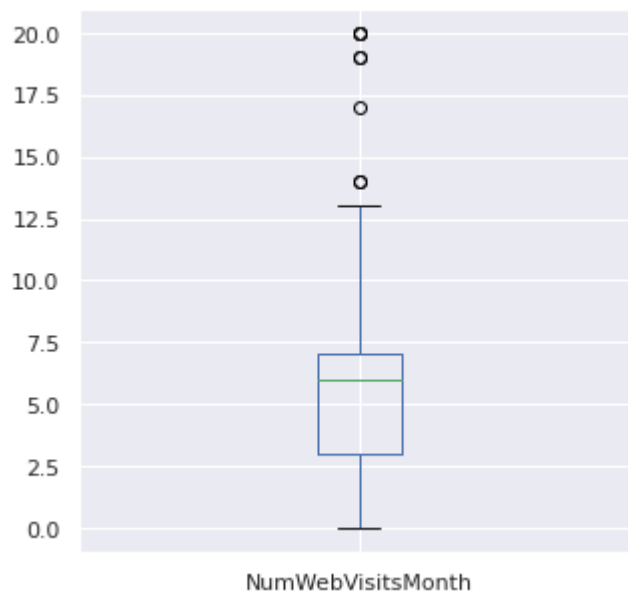
```
In [ ]: data1['NumStorePurchases'].std() # los valores se alejan del promedio en 3.2  
5
```

```
Out[ ]: 3.250958145674417
```

20. Analisis Exploratorio de la Caracteristica **NumWebVisitsMonth**

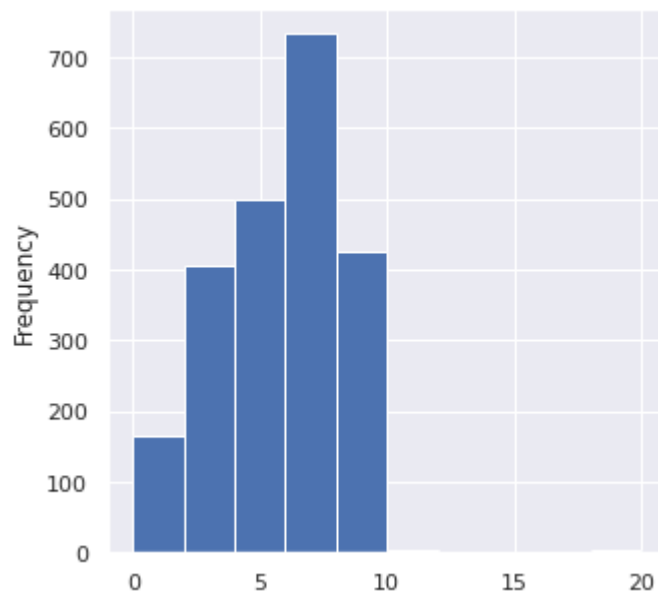
Número de visitas al sitio web de la empresa en el último mes.

```
In [ ]: data1['NumWebVisitsMonth'].plot(kind='box')  
sns.set(rc={'figure.figsize':(5,5)})  
plt.show()
```



Histograma y Medidas de tendencia central y otros valores de relevancia en el análisis

```
In [ ]: data1['NumWebVisitsMonth'].plot(kind='hist')
sns.set(rc={'figure.figsize':(5,5)})
plt.show()
```



```
In [ ]: data1['NumWebVisitsMonth'].quantile(0.25)    # indica que un 25% de las person
as presentan un valor menor o igual a 3
```

```
Out[ ]: 3.0
```

```
In [ ]: data1['NumWebVisitsMonth'].quantile(0.5)    # indica que un 50% de las personas
presentan un valor menor o igual a 6
```

```
Out[ ]: 6.0
```

```
In [ ]: data1['NumWebVisitsMonth'].quantile(0.75)    # indica que un 75% de las persona
s presentan un valor menor o igual a 7
```

```
Out[ ]: 7.0
```

```
In [ ]: data1['NumWebVisitsMonth'].min()    # valor minimo es cero
```

```
Out[ ]: 0
```

```
In [ ]: data1['NumWebVisitsMonth'].max()    # el valor maximo es 20
```

```
Out[ ]: 20
```

```
In [ ]: data1['NumWebVisitsMonth'].mode()    # el valor que mas se repite es 7
```

```
Out[ ]: 0    7
dtype: int64
```

```
In [ ]: data1['NumWebVisitsMonth'].median() # indica que un 50% de las personas pres  
entan un valor menor o igual de 6 visitas a la pagina web
```

```
Out[ ]: 6.0
```

```
In [ ]: data1['NumWebVisitsMonth'].mean() # el valor promedio de visitas a la pagina  
web es de 5.31
```

```
Out[ ]: 5.316517857142857
```

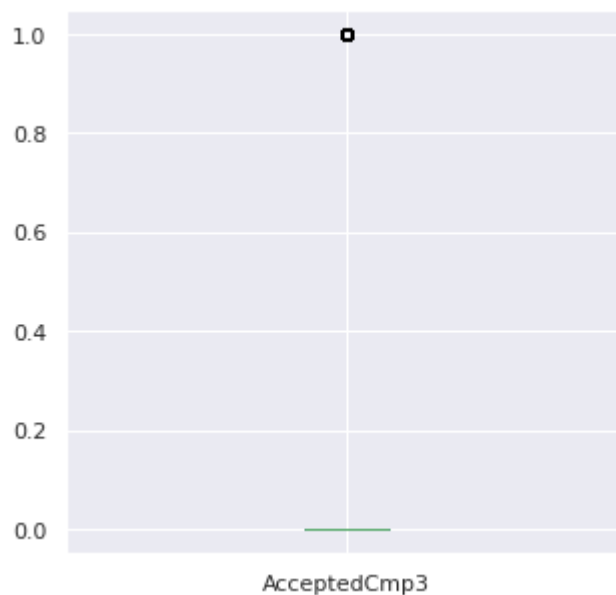
```
In [ ]: data1['NumWebVisitsMonth'].std() # Los valores se alejan del promedio en 2.4  
2
```

```
Out[ ]: 2.426645009547285
```

21. Analisis Exploratorio de la Caracteristica **AcceptedCmp3**

1 si el cliente aceptó la oferta en la 3ª campaña, 0 en caso contrario.

```
In [ ]: data1['AcceptedCmp3'].plot(kind='box') # caracteristica del tipo binaria, no  
se aprovecha tanto para la generacion de graficas,  
sns.set(rc={'figure.figsize':(5,5)}) #se procede más adelante a utilizar e  
n caso de ser util  
plt.show()
```



```
In [ ]: data1['AcceptedCmp3'].quantile(0.25)
```

```
Out[ ]: 0.0
```

```
In [ ]: data1['AcceptedCmp3'].quantile(0.5)
```

```
Out[ ]: 0.0
```

```
In [ ]: data1['AcceptedCmp3'].quantile(0.75)
```

```
Out[ ]: 0.0
```

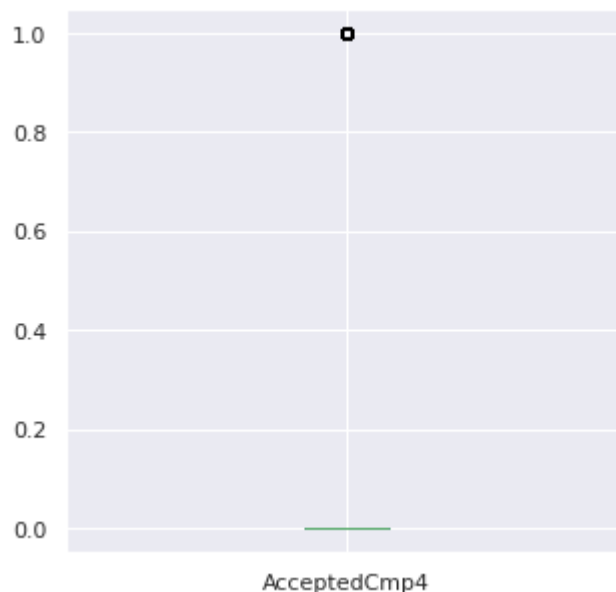
```
In [ ]: data1['AcceptedCmp3'].mode()
```

```
Out[ ]: 0    0
dtype: int64
```

22. Analisis Exploratorio de la Característica **AcceptedCmp4**

1 si el cliente aceptó la oferta en la cuarta campaña, 0 en caso contrario.

```
In [ ]: data1['AcceptedCmp4'].plot(kind='box') # característica del tipo binaria, no
sns.set(rc={'figure.figsize':(5,5)})          # se aprovecha tanto para la generacion de graficas,
plt.show()                                     # se procede más adelante a utilizar e
n caso de ser util
```



```
In [ ]: data1['AcceptedCmp4'].mode()
```

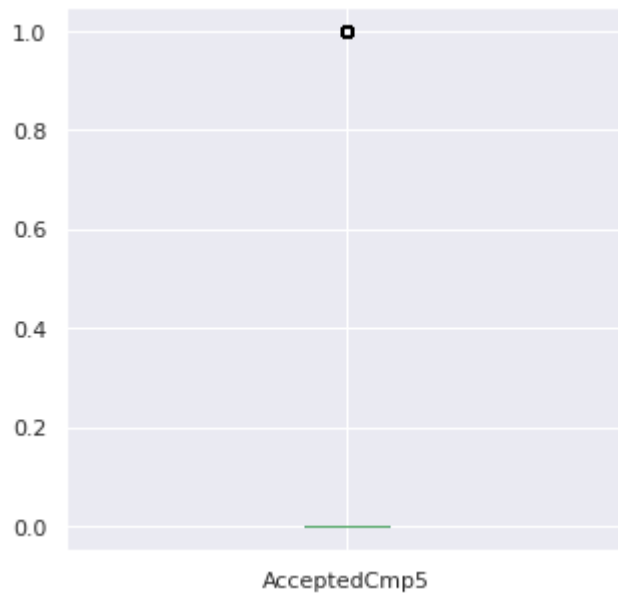
```
Out[ ]: 0    0
dtype: int64
```

Histograma y Medidas de tendencia central y otros valores de relevancia en el análisis

23. Analisis Exploratorio de la Característica **AcceptedCmp5**

1 si el cliente aceptó la oferta en la quinta campaña, 0 en caso contrario.

```
In [ ]: data1['AcceptedCmp5'].plot(kind='box') # característica del tipo binaria, no
se aprovecha tanto para la generacion de graficas,
sns.set(rc={'figure.figsize':(5,5)}) #se procede más adelante a utilizar e
n caso de ser util
plt.show()
```



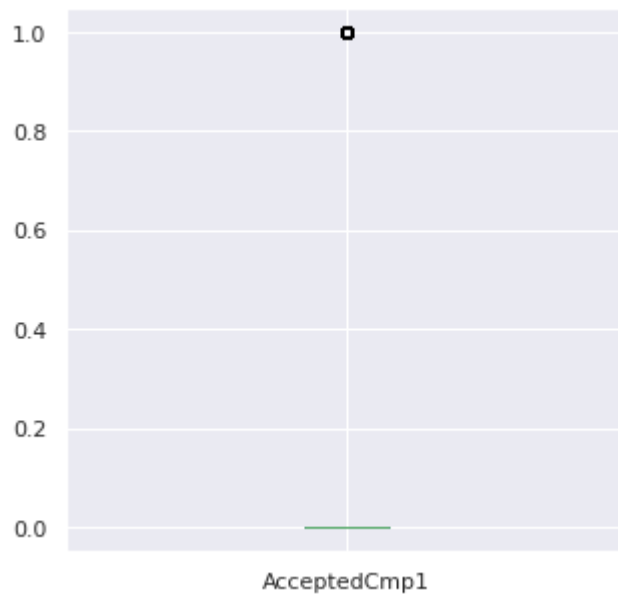
```
In [ ]: data1['AcceptedCmp5'].mode()
```

```
Out[ ]: 0    0
dtype: int64
```

24. Analisis Exploratorio de la Caracteristica **AcceptedCmp1**

1 si el cliente aceptó la oferta en la 1ª campaña, 0 en caso contrario.

```
In [ ]: data1['AcceptedCmp1'].plot(kind='box') # característica del tipo binaria, no
se aprovecha tanto para la generación de graficas,
sns.set(rc={'figure.figsize':(5,5)}) #se procede más adelante a utilizar e
n caso de ser útil
plt.show()
```



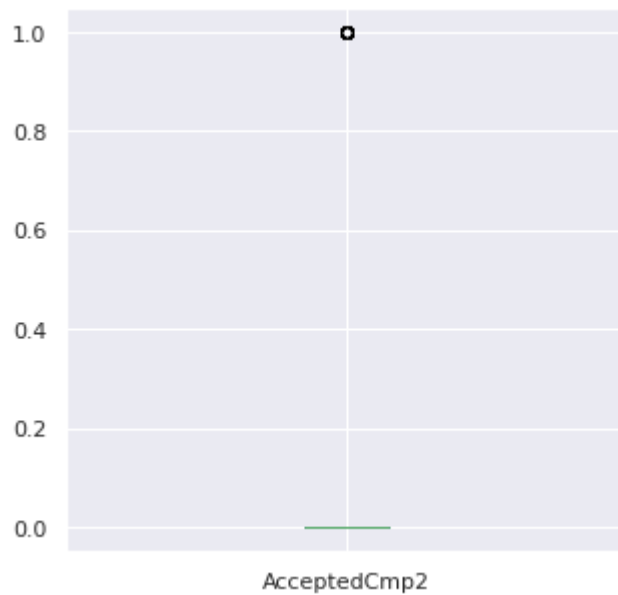
```
In [ ]: data1['AcceptedCmp1'].mode()
```

```
Out[ ]: 0    0
dtype: int64
```

25. Analisis Exploratorio de la Característica **AcceptedCmp2**

1 si el cliente aceptó la oferta en la segunda campaña, 0 en caso contrario.

```
In [ ]: data1['AcceptedCmp2'].plot(kind='box') # característica del tipo binaria, no
se aprovecha tanto para la generación de graficas,
sns.set(rc={'figure.figsize':(5,5)}) #se procede más adelante a utilizar e
n caso de ser útil
plt.show()
```



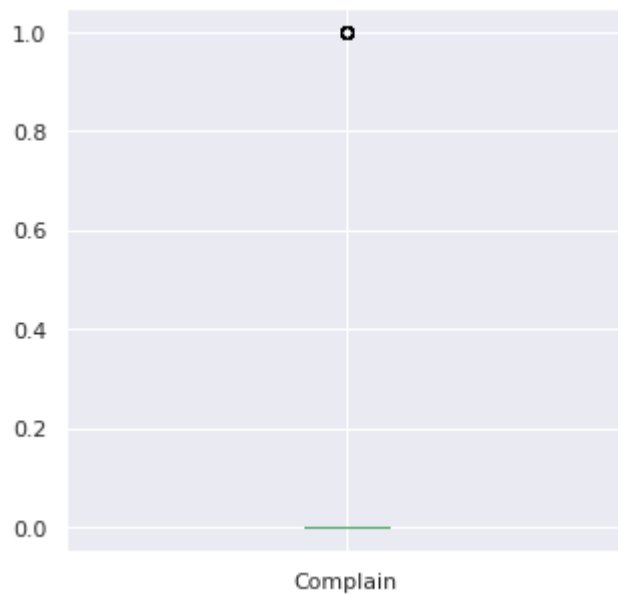
```
In [ ]: data1['AcceptedCmp2'].mode()
```

```
Out[ ]: 0    0
dtype: int64
```

26. Analisis Exploratorio de la Característica **Complain**

1 si el cliente se quejó en los últimos 2 años, 0 en caso contrario.


```
In [ ]: data1['Complain'].plot(kind='box')    # característica del tipo binaria, no se
aprovecha tanto para la generación de graficas,
sns.set(rc={'figure.figsize':(5,5)}) #se procede más adelante a utilizar en c
aso de ser util
plt.show()
```



Histograma y Medidas de tendencia central y otros valores de relevancia en el análisis

```
In [ ]: data1['Complain'].mode()
```

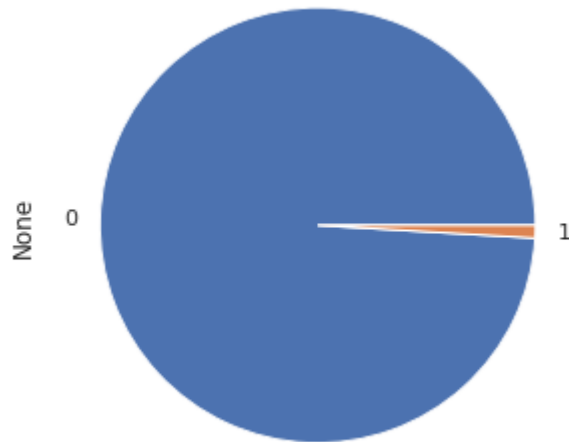
```
Out[ ]: 0    0
dtype: int64
```

```
In [ ]: data1.groupby('Complain').size()
```

```
Out[ ]: Complain
0      2219
1         21
dtype: int64
```

```
In [ ]: data1.groupby('Complain').size().plot(kind = 'pie')    # se observa que La ma  
        yoria de los clientes no se quejan
```

```
Out[ ]: <matplotlib.axes._subplots.AxesSubplot at 0x7faaa4abde90>
```



27. Analisis Exploratorio de la Caracteristica **Z_CostContact**

Un unico valor en toda la caracteristica

```
In [ ]: data1['Z_CostContact'].head(5)
```

```
Out[ ]: 0      3  
        1      3  
        2      3  
        3      3  
        4      3  
        Name: Z_CostContact, dtype: int64
```

```
In [ ]: data1.groupby('Z_CostContact').size()
```

```
Out[ ]: Z_CostContact  
        3      2240  
        dtype: int64
```

```
In [ ]: data1['Z_CostContact'].nunique()
```

```
Out[ ]: 1
```

28. Analisis Exploratorio de la Caracteristica **Z_Revenue**

Un unico valor en toda la caracteristica

```
In [ ]: data1['Z_Revenue'].mode()
```

```
Out[ ]: 0    11
dtype: int64
```

```
In [ ]: data1.groupby('Z_Revenue').size()
```

```
Out[ ]: Z_Revenue
11    2240
dtype: int64
```

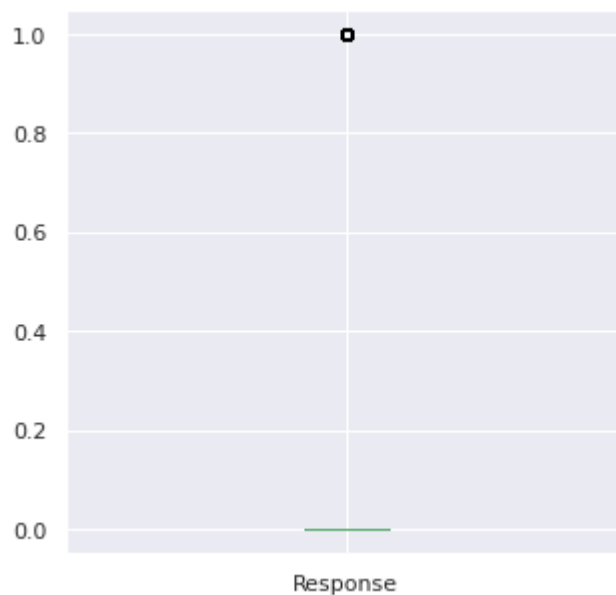
```
In [ ]: data1['Z_Revenue'].nunique()
```

```
Out[ ]: 1
```

29. Analisis Exploratorio de la Característica **Response**

1 if customer accepted the offer in the last campaign, 0 otherwise.

```
In [ ]: data1['Response'].plot(kind='box')    # característica del tipo binaria, no se
aprovecha tanto para la generacion de graficas,
sns.set(rc={'figure.figsize':(5,5)})    #se procede más adelante a utilizar en c
aso de ser util
plt.show()
```



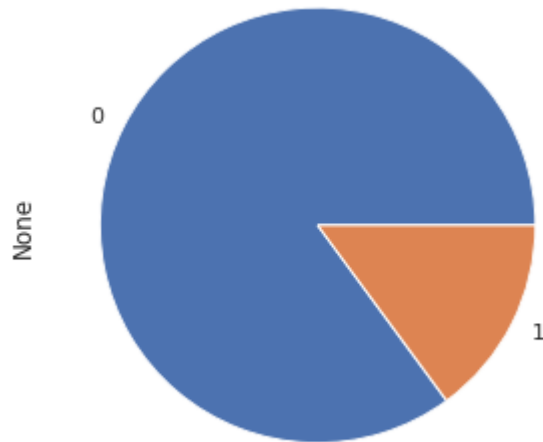
```
In [ ]: data1['Response'].mode()    # La mayoría de los clientes no responden a la cam
paña
```

```
Out[ ]: 0    0
dtype: int64
```

```
In [ ]: data1.groupby('Response').size()
```

```
Out[ ]: Response
0      1906
1       334
dtype: int64
```

```
In [ ]: data1.groupby('Response').size().plot(kind = 'pie')
sns.set(rc={'figure.figsize':(5,5)})
plt.show()
```



Tablas de Analisis

Encontrará tablas, que resumen información por características, permiten ver la segmentación de los clientes por Edad, Ingresos, Comportamiento de Compra, Además de extraer información importante para perfilar el comportamiento de los clientes.

Transformaciones de datos Necesarias

1. Crear columnas que agreguen valor al proceso de análisis de los datos
2. Transformar o cambiar columnas tipo objeto a numérico para facilitar la manipulación de los datos
3. Crear un rangos o escalas en las características que permita utilizarlas en las tablas pivote

```
In [ ]: data2 = data1.copy() # con data2 vamos a proceder a proceder a convertir Las
características objeto en numericas
```

```
In [ ]: # Hay que transformar a numéricas las variables categóricas (object) para poder trabajar con ellas
data2.dtypes
```

```
Out[ ]: ID                                int64
Year_Birth                             int64
Education                             object
Marital_Status                         object
Income                                float64
Kidhome                               int64
Teenhome                              int64
Dt_Customer                           datetime64[ns]
Recency                               int64
MntWines                              int64
MntFruits                             int64
MntMeatProducts                       int64
MntFishProducts                       int64
MntSweetProducts                      int64
MntGoldProds                          int64
NumDealsPurchases                     int64
NumWebPurchases                       int64
NumCatalogPurchases                  int64
NumStorePurchases                     int64
NumWebVisitsMonth                     int64
AcceptedCmp3                          int64
AcceptedCmp4                          int64
AcceptedCmp5                          int64
AcceptedCmp1                          int64
AcceptedCmp2                          int64
Complain                              int64
Z_CostContact                         int64
Z_Revenue                             int64
Response                              int64
dtype: object
```

```
In [ ]: obj_df = data2.select_dtypes(include=['object']).copy() # se eligen las variables categóricas (object) y se hace una copia
print(obj_df.columns)
```

```
Index(['Education', 'Marital_Status'], dtype='object')
```

```
In [ ]: from sklearn.preprocessing import LabelEncoder
```

```
In [ ]: lb_encoder = LabelEncoder()
```

```
In [ ]: #crear un ciclo for para que a cada una de las variables que tengan formato 'object' las cambie a formato numero
for col in obj_df.columns:
    data2[col] = lb_encoder.fit_transform(data2[col])
```

```
In [ ]: # Comprobando de nuevo tipo de las variables
data2.dtypes
```

```
Out[ ]: ID                                int64
Year_Birth                             int64
Education                             int64
Marital_Status                         int64
Income                                float64
Kidhome                               int64
Teenhome                              int64
Dt_Customer                           datetime64[ns]
Recency                               int64
MntWines                              int64
MntFruits                             int64
MntMeatProducts                       int64
MntFishProducts                       int64
MntSweetProducts                      int64
MntGoldProds                          int64
NumDealsPurchases                     int64
NumWebPurchases                       int64
NumCatalogPurchases                  int64
NumStorePurchases                     int64
NumWebVisitsMonth                     int64
AcceptedCmp3                          int64
AcceptedCmp4                          int64
AcceptedCmp5                          int64
AcceptedCmp1                          int64
AcceptedCmp2                          int64
Complain                              int64
Z_CostContact                          int64
Z_Revenue                             int64
Response                              int64
dtype: object
```

Crear una columna para la edad del cliente

```
In [ ]: data1['Age'] = 2014 - data1['Year_Birth'] # La fecha 2014 se elige porque co
rresponde a la fecha en donde al parecer se realizó el estudio o de la fecha
# en la que proviene el dataset.
Age sería la edad de los clientes a la fecha 2014
```

```
In [ ]: data1.columns
```

```
Out[ ]: Index(['ID', 'Year_Birth', 'Education', 'Marital_Status', 'Income', 'Kidhom
e',
               'Teenhome', 'Dt_Customer', 'Recency', 'MntWines', 'MntFruits',
               'MntMeatProducts', 'MntFishProducts', 'MntSweetProducts',
               'MntGoldProds', 'NumDealsPurchases', 'NumWebPurchases',
               'NumCatalogPurchases', 'NumStorePurchases', 'NumWebVisitsMonth',
               'AcceptedCmp3', 'AcceptedCmp4', 'AcceptedCmp5', 'AcceptedCmp1',
               'AcceptedCmp2', 'Complain', 'Z_CostContact', 'Z_Revenue', 'Response',
               'Age'],
              dtype='object')
```

```
In [ ]: data1['Age'].head(5)
```

```
Out[ ]: 0    57
        1    60
        2    49
        3    30
        4    33
        Name: Age, dtype: int64
```

```
In [ ]: data1['Age'].min() # La edad minima es de 18 años
```

```
Out[ ]: 18
```

```
In [ ]: data1['Age'].max() # La edad maxima es de 121 años
```

```
Out[ ]: 121
```

```
In [ ]: data1['Age'].mode() # La edad que más se repite es 38 años
```

```
Out[ ]: 0    38
        dtype: int64
```

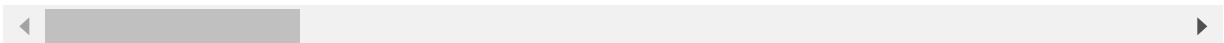
```
In [ ]: data1['Age'].mean() # La edad promedio es de 45 años
```

```
Out[ ]: 45.19419642857143
```

```
In [ ]: data1.head(3)
```

```
Out[ ]:
```

	ID	Year_Birth	Education	Marital_Status	Income	Kidhome	Teenhome	Dt_Customer	Rec
0	5524	1957	Graduation	Single	58138.0	0	0	2012-04-09	
1	2174	1954	Graduation	Single	46344.0	1	1	2014-08-03	
2	4141	1965	Graduation	Together	71613.0	0	0	2013-08-21	



Mostar Información por Edades y Rangos de Edades

Una tabla que muestre por edades el nivel de ingreso de los clientes

```
In [ ]: round(data1.pivot_table('Income', index= 'Age', aggfunc= 'mean',fill_value=0),  
2)
```


Out[]:

Income	
Age	
18	10960.50
19	60937.00
20	85449.33
21	74139.80
22	47830.15
23	60900.33
24	40919.28
25	42554.55
26	47947.76
27	46888.67
28	43537.73
29	39192.94
30	39132.79
31	48076.99
32	52654.86
33	47009.96
34	49927.03
35	44956.53
36	46082.16
37	62653.04
38	47539.21
39	52417.13
40	48338.90
41	47332.20
42	51126.60
43	49378.20
44	52369.62
45	51208.11
46	48597.45
47	53782.43
48	50960.46
49	55668.66
50	56352.65
51	48872.88

Income	
Age	
52	60988.95
53	57000.51
54	54817.63
55	56227.17
56	58467.44
57	53897.53
58	56706.60
59	57484.23
60	58483.45
61	57810.00
62	56903.67
63	55892.45
64	50197.17
65	61547.20
66	65405.05
67	66327.56
68	59944.25
69	70375.38
70	66477.14
71	60760.93
73	93027.00
74	51141.00
114	36640.00
115	83532.00
121	60182.00

```
In [ ]: plt.figure(figsize = (20,20))
round(data1.pivot_table('Income', index= 'Age', aggfunc= 'mean',fill_value=0),
2).plot()
plt.title("Nivel de Ingreso Por Edad Clientes")
plt.ylabel("Ingreso")
plt.xlabel("Por Años")
plt.show();
```

<Figure size 1440x1440 with 0 Axes>



Tabla Pivote por Rangos de Edad

Una tabla que permita ver los Ingresos por Rangos de Edad

```
In [ ]: data1['age_range'] = (data1['Age']//5)*5      # se crea una columna qu correspo
nde a rango de edad
```

```
In [ ]: round(data1.pivot_table(['Income'], ['age_range'], aggfunc= 'mean', fill_value=0),2)
```

Out[]:

Income	
age_range	
15	46658.00
20	53683.17
25	43849.21
30	47567.93
35	50199.41
40	49754.75
45	52235.09
50	55498.55
55	56651.57
60	56294.77
65	63772.24
70	64677.16
110	36640.00
115	83532.00
120	60182.00

Tabla Pivote.

Que muestre Información sobre la Edad, Ingreso y además si existe una diferencia entre aquellos que responden o no a las campañas publicitarias

En promedio por rangos de edad aquellos de mayor edad tienen mayor ingreso, y aquellos que dan respuesta positiva a la campaña tienen un mayor ingreso que los que no.

```
In [ ]: round(data1.pivot_table(['Income'], ['age_range'], ['Response'], aggfunc= 'mean',
fill_value=0),2)    # Response 1 indica si el cliente aceptó la oferta en la
ultima campaña, 0 que no.
```

Out[]:

	Income	
	Response 0	1
age_range		
15	47169.80	45378.50
20	45447.38	82508.42
25	41942.80	55807.64
30	44090.53	59777.49
35	49649.64	54639.90
40	48883.83	53977.36
45	51511.21	57509.11
50	54076.08	65049.43
55	54758.11	68336.91
60	55579.22	61810.50
65	62803.75	66809.77
70	67080.67	57466.62
110	36640.00	0.00
115	83532.00	0.00
120	60182.00	0.00

```
In [ ]: round(data1.pivot_table(['Income'], ['age_range'], ['Response'], aggfunc= 'mean',  
fill_value=0),2).plot(kind = 'bar' )  
sns.set(rc={'figure.figsize':(7,7)})  
plt.show()
```



```
In [ ]: round(data1.pivot_table(['Income'], ['age_range'], ['Education'], aggfunc= 'mean', fill_value=0), 2)
```

Out[]:

	Income				
Education	2n Cycle	Basic	Graduation	Master	PhD
age_range					
15	57124.67	14421.00	46937.00	0.00	0.00
20	48948.80	18012.43	55769.06	81595.40	69926.00
25	32391.86	17403.20	46690.32	45802.89	53447.83
30	48668.90	25161.50	46197.49	41956.15	58285.64
35	39110.54	20756.83	53765.49	48525.91	55874.16
40	43827.58	18949.17	49222.01	52869.23	52596.16
45	59966.14	22796.75	52631.61	49687.66	53684.20
50	59967.78	23384.75	55747.72	58262.46	54169.41
55	58727.09	15056.00	56723.76	56986.33	56075.75
60	58877.89	20040.50	56744.48	53599.18	58446.58
65	77951.50	28389.00	62447.84	64005.14	64611.38
70	0.00	0.00	67273.33	63848.50	64221.50
110	36640.00	0.00	0.00	0.00	0.00
115	0.00	0.00	0.00	0.00	83532.00
120	60182.00	0.00	0.00	0.00	0.00

```
In [ ]: compras_promedio1 = data1[['MntWines', 'MntFruits', 'MntMeatProducts', 'MntFishProducts', 'MntSweetProducts', 'MntGoldProds', 'age_range']].\
groupby("age_range").mean().sort_values(by="age_range").reset_index()
```

```
In [ ]: compras_promedio1.head(12)
```

```
Out[ ]:
```

	age_range	MntWines	MntFruits	MntMeatProducts	MntFishProducts	MntSweetProducts	M
0	15	203.428571	16.714286	233.000000	38.571429	41.857143	
1	20	349.907407	41.537037	298.537037	55.092593	34.592593	
2	25	228.656250	27.631250	173.037500	35.118750	29.875000	
3	30	240.384236	26.369458	160.068966	38.423645	25.990148	
4	35	244.310734	24.121469	141.573446	35.641243	25.598870	
5	40	268.948187	22.673575	148.500000	31.349741	24.626943	
6	45	306.375862	24.893103	150.658621	32.182759	24.351724	
7	50	372.620370	28.194444	163.643519	38.851852	26.310185	
8	55	356.326693	29.011952	185.063745	42.613546	29.095618	
9	60	344.602871	23.516746	166.330144	39.215311	26.693780	
10	65	476.967033	34.032967	252.637363	49.978022	39.362637	
11	70	614.812500	46.187500	298.812500	87.000000	34.937500	

Tabla Pivote.

Que muestre Información por rangos de Edad, Ingresos y ademas según aquellos que se han quejado o no en los últimos 2 años.


```
In [ ]: round(data1.pivot_table('MntWines', index= 'age_range', columns= 'Complain', aggfunc= 'mean', fill_value=0),2)
```

```
# Clientes que se han quejado y los que no , como es el comportamiento de compra de vinos por rango de edad
```

Out[]:

	Complain	0	1
age_range			
15	148.00	536.00	
20	349.91	0.00	
25	229.99	16.00	
30	243.78	13.67	
35	245.97	128.40	
40	268.95	0.00	
45	306.38	0.00	
50	374.24	24.00	
55	357.29	297.00	
60	349.09	114.50	
65	476.97	0.00	
70	613.87	629.00	
110	0.00	15.00	
115	755.00	0.00	
120	8.00	0.00	

Tablas Pivote Según El Comportamiento de Compra de los Clientes

según diferentes variables o condiciones, por edades, ingresos o educación.

```
In [ ]: plt.figure(figsize = (20,20))
round(data1.pivot_table('MntWines', index= 'age_range', columns= 'Complain', a
ggfunc= 'mean',fill_value=0),2).plot()
plt.title("Comportamiento de Compra de Clientes")
plt.ylabel("Compra Promedio Vinos")
plt.xlabel("Por edades")
plt.show();
```

<Figure size 1440x1440 with 0 Axes>

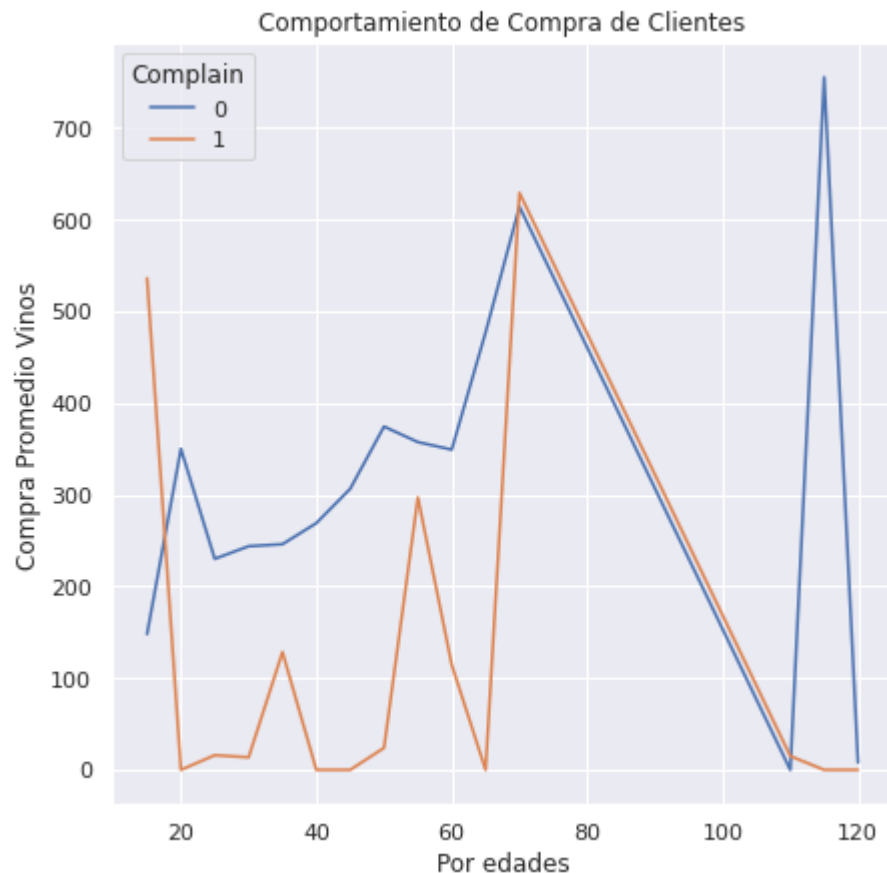


Tabla Pivote para Analizar el Comportamiento de Consumo por Productos

```
In [ ]: data1['Year'] = data1['Dt_Customer'].dt.year
```

```
In [ ]: data1.columns
```

```
Out[ ]: Index(['ID', 'Year_Birth', 'Education', 'Marital_Status', 'Income', 'Kidhome',
              'Teenhome', 'Dt_Customer', 'Recency', 'MntWines', 'MntFruits',
              'MntMeatProducts', 'MntFishProducts', 'MntSweetProducts',
              'MntGoldProds', 'NumDealsPurchases', 'NumWebPurchases',
              'NumCatalogPurchases', 'NumStorePurchases', 'NumWebVisitsMonth',
              'AcceptedCmp3', 'AcceptedCmp4', 'AcceptedCmp5', 'AcceptedCmp1',
              'AcceptedCmp2', 'Complain', 'Z_CostContact', 'Z_Revenue', 'Response',
              'Age', 'age_range', 'Year'],
              dtype='object')
```

```
In [ ]: compras_promedio = data1[['MntWines', 'MntFruits', 'MntMeatProducts', 'MntFishProducts', 'MntSweetProducts', 'MntGoldProds', 'Year']].\
groupby("Year").mean().sort_values(by="Year").reset_index()
```

Tabla Pivote que muestra por el año de ligamen del cliente con la empresa, y como se observa el promedio de consumo de los productos.

```
In [ ]: compras_promedio.head(5)      # se observan las compras promedio de los productos por año
```

Out[]:

	Year	MntWines	MntFruits	MntMeatProducts	MntFishProducts	MntSweetProducts	MntGoldProds
0	2012	388.803644	29.718623	197.524291	44.473684	31.951417	53.01
1	2013	300.317073	26.291001	165.540791	36.513036	26.904962	46.14
2	2014	236.391382	23.296230	142.842011	33.524237	23.064632	31.54

```
In [ ]: round(data1.pivot_table(['MntWines', 'MntFruits', 'MntMeatProducts', 'MntFishProducts', 'MntSweetProducts', 'MntGoldProds'], ['Year'], aggfunc= 'mean', fill_value=0), 2)
```

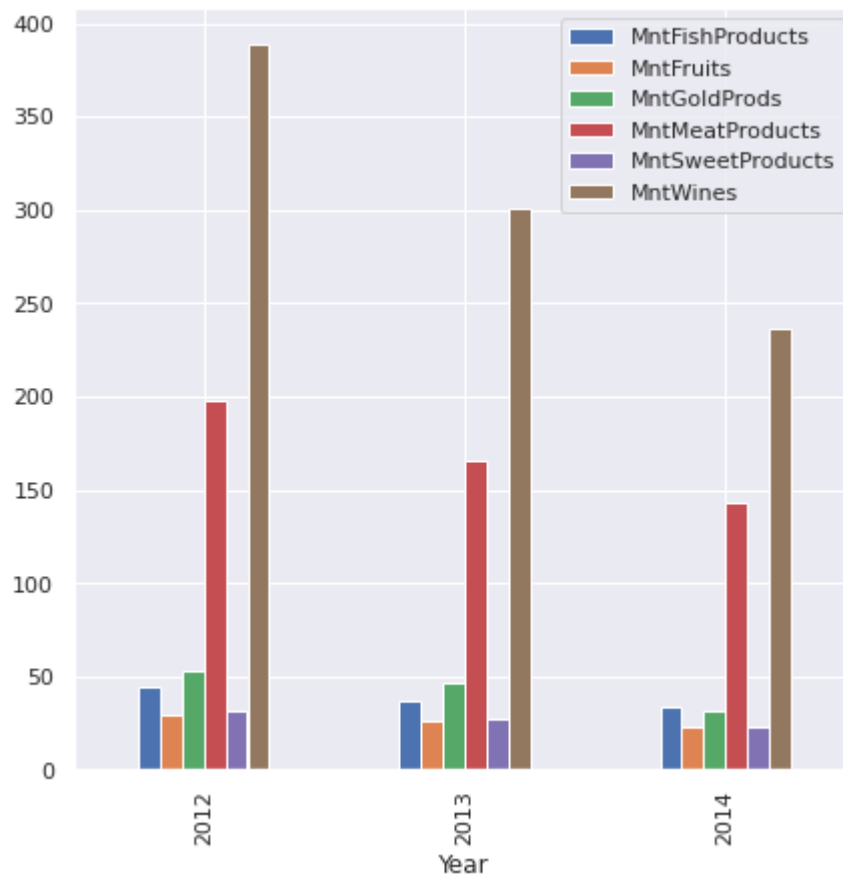
Out[]:

	MntFishProducts	MntFruits	MntGoldProds	MntMeatProducts	MntSweetProducts	MntWines
Year						
2012	44.47	29.72	53.01	197.52	31.95	388.80
2013	36.51	26.29	46.14	165.54	26.90	300.32
2014	33.52	23.30	31.54	142.84	23.06	236.39

```
In [ ]: round(data1.pivot_table(['MntWines', 'MntFruits', 'MntMeatProducts', 'MntFishProducts', 'MntSweetProducts', 'MntGoldProds'], ['Year'], aggfunc= 'mean', fill_value=0), 2).plot(kind = 'bar')

# El comportamiento de compra de los clientes segmentado por el año, permite ver que la característica que más compran los clientes es Bebidas Alcoholicas (Wines) seguidamente
# Los productos carnicos (Meat), ademas el nivel compra de los clientes viene en descenso (Los clientes están comprando menos)
```

```
Out[ ]: <matplotlib.axes._subplots.AxesSubplot at 0x7faaa475cd90>
```



Tablas Pivotes que permitan Mostrar la Tendencia de Compra de los Diferentes Productos

```
In [ ]: plt.figure(figsize=(14,8))
plt.title("Características a travez del Tiempo", fontdict={"fontsize": 15})

lines = ['MntWines','MntFruits','MntMeatProducts','MntFishProducts','MntSweetP
roducts','MntGoldProds']

for line in lines:
    ax = sns.lineplot(x='Year', y=line, data=compras_promedio)
plt.ylabel("value")
plt.legend(lines)

# El nivel compra de los clientes viene en descenso (Los clientes están compra
ndo menos)
```

Out[]: <matplotlib.legend.Legend at 0x7faaa475c790>

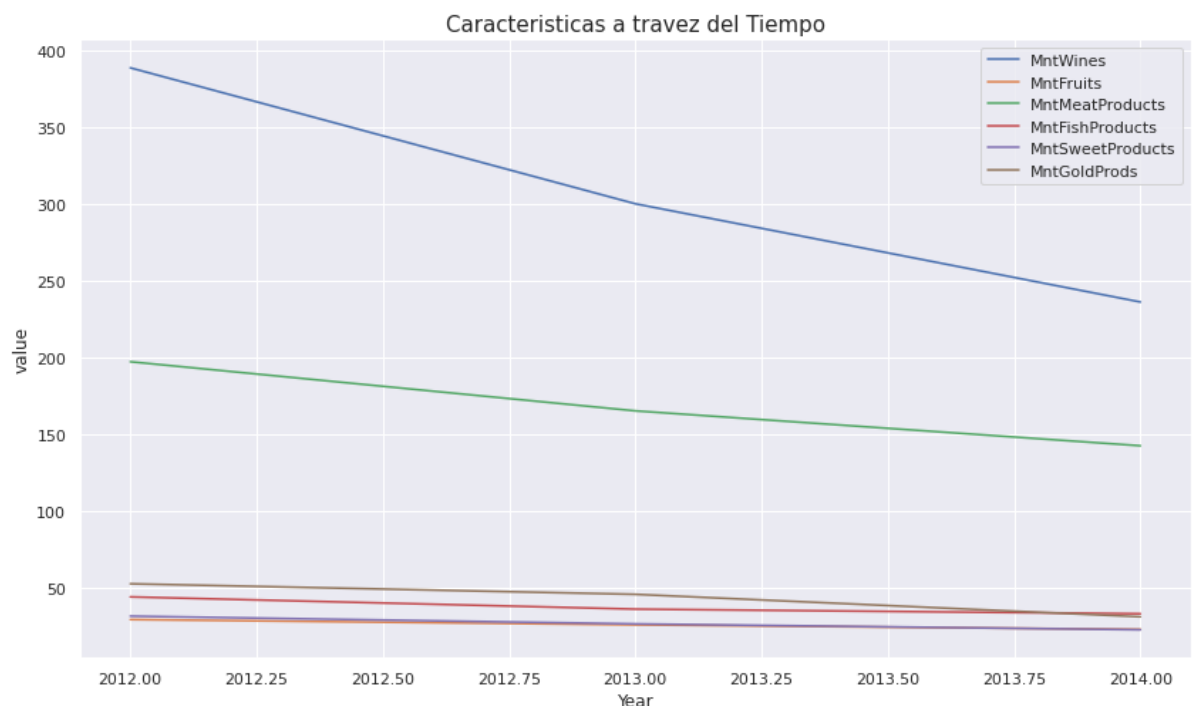


Tabla Pivote que Muestra Comprotamiento de Compra Según Estado Marital

```
In [ ]: compras_promedio2 = data1[['MntWines','MntFruits','MntMeatProducts','MntFishPr
oducts','MntSweetProducts','MntGoldProds','Marital_Status']].\
groupby("Marital_Status").mean().sort_values(by="Marital_Status").reset_index
()
```

```
In [ ]: round((compras_promedio2),2)
```

```
Out[ ]:
```

	Marital_Status	MntWines	MntFruits	MntMeatProducts	MntFishProducts	MntSweetProducts
0	Absurd	355.50	84.50	312.50	205.50	30.50
1	Alone	184.67	4.00	26.33	7.67	7.00
2	Divorced	324.84	27.43	150.21	35.04	26.82
3	Married	299.48	25.73	160.68	35.38	26.70
4	Single	288.33	26.84	182.11	38.22	27.26
5	Together	306.83	25.35	168.10	38.99	26.12
6	Widow	369.27	33.09	189.29	51.39	39.01
7	YOLO	322.00	3.00	50.00	4.00	3.00

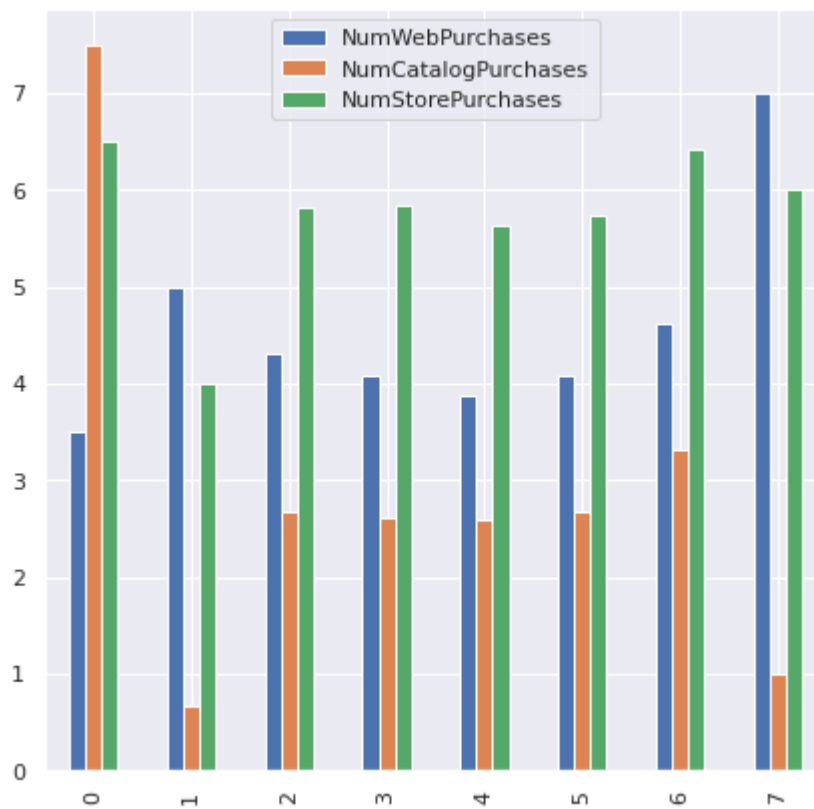
```
In [ ]: compras_promedio4 = data1[['NumWebPurchases','NumCatalogPurchases','NumStorePurchases','Marital_Status']].\ngroupby("Marital_Status").mean().sort_values(by="Marital_Status").reset_index()
```

```
In [ ]: round((compras_promedio4),2)
```

```
Out[ ]:
```

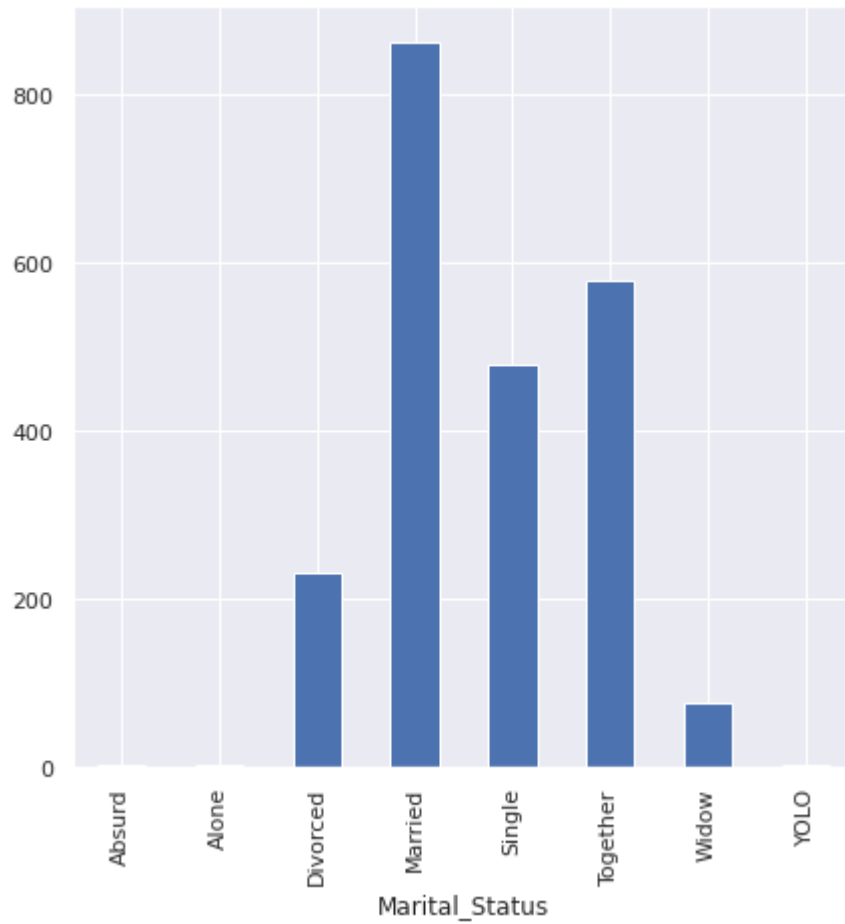
	Marital_Status	NumWebPurchases	NumCatalogPurchases	NumStorePurchases
0	Absurd	3.50	7.50	6.50
1	Alone	5.00	0.67	4.00
2	Divorced	4.31	2.67	5.82
3	Married	4.09	2.62	5.85
4	Single	3.87	2.60	5.64
5	Together	4.08	2.68	5.74
6	Widow	4.62	3.32	6.42
7	YOLO	7.00	1.00	6.00

```
In [ ]: round((compras_promedio4),2).plot(kind = 'bar')
sns.set(rc={'figure.figsize':(7,7)})
plt.show()
```



```
In [ ]: data1.groupby('Marital_Status').size().plot(kind = 'bar')
sns.set(rc={'figure.figsize':(7,7)})
plt.show()
```

El mayor grupo al que pertenecen los clientes es al grupo de Casados, union libre y le sigue la condicion sontero(single)




```
In [ ]: round(data1.pivot_table('MntWines', index= 'Marital_Status', columns= 'Complain', aggfunc= 'mean',fill_value=0),2)
```

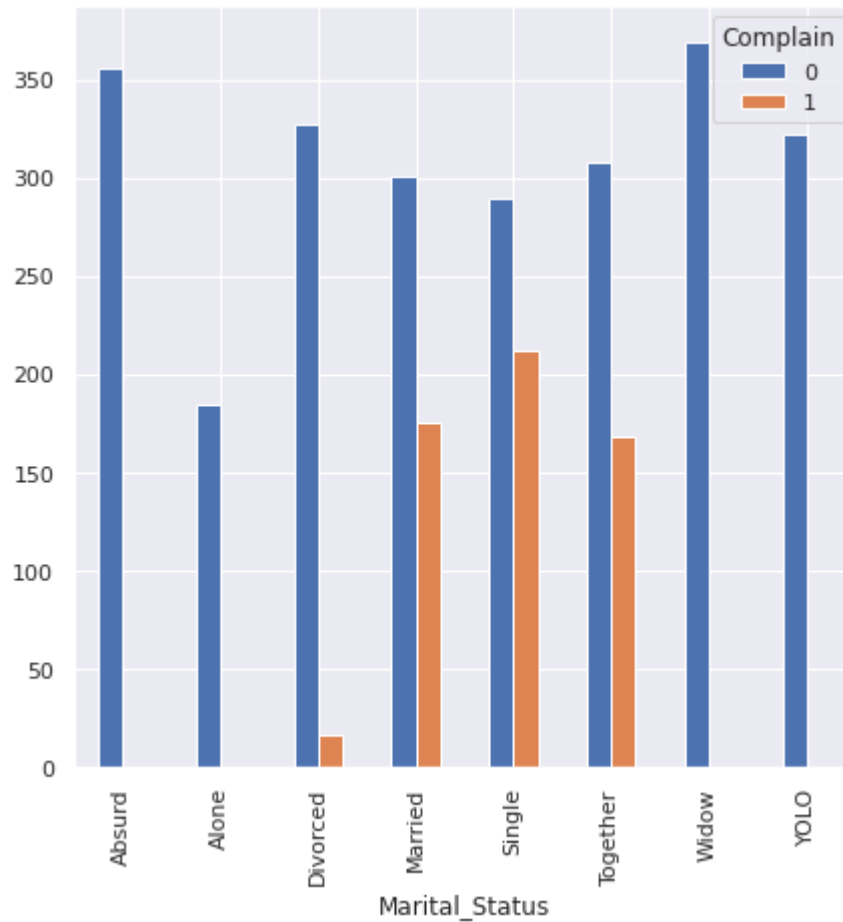
```
# El nivel de consumo de bebidas (wines) de Los clientes según el estatus marital y según si se quejan o no,  
# nos indica que el grupo de clientes Viudos (widow) son Los que más consumen bebidas (wines) y le siguen Los  
# que consideran las personas con estatus Absurd y Los Divorciados.  
# el mayor consumo se presenta en aquellos que no se quejan.
```

Out[]:

Complain	0	1
Marital_Status		
Absurd	355.50	0.00
Alone	184.67	0.00
Divorced	327.53	16.00
Married	300.64	175.50
Single	289.30	211.83
Together	308.03	168.40
Widow	369.27	0.00
YOLO	322.00	0.00

```
In [ ]: round(data1.pivot_table('MntWines', index= 'Marital_Status', columns= 'Complain', aggfunc= 'mean', fill_value=0),2).plot(kind = 'bar')
```

```
Out[ ]: <matplotlib.axes._subplots.AxesSubplot at 0x7faaa48e9950>
```



```
In [ ]: data1.pivot_table('NumWebPurchases', index= ('Marital_Status'), aggfunc='mean')
```

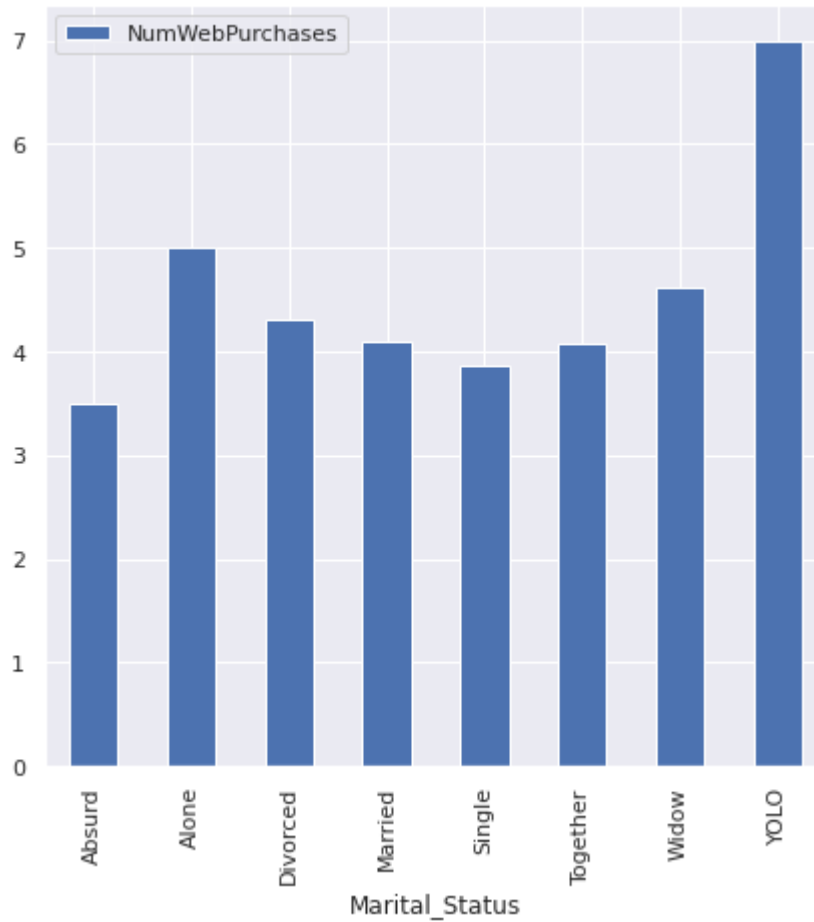
```
Out[ ]:
```

NumWebPurchases	
Marital_Status	
Absurd	3.500000
Alone	5.000000
Divorced	4.310345
Married	4.087963
Single	3.872917
Together	4.081034
Widow	4.623377
YOLO	7.000000

```
In [ ]: data1.pivot_table('NumWebPurchases', index= ('Marital_Status'), aggfunc='mean')
        .plot(kind = 'bar')

# El comportamiento de compra por medio de la Web , permite observar que Lo qu
e más compran
# por el medio de la WEB corresponde aquellos que se denominan YOLO y Los alon
e.
```

```
Out[ ]: <matplotlib.axes._subplots.AxesSubplot at 0x7faaa4995350>
```



```
In [ ]: data1.pivot_table('NumStorePurchases', index= ('Marital_Status'), aggfunc='mean').plot(kind = 'bar')
```

```
Out[ ]: <matplotlib.axes._subplots.AxesSubplot at 0x7faaa4c53a90>
```

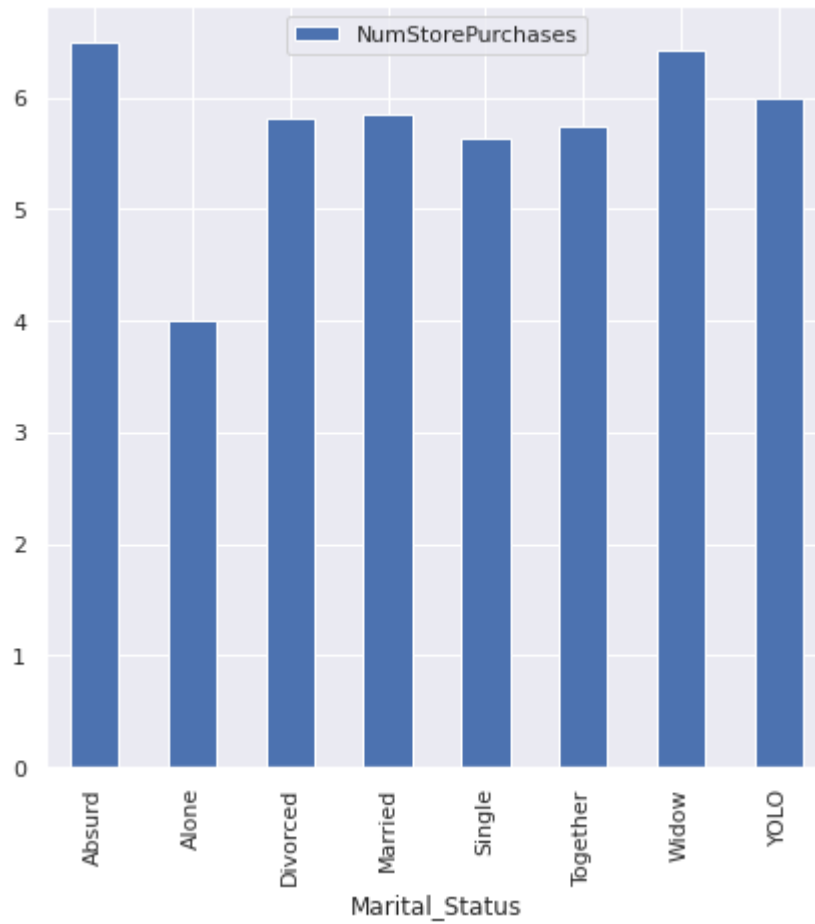


Tabla Pivote que permite ver el nivel de consumo de productos alcoholicos según el nivel de estudio y estatus marital

```
In [ ]: round(data1.pivot_table('MntWines', ['Education', 'Marital_Status'], ['Response'],aggfunc= 'mean',fill_value=0),2)
```

Out[]:

		Response	0	1
Education	Marital_Status			
2n Cycle	Divorced		334.11	412.60
	Married		137.40	313.38
	Single		168.23	425.50
	Together		181.37	315.33
	Widow		221.80	0.00
Basic	Divorced		0.00	0.00
	Married		15.20	0.00
	Single		3.11	0.00
	Together		1.75	3.50
	Widow		3.00	0.00
Graduation	Absurd		0.00	471.00
	Alone		5.00	0.00
	Divorced		277.93	419.79
	Married		253.49	445.63
	Single		210.14	467.56
	Together		281.02	569.32
	Widow		272.83	423.50
Master	Absurd		240.00	0.00
	Alone		534.00	0.00
	Divorced		268.58	471.00
	Married		294.95	596.40
	Single		354.07	476.78
	Together		291.67	391.75
	Widow		344.83	522.67
PhD	Alone		0.00	15.00
	Divorced		302.97	559.17
	Married		391.01	628.16
	Single		296.96	509.20
	Together		349.82	752.75
	Widow		408.94	673.14
	YOLO		322.00	322.00

```
In [ ]: round(data1.pivot_table(['MntWines','MntFruits','MntMeatProducts'], ['Marital_Status'], ['Response'], aggfunc= 'mean', fill_value=0),2)
```

```
# Como se comporta el consumo de los clientes según su estatus marital y si responden o no a las campañas.
# Aquellos que reaccionan positivamente a las campañas presentan un nivel de consumo mayor que aquellos no
# reaccionan a la campaña salvo en la categoría de marital estatus Absurd y el producto Meat, donde los que
# no responden positivamente a la campaña tienen un mayor consumo (muy alto) que aquellos que sí responden

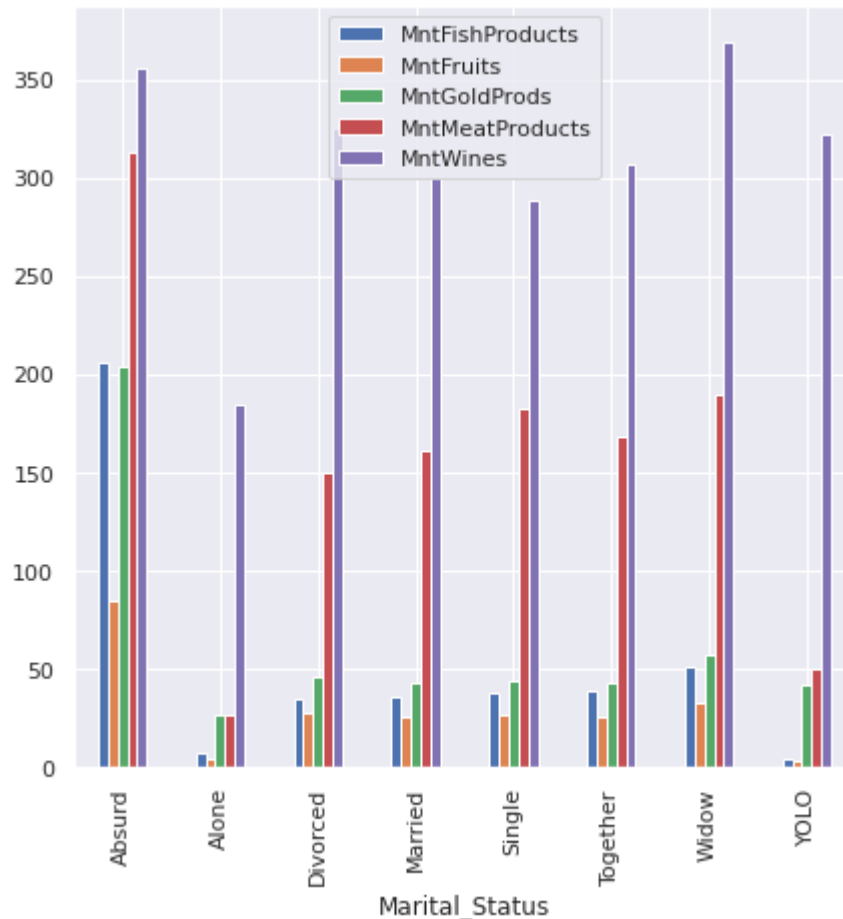
# Response: 1 indica si el cliente aceptó la oferta en la última campaña, 0 que no
```

```
Out[ ]:
```

	MntFruits		MntMeatProducts		MntWines	
Response	0	1	0	1	0	1
Marital_Status						
Absurd	67.00	102.00	500.00	125.00	240.00	471.00
Alone	6.00	0.00	35.50	8.00	269.50	15.00
Divorced	22.28	47.17	112.60	294.38	284.97	477.71
Married	24.62	34.45	146.52	271.40	271.59	517.51
Single	23.23	39.57	141.15	326.63	234.43	478.53
Together	24.08	36.33	154.02	290.15	280.04	538.93
Widow	32.88	33.74	158.67	282.74	311.12	546.79
YOLO	3.00	3.00	50.00	50.00	322.00	322.00

```
In [ ]: round(data1.pivot_table(['MntWines', 'MntFruits', 'MntMeatProducts', 'MntFishProducts', 'MntGoldProds'], ['Marital_Status'], aggfunc= 'mean', fill_value=0), 2).plot(kind = 'bar')
```

```
Out[ ]: <matplotlib.axes._subplots.AxesSubplot at 0x7faaa4c3a690>
```



Comportamiento de Compra según Grado Académico

```
In [ ]: compras_promedio3 = data1[['MntWines', 'MntFruits', 'MntMeatProducts', 'MntFishProducts', 'MntSweetProducts', 'MntGoldProds', 'Education']].\ngroupby("Education").mean().sort_values(by="Education").reset_index()
```

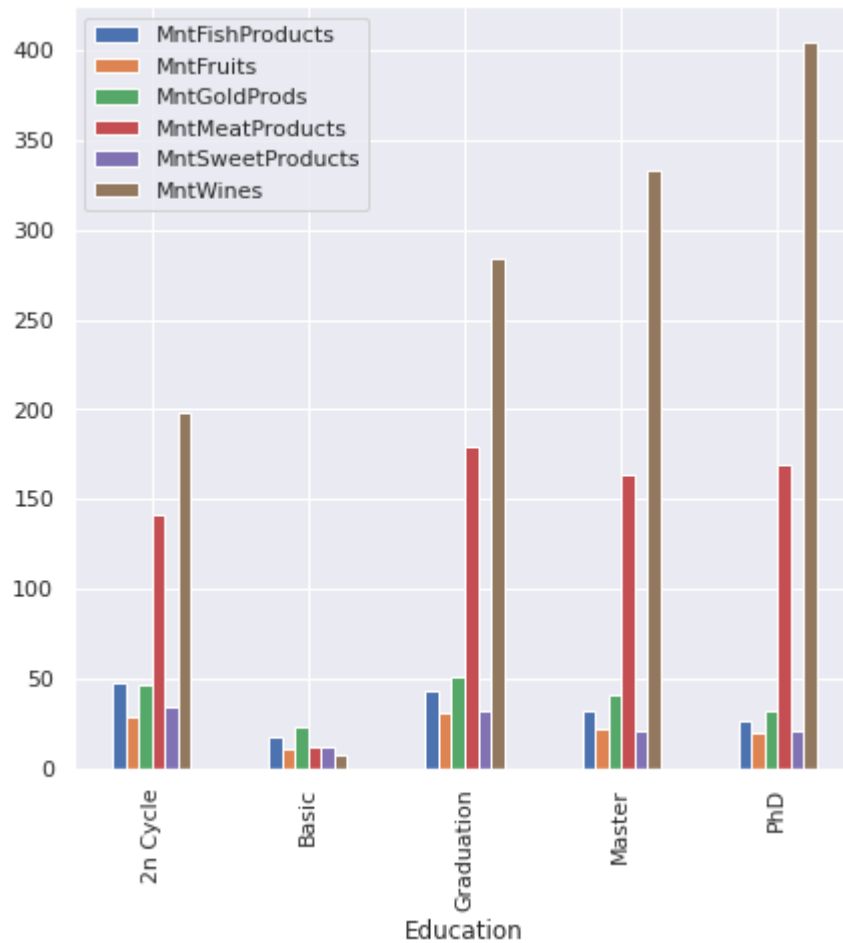
```
In [ ]: round((compras_promedio3), 2)
```

```
Out[ ]:
```

	Education	MntWines	MntFruits	MntMeatProducts	MntFishProducts	MntSweetProducts	MntGoldProds
0	2n Cycle	198.18	28.96	141.26	47.48	34.25	20.00
1	Basic	7.24	11.11	11.44	17.06	12.11	2.00
2	Graduation	284.27	30.77	179.49	43.15	31.37	20.00
3	Master	333.08	21.65	163.38	32.10	21.18	20.00
4	PhD	404.50	20.05	168.60	26.73	20.22	20.00

```
In [ ]: round(data1.pivot_table(['MntWines', 'MntFruits', 'MntMeatProducts', 'MntFishProducts', 'MntSweetProducts', 'MntGoldProds'], ['Education'], aggfunc= 'mean', fill_value=0), 2).plot(kind = 'bar')
```

```
Out[ ]: <matplotlib.axes._subplots.AxesSubplot at 0x7faaa491f950>
```



```
In [ ]: compras_promedio5 = data1[['NumDealsPurchases', 'NumWebPurchases', 'NumCatalogPurchases', 'NumStorePurchases', 'Education']].\
groupby("Education").mean().sort_values(by="Education").reset_index()
```



```
In [ ]: round((compras_promedio5),2)
```

```
# Comportamiento de compra de los clientes según su nivel de educativo, el mayor comportamiento de compra es por medio de  
# de las tiendas, seguidamente en la web y le sigue las compras por catalogo.
```

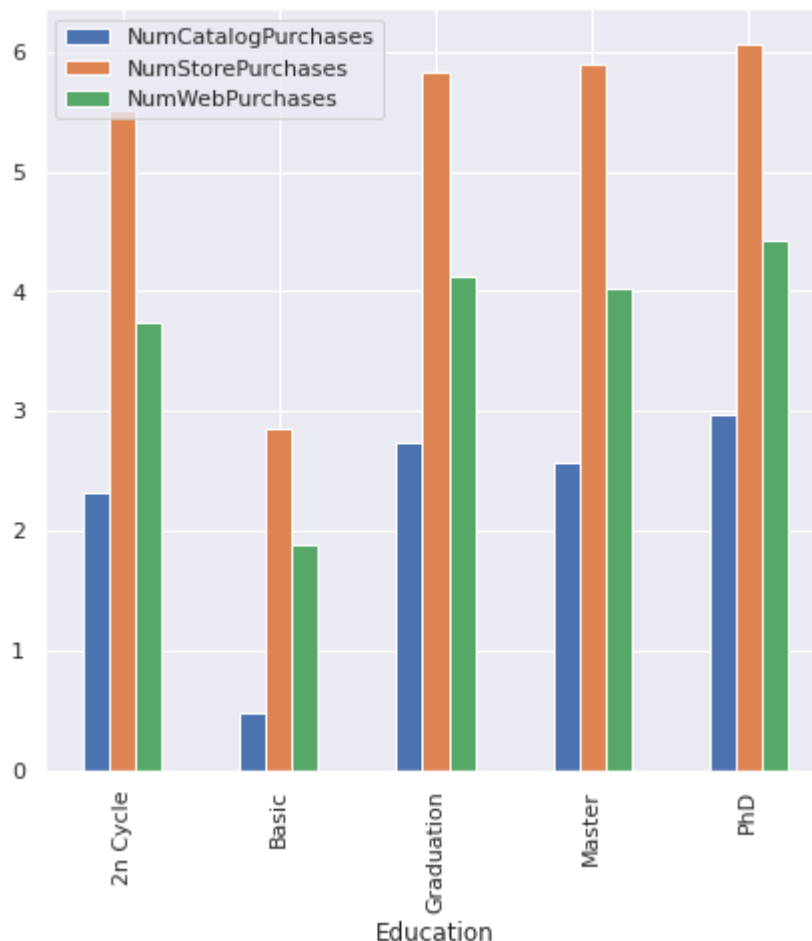
```
Out[ ]:
```

	Education	NumDealsPurchases	NumWebPurchases	NumCatalogPurchases	NumStorePurchases
0	2n Cycle	2.25	3.73	2.32	5.32
1	Basic	1.80	1.89	0.48	2.84
2	Graduation	2.31	4.13	2.73	5.84
3	Master	2.43	4.03	2.57	5.84
4	PhD	2.37	4.42	2.97	6.04

```
In [ ]: round(data1.pivot_table(['NumWebPurchases','NumCatalogPurchases','NumStorePurchases'], ['Education'], aggfunc='mean', fill_value=0),2).plot(kind='bar')
```

```
# Se observa que las compras en las tiendas es la que los clientes prefieren más, y aquellos que más van a ir a las  
# tiendas serán por nivel educativo PhD, Master y Graduation
```

```
Out[ ]: <matplotlib.axes._subplots.AxesSubplot at 0x7faaa4575790>
```



```
In [ ]: data1[['NumDealsPurchases', 'NumWebPurchases', 'NumCatalogPurchases', 'NumStorePurchases']].describe() # en promedio las personas realizan mas compras en la tienda y en segundo lugar por la web
```

Out[]:

	NumDealsPurchases	NumWebPurchases	NumCatalogPurchases	NumStorePurchases
count	2240.000000	2240.000000	2240.000000	2240.000000
mean	2.325000	4.084821	2.662054	5.790179
std	1.932238	2.778714	2.923101	3.250958
min	0.000000	0.000000	0.000000	0.000000
25%	1.000000	2.000000	0.000000	3.000000
50%	2.000000	4.000000	2.000000	5.000000
75%	3.000000	6.000000	4.000000	8.000000
max	15.000000	27.000000	28.000000	13.000000

Tabla Pivote que Muestre el Total de Consumo

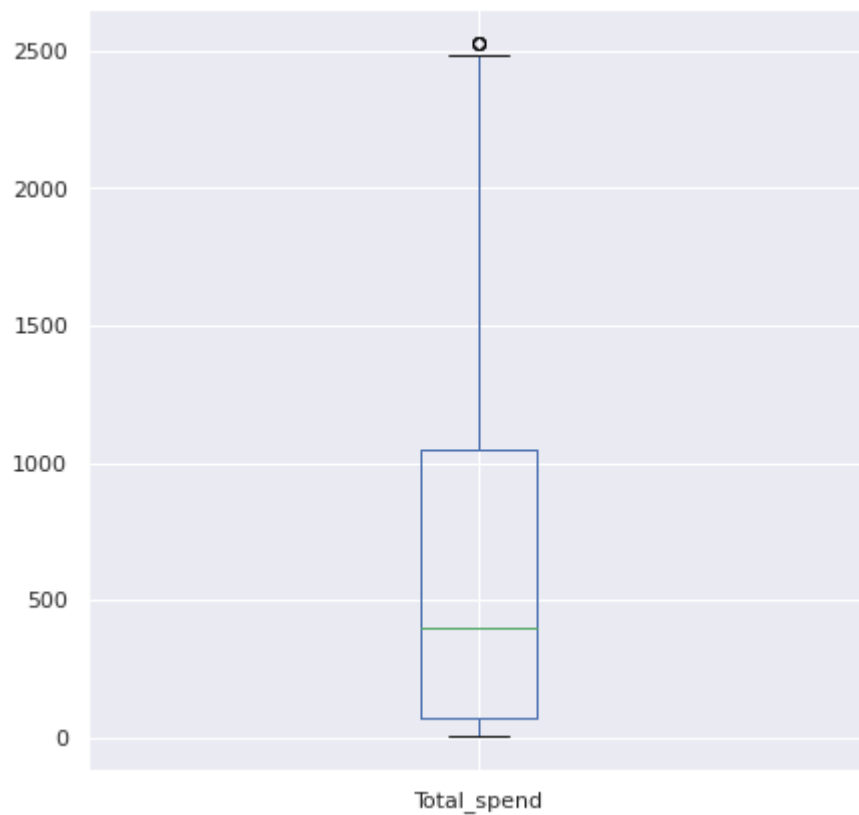
```
In [ ]: data1['Total_spend'] = data1['MntWines'] + data1['MntFruits'] + data1['MntMeatProducts'] + data1['MntFishProducts'] + data1['MntSweetProducts'] + data1['MntGoldProds']
# se crea una columna que represente el total de compras.
```

```
In [ ]: data1.head(3)
```

Out[]:

	ID	Year_Birth	Education	Marital_Status	Income	Kidhome	Teenhome	Dt_Customer	Rec
0	5524	1957	Graduation	Single	58138.0	0	0	2012-04-09	
1	2174	1954	Graduation	Single	46344.0	1	1	2014-08-03	
2	4141	1965	Graduation	Together	71613.0	0	0	2013-08-21	

```
In [ ]: data1['Total_spend'].plot(kind='box')
sns.set(rc={'figure.figsize':(5,5)})
plt.show()
```



```
In [ ]: data1['Total_spend'].plot(kind='hist')
sns.set(rc={'figure.figsize':(5,5)})
plt.show()
```

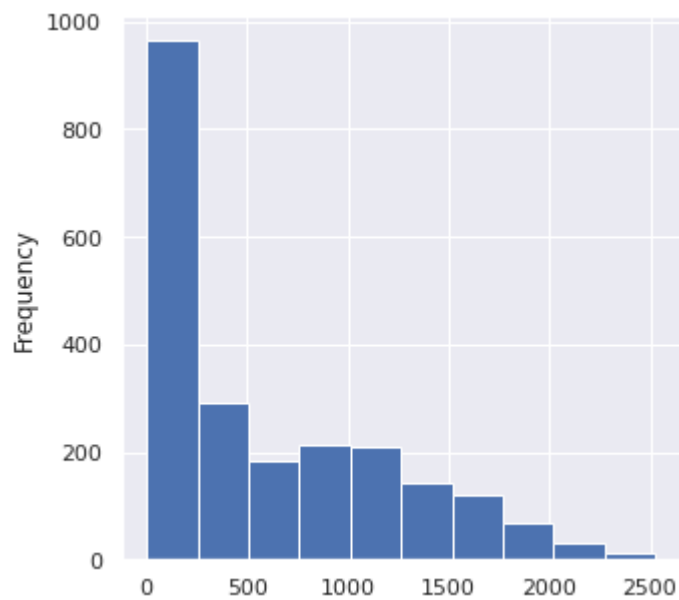


Tabla Pivote que Muestra el Total de Compra por Estatus Marital (estado civil)

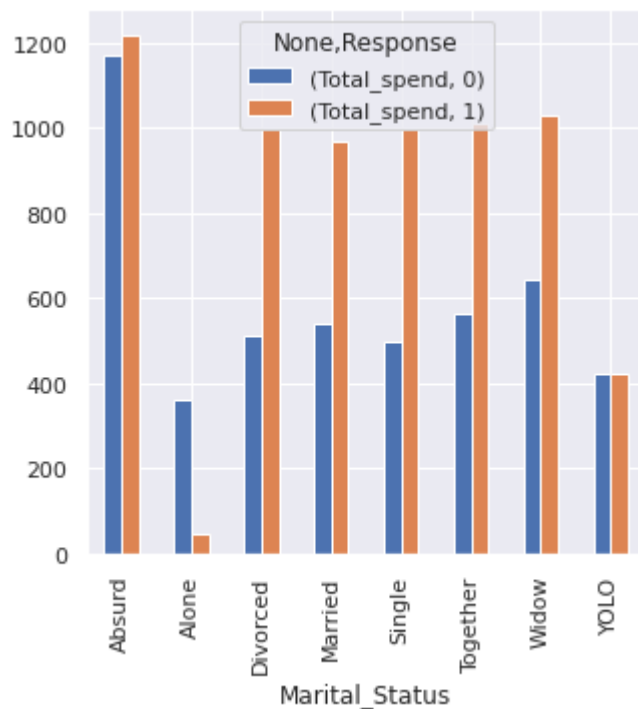
```
In [ ]: round(data1.pivot_table(['Total_spend'], ['Marital_Status'], ['Response'], aggfunc= 'mean', fill_value=0), 2) # 1 indica si el cliente aceptó la oferta en la última campaña, 0 que no.
```

Out[]:

	Total_spend	
	0	1
Marital_Status		
Absurd	1169.00	1216.00
Alone	360.50	49.00
Divorced	510.58	994.15
Married	542.52	968.18
Single	496.66	993.99
Together	562.04	1010.08
Widow	644.10	1027.95
YOLO	424.00	424.00

```
In [ ]: round(data1.pivot_table(['Total_spend'], ['Marital_Status'], ['Response'], aggfunc= 'mean', fill_value=0), 2).plot(kind= 'bar')
```

Out[]: <matplotlib.axes._subplots.AxesSubplot at 0x7faaa4ad5950>



La mayoría de los clientes que respondieron a la campaña presentaron un nivel de compra mayor que aquellos que no respondieron a la campaña publicitaria o promociones.

```
In [ ]: data1['Total_spend'].min()
```

```
Out[ ]: 5
```

```
In [ ]: data1['Total_spend'].max()
```

```
Out[ ]: 2525
```

```
In [ ]: data1['Total_spend'].mean()    # el consumo promedio o compra promedio de los c  
      # lientes corresponde a 605.79
```

```
Out[ ]: 605.7982142857143
```

Mostrar el Total de Hijos por Hogar tanto Niños y Adolescentes

```
In [ ]: data1['Total_Sons'] = data1['Kidhome']+data1['Teenhome']  
      # se crea una columna que represente el total de hijos por hogar
```

```
In [ ]: data1['Total_Sons'].head(3)
```

```
Out[ ]: 0    0  
      1    2  
      2    0  
      Name: Total_Sons, dtype: int64
```

```
In [ ]: data1['Total_Sons'].min()    # una vez que se suman las características kid y t  
      # een , el valor mínimo de hijos por hogar es cero
```

```
Out[ ]: 0
```

```
In [ ]: data1['Total_Sons'].max()    # una vez que se suman las características kid y t  
      # een , el valor máximo de hijos por hogar es de 3
```

```
Out[ ]: 3
```

```
In [ ]: data1['Total_Sons'].mean()    # una vez que se suman las características kid y  
      # teen, el valor promedio de hijos por hogar se acerca a 1
```

```
Out[ ]: 0.9504464285714286
```

```
In [ ]: round(data1.pivot_table(['Total_spend'], ['NumDealsPurchases'], aggfunc= 'mean',  
fill_value=0),2)
```

Out[]:

Total_spend	
NumDealsPurchases	
0	1234.76
1	720.34
2	429.71
3	474.36
4	498.51
5	581.84
6	592.64
7	638.30
8	698.07
9	635.12
10	795.40
11	1138.60
12	676.00
13	952.00
15	904.57

```
In [ ]: plt.figure(figsize = (20,20))
round(data1.pivot_table('Total_spend', index= 'age_range', columns= 'Complain'
, aggfunc= 'mean',fill_value=0),2).plot()
plt.title("Comportamiento de Compra de Clientes")
plt.ylabel("Compra Promedio Total")
plt.xlabel("Por edades")
plt.show();
```

<Figure size 1440x1440 with 0 Axes>

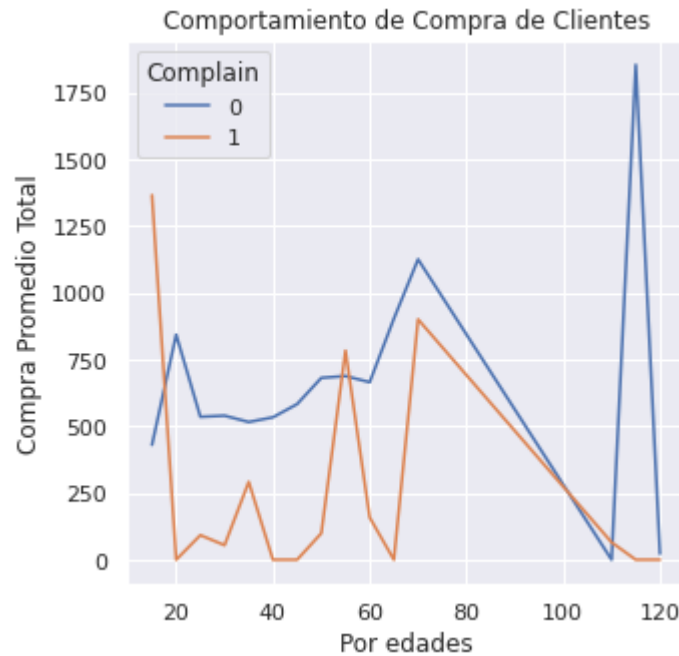


Tabla Total Spend, Education y Marital Status según si Responden o No a las Campañas

```
In [ ]: round(data1.pivot_table('Total_spend', ['Education', 'Marital_Status'], ['Response'], aggfunc= 'mean', fill_value=0), 2)
```

Out[]:

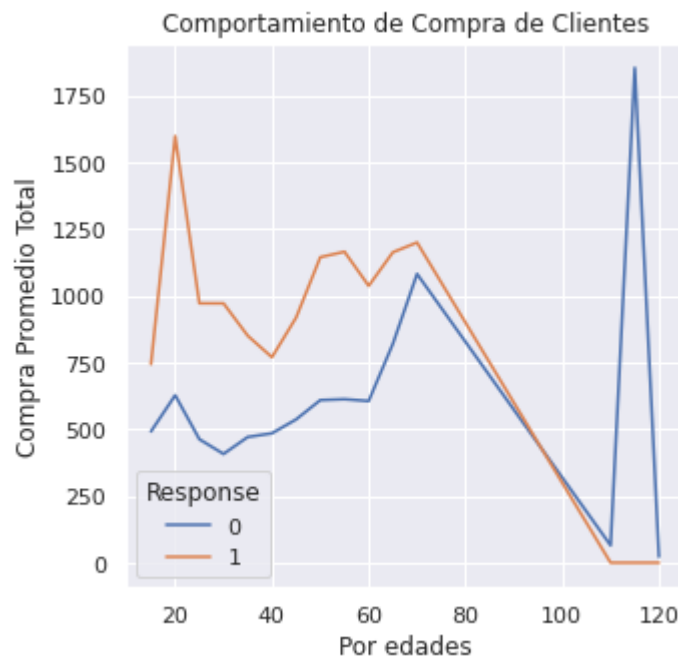
		Response	0	1
Education	Marital_Status			
2n Cycle	Divorced		565.17	960.80
	Married		394.44	691.62
	Single		450.74	995.67
	Together		476.52	808.67
	Widow		677.20	0.00
Basic	Divorced		29.00	0.00
	Married		122.85	0.00
	Single		57.72	0.00
	Together		55.75	60.50
	Widow		102.00	0.00
Graduation	Absurd		0.00	1216.00
	Alone		89.00	0.00
	Divorced		536.51	1036.95
	Married		559.72	924.14
	Single		503.60	1020.17
	Together		598.57	1155.90
	Widow		618.07	962.67
Master	Absurd		1169.00	0.00
	Alone		632.00	0.00
	Divorced		471.10	965.67
	Married		493.95	1162.93
	Single		666.86	996.78
	Together		542.11	707.67
	Widow		603.67	1155.83
PhD	Alone		0.00	49.00
	Divorced		455.59	967.72
	Married		657.95	1005.22
	Single		470.66	946.60
	Together		593.01	1144.42
	Widow		724.94	974.29
	YOLO		424.00	424.00

Total de Compra por Rangos de Edad y Respuesta a la Campaña

1 indica si el cliente aceptó la oferta en la ultima campaña, 0 que no.

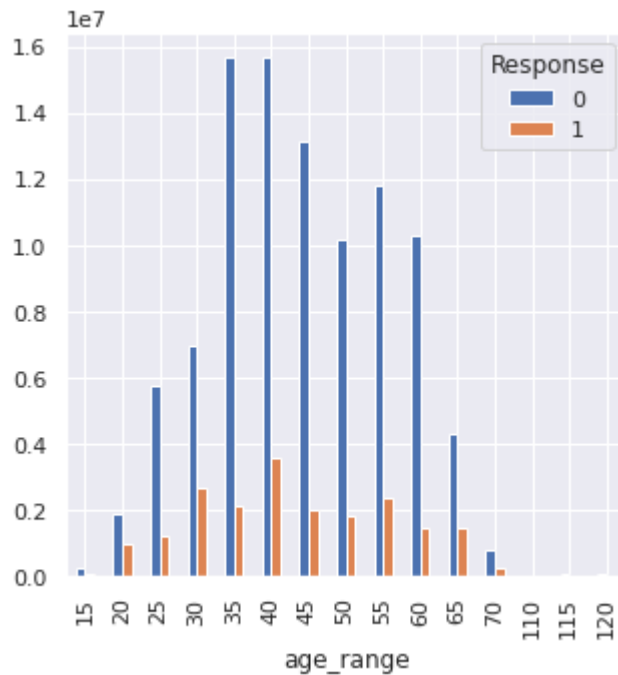
```
In [ ]: plt.figure(figsize = (20,20))
round(data1.pivot_table('Total_spend', index= 'age_range', columns= 'Response'
, aggfunc= 'mean',fill_value=0),2).plot()
plt.title("Comportamiento de Compra de Clientes")
plt.ylabel("Compra Promedio Total")
plt.xlabel("Por edades")
plt.show();
```

<Figure size 1440x1440 with 0 Axes>



```
In [ ]: plt.figure(figsize = (20,20))
round(data1.pivot_table('Income', index= 'age_range', columns= 'Response', agg
func= 'sum',fill_value=0),2).plot(kind = 'bar')
plt.show();
```

<Figure size 1440x1440 with 0 Axes>



Comportamiento de Compra Total por el Total de Hijos por hogar

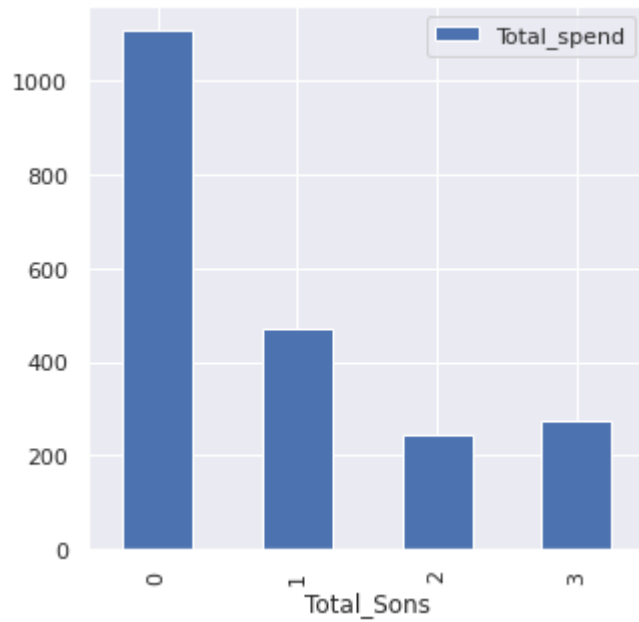
```
In [ ]: round(data1.pivot_table(['Total_spend'], ['Total_Sons'], aggfunc= 'mean', fill_value=0), 2)
```

Out[]:

Total_spend	
Total_Sons	
0	1106.03
1	472.73
2	245.95
3	274.60

```
In [ ]: round(data1.pivot_table(['Total_spend'], ['Total_Sons'], aggfunc= 'mean', fill_value=0), 2).plot(kind = 'bar') # según esta información aquellos que realizan más monto de compras son los que no tienen hijos
```

```
Out[ ]: <matplotlib.axes._subplots.AxesSubplot at 0x7faaa419f3d0>
```



Correlaciones de Características

```
In [ ]: data2['Age'] = 2014 - data2['Year_Birth']
```

In []: data2.corr()

Out[]:

	ID	Year_Birth	Education	Marital_Status	Income	Kidhome	Te
ID	1.000000	0.000028	-0.003839	0.019580	0.012996	0.002406	-C
Year_Birth	0.000028	1.000000	-0.171390	-0.060580	-0.160899	0.230176	-(
Education	-0.003839	-0.171390	1.000000	0.007090	0.119979	-0.045564	(
Marital_Status	0.019580	-0.060580	0.007090	1.000000	0.021145	-0.022553	-C
Income	0.012996	-0.160899	0.119979	0.021145	1.000000	-0.425326	C
Kidhome	0.002406	0.230176	-0.045564	-0.022553	-0.425326	1.000000	-C
Teenhome	-0.002580	-0.352111	0.118485	-0.003596	0.018965	-0.036133	1
Recency	-0.046524	-0.019871	-0.011728	0.014159	-0.004061	0.008827	C
MntWines	-0.022878	-0.157773	0.197576	0.008205	0.576903	-0.496297	C
MntFruits	0.004600	-0.017917	-0.080412	0.000593	0.428791	-0.372581	-C
MntMeatProducts	-0.004437	-0.030872	0.033625	0.030689	0.577805	-0.437129	-C
MntFishProducts	-0.024475	-0.041625	-0.112223	0.035808	0.437564	-0.387644	-C
MntSweetProducts	-0.007642	-0.018133	-0.105217	0.017382	0.436131	-0.370673	-C
MntGoldProds	-0.013438	-0.061818	-0.095489	0.001688	0.321938	-0.349595	-C
NumDealsPurchases	-0.037166	-0.060846	0.030075	-0.021772	-0.082315	0.221798	C
NumWebPurchases	-0.018924	-0.145040	0.081908	-0.001894	0.380554	-0.361647	C
NumCatalogPurchases	-0.003440	-0.121275	0.070782	0.015125	0.586826	-0.502237	-C
NumStorePurchases	-0.014927	-0.128272	0.070483	0.001412	0.526600	-0.499683	C
NumWebVisitsMonth	-0.007446	0.121139	-0.040281	-0.031210	-0.549785	0.447846	C
AcceptedCmp3	-0.036040	0.061774	0.005836	-0.027113	-0.016064	0.014674	-C
AcceptedCmp4	-0.025387	-0.060510	0.053266	0.014381	0.182718	-0.161600	C
AcceptedCmp5	-0.007517	0.007123	0.033346	0.012817	0.334893	-0.205634	-C
AcceptedCmp1	-0.021614	-0.005930	-0.010845	-0.017097	0.274891	-0.172339	-C
AcceptedCmp2	-0.015061	-0.006539	0.021369	0.018417	0.087581	-0.081716	-C
Complain	0.033883	-0.030128	-0.050540	-0.005718	-0.027187	0.040207	C
Z_CostContact	NaN	NaN	NaN	NaN	NaN	NaN	
Z_Revenue	NaN	NaN	NaN	NaN	NaN	NaN	
Response	-0.021968	0.021325	0.090819	-0.011403	0.132867	-0.080008	-C
Age	-0.000028	-1.000000	0.171390	0.060580	0.160899	-0.230176	(



```
In [ ]: cor_matrix = data2.corr().abs() # esta versión permite colorear aquellas cor
relaciones que nos llaman la atención tanto positivas como negativas
cor_matrix.style.background_gradient(sns.light_palette('red', as_cmap=True))
# código es muy útil ayuda cuando hay muchas variables
```

Out[]:

	ID	Year_Birth	Education	Marital_Status	Income	Kidhome	Teen
ID	1.000000	0.000028	0.003839	0.019580	0.012996	0.002406	0.0
Year_Birth	0.000028	1.000000	0.171390	0.060580	0.160899	0.230176	0.3
Education	0.003839	0.171390	1.000000	0.007090	0.119979	0.045564	0.1
Marital_Status	0.019580	0.060580	0.007090	1.000000	0.021145	0.022553	0.0
Income	0.012996	0.160899	0.119979	0.021145	1.000000	0.425326	0.0
Kidhome	0.002406	0.230176	0.045564	0.022553	0.425326	1.000000	0.0
Teenhome	0.002580	0.352111	0.118485	0.003596	0.018965	0.036133	1.0
Recency	0.046524	0.019871	0.011728	0.014159	0.004061	0.008827	0.0
MntWines	0.022878	0.157773	0.197576	0.008205	0.576903	0.496297	0.0
MntFruits	0.004600	0.017917	0.080412	0.000593	0.428791	0.372581	0.1
MntMeatProducts	0.004437	0.030872	0.033625	0.030689	0.577805	0.437129	0.2
MntFishProducts	0.024475	0.041625	0.112223	0.035808	0.437564	0.387644	0.2
MntSweetProducts	0.007642	0.018133	0.105217	0.017382	0.436131	0.370673	0.1
MntGoldProds	0.013438	0.061818	0.095489	0.001688	0.321938	0.349595	0.0
NumDealsPurchases	0.037166	0.060846	0.030075	0.021772	0.082315	0.221798	0.3
NumWebPurchases	0.018924	0.145040	0.081908	0.001894	0.380554	0.361647	0.1
NumCatalogPurchases	0.003440	0.121275	0.070782	0.015125	0.586826	0.502237	0.1
NumStorePurchases	0.014927	0.128272	0.070483	0.001412	0.526600	0.499683	0.0
NumWebVisitsMonth	0.007446	0.121139	0.040281	0.031210	0.549785	0.447846	0.1
AcceptedCmp3	0.036040	0.061774	0.005836	0.027113	0.016064	0.014674	0.0
AcceptedCmp4	0.025387	0.060510	0.053266	0.014381	0.182718	0.161600	0.0
AcceptedCmp5	0.007517	0.007123	0.033346	0.012817	0.334893	0.205634	0.1
AcceptedCmp1	0.021614	0.005930	0.010845	0.017097	0.274891	0.172339	0.1
AcceptedCmp2	0.015061	0.006539	0.021369	0.018417	0.087581	0.081716	0.0
Complain	0.033883	0.030128	0.050540	0.005718	0.027187	0.040207	0.0
Z_CostContact	nan	nan	nan	nan	nan	nan	
Z_Revenue	nan	nan	nan	nan	nan	nan	
Response	0.021968	0.021325	0.090819	0.011403	0.132867	0.080008	0.1
Age	0.000028	1.000000	0.171390	0.060580	0.160899	0.230176	0.3

Información de las Correlaciones

Existe una correlacion positiva alta en entre la característica Income y las caracteristas **MntWines, MntFruits, MntMeatProducts, MntFishProducts, MntSweetProducts , MntGoldProds**. Por lo tanto si aumenta Income se generará un aumento en las otras características.

Existe una correlacion alta entre la característica de DeasIPurchase con las características de **Kidhome y TeenHome**, entre mayor cantidad de hijos por hogar, mayor la cantidad de compras realizadas con descuentos. Correlacion positiva, si aumenta una tambien umenta la otra.

Existe una correlacion alta entre las características de los productos (**MntWines, MntFruits, MntMeatProducts, MntFishProducts, MntSweetProducts , MntGoldProds**.) y la característica de compras en la tienda (NumstorePurchases) si aumenta una tambien aumenta la otra.

Existe correlacion positiva entre **Income y Kidhome**, de un 0.42, entonces si incrementa el valor de una genera un incremento de la otra. Las personas con mayor nivel económico parecen estar más dispuestos a tener hijos.

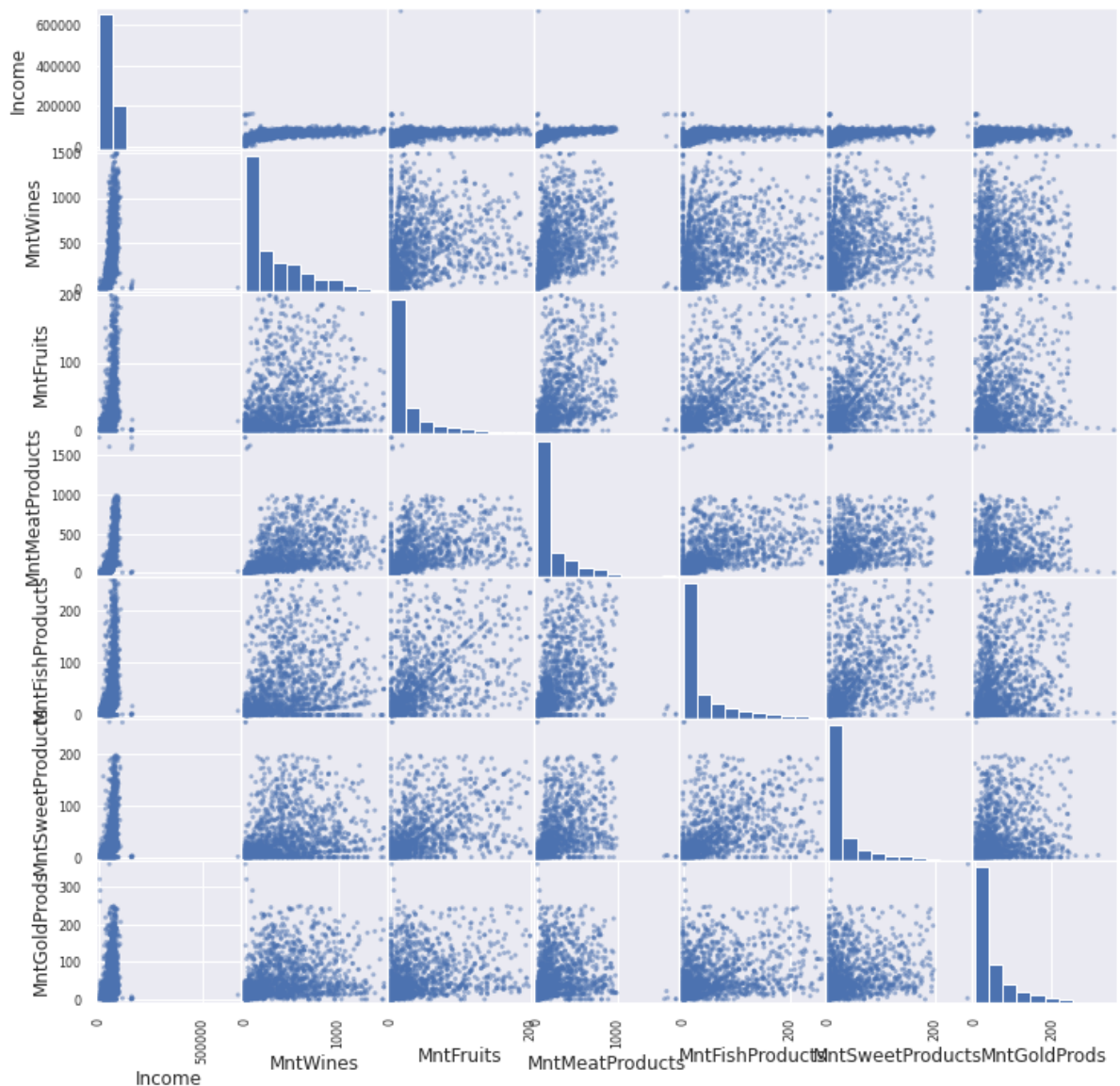
La característica de **Response** tiene una correlacion positiva con las características **AcceptedCmp5 y AcceptedCmp1**

la característica **MntWines** presenta correlacion positiva con **Income y Kidhome**, por lo que indica que al aumentar una tambien incrementa la otra, por lo tanto a mayor numero de hijos, puede presentar un mayor numero de compra de MntWines, y ademas que a mayor Income mayor nivel de compra de la características MntWines.

NumCatalogPurchases tiene alta correslacion con las características **MntWines y MntMeatProducts**, lo que indica que si aumenta una tambien la otra lo hará.

Age la edad de los clientes tiene correlación positiva con la características **TeenHome**, por lo tanto si incrementa una tambien lo hará la otra. Lo que indique que aquellos que presentan hijos adolescentes en el hogar son personas que pasan cierta edad.

```
In [ ]: from pandas.plotting import scatter_matrix
scatter_matrix((data2[['Income', 'MntWines', 'MntFruits', 'MntMeatProducts', 'MntFishProducts', 'MntSweetProducts', 'MntGoldProds']]), figsize = (12, 12));
```



Cuando un cliente compra Fruits, hay alta posibilidad que compra Wines o Meat, por la alta correlacion que existe entre estas caracteristicas.

Grafica de Correlaciones

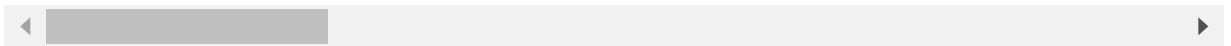
```
In [ ]: data3 = data2.iloc[:, [2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 28, 29]]
```

In []: data3

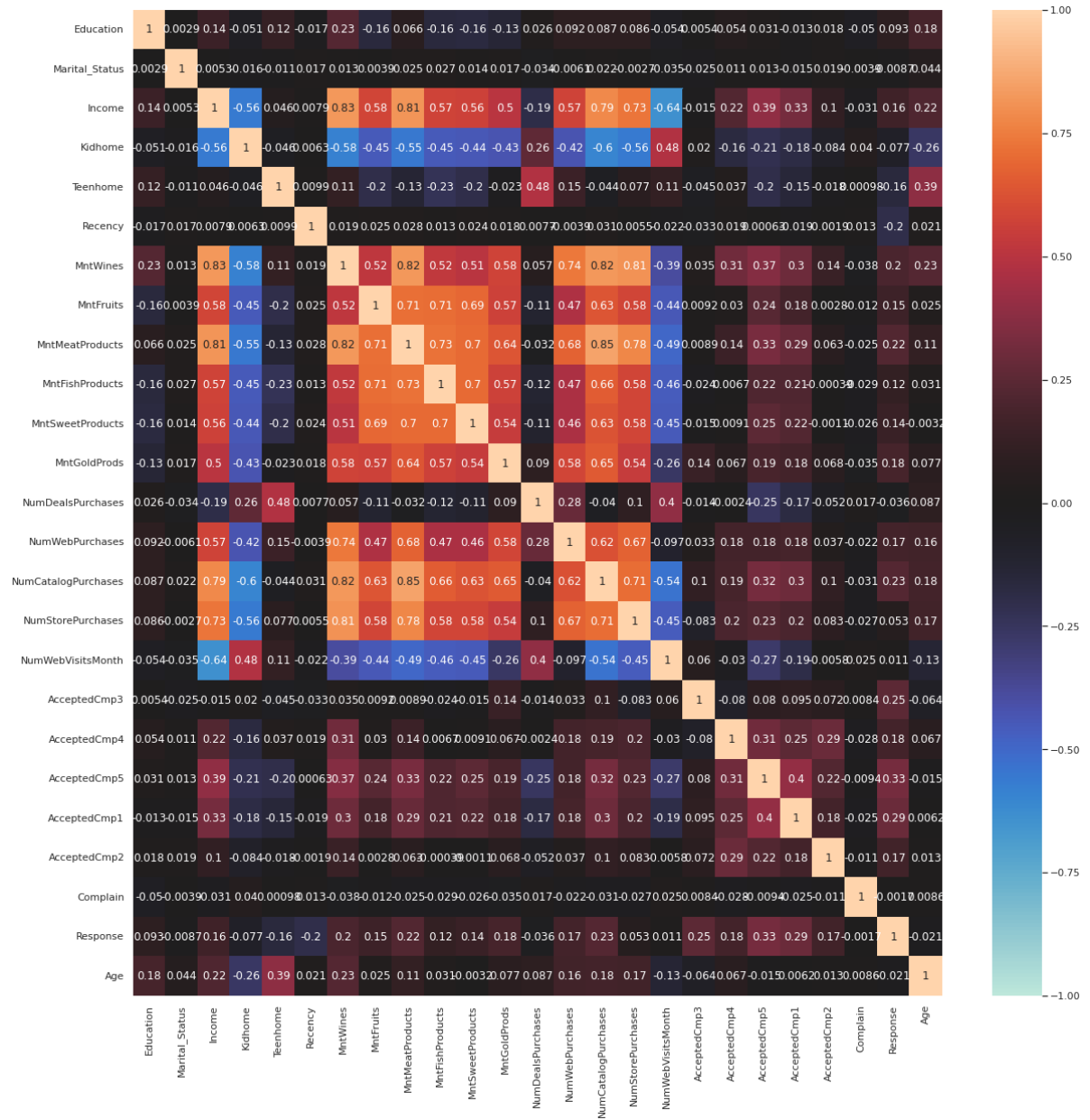
Out[]:

	Education	Marital_Status	Income	Kidhome	Teenhome	Dt_Customer	Recency	MntWine:
0	2	4	58138.0	0	0	2012-04-09	58	63:
1	2	4	46344.0	1	1	2014-08-03	38	1
2	2	5	71613.0	0	0	2013-08-21	26	42:
3	2	5	26646.0	1	0	2014-10-02	26	1
4	4	3	58293.0	1	0	2014-01-19	94	17:
...
2235	2	3	61223.0	0	1	2013-06-13	46	70:
2236	4	5	64014.0	2	1	2014-10-06	56	40:
2237	2	2	56981.0	0	0	2014-01-25	91	90:
2238	3	5	69245.0	0	1	2014-01-24	8	42:
2239	4	3	52869.0	1	1	2012-10-15	40	8.

2240 rows × 26 columns




```
In [ ]: f,ax = plt.subplots(figsize=(20,20))
sns.heatmap(data3.corr(method='spearman'),annot=True,vmin=-1, vmax=1, center=
0)
plt.show()
```



Correlacion alta positiva entre la característica de income y mntWines, 0.83. lo que indica que si aumenta una también aumenta la otra, a mayor ingreso mayor nivel de compra de wines (bebidas alcoholicas)

Income presenta también correlacion alta con las otras características de los productos MntFruits, MntMeatProducts, MntFishProducts, MntSweetProducts, MntGoldProds. Si aumenta una también lo hará la otra. lo que indica que a mayor nivel de ingreso mayor nivel de compra de estos productos.

La característica de NumDealsPurchases tiene una correlacion positiva con las características que indican si hay hijos en el hogar (Kidhome, Teenhome) lo que indica que a mayor hijos o presencia de los mismos mayor compra con descuentos van a presentar los clientes.

NumCatalogPurchases tiene alta correlacion con las características MntWines y MntMeatProducts, lo que indica que si aumenta una también la otra lo hará.

La Característica **Response** tiene correlacion media con MntMeatProducts, MntWines, por lo que si hay una respuesta positiva en las campañas, posiblemente existirá un aumento en las características citadas.

Mientras que la característica **Response** tiene correlación negativa con Recency, lo que indica que al aumentar una disminuye la otra, por lo tanto a mayor días desde la compra del cliente, menor reacción positiva por parte del cliente hacia la campaña.

Existe una correlacion negativa entre la característica NumWebVisitsMonth, y las características de las compras de los productos **MntWines**, **MntFruits**, **MntMeatProducts**, **MntFishProducts**, **MntSweetProducts**, **MntGoldProds**, por lo tanto si aumenta una disminuye la otra.

NumWebVisitsMonth tiene una alta correlacion negativa con la característica **Income** por lo que si incrementa una entonces disminuye la otra.

ANOVA

```
In [ ]: import scipy.stats as stats
```

```
In [ ]: from sklearn.preprocessing import LabelEncoder
```

```
In [ ]: lb_encoder = LabelEncoder()
```

```
In [ ]: print("Influencia del Respuesta campana sobre la vistia web")
print(stats.f_oneway(data3[data3["Response"]== 0]["NumWebVisitsMonth"], data3[
data3["Response"]== 1]["NumWebVisitsMonth"]))
print()
print("Influencia del Respuesta campana sobre compra de frutas")
print(stats.f_oneway(data3[data3["Response"]== 0]["MntFruits"], data3[data3["R
esponse"]== 1]["MntFruits"]))
print()
print("Influencia del Respuesta campana sobre Las Ultima compra")
print(stats.f_oneway(data3[data3["Response"]== 0]["Recency"], data3[data3["Res
ponse"]== 1]["Recency"]))
print()
```

Influencia del Respuesta campana sobre la vistia web

F_onewayResult(statistic=0.035569030632842884, pvalue=0.850425363665631)

Influencia del Respuesta campana sobre compra de frutas

F_onewayResult(statistic=35.690773060613886, pvalue=2.6831159908772832e-09)

Influencia del Respuesta campana sobre Las Ultima compra

F_onewayResult(statistic=91.73834607514921, pvalue=2.5203765627580955e-21)

Se observa que el nivel de Respuesta positiva o no a la campaña influye en las visitas a las paginas web. pv 0.85 >= 0.05

Así como la Repuesta positiva o no a la campaña influye en el consumo o compras de los clientes. pv 2.68 >= 0.05

De igual forma la Repuesta positiva o no a la campaña influye en los días que tiene una persona de su ultima compra (influye en que vuelva a comprar o compre más seguido) pv 2.52 >= 0.05

Datos e Información sobre la Influencia de la Publicidad y las Decisiones de Compra de los Clientes.

Con respecto a esto se procede a citar algunos aspectos de la influencia de la publicidad y como esta promueve el consumo:

Una de las funciones de la publicidad es **Persuadir**, se crea preferencia por la marca, se debe convencer a los consumidores de que compren en el momento, como también que acepten visitas por el equipo de ventas.

La función de la publicidad no es vender: la publicidad es responsable de crear condiciones para la venta de productos. Para ello, transmite la idea al segmento de mercado objetivo, esperando que pueda coincidir con la idea y tenerla en cuenta a la hora de comprar.

han identificado diferentes factores que afectan la intención de compra entre los cuales se encuentran: la publicidad, el brand equity y el brand engagement. La publicidad actúa como un divulgador de información o de ideas con la intención de cambiar el comportamiento de un grupo de personas para que piense o actúe de determinada manera, también actúa como un diferenciador de una marca frente a la competencia.

Modelo de Machine Learning

```
In [ ]: data3 = data2.iloc[:, [2, 3, 4, 5, 6, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 28, 29]]
```

```
In [ ]: data3.head(3)
```

```
Out[ ]:
```

	Education	Marital_Status	Income	Kidhome	Teenhome	Recency	MntWines	MntFruits	Mntl
0	2	4	58138.0	0	0	58	635	88	
1	2	4	46344.0	1	1	38	11	1	
2	2	5	71613.0	0	0	26	426	49	

```
In [ ]: data3.dtypes
```

```
Out[ ]: Education          int64
Marital_Status          int64
Income                  float64
Kidhome                 int64
Teenhome                int64
Recency                 int64
MntWines                int64
MntFruits               int64
MntMeatProducts         int64
MntFishProducts         int64
MntSweetProducts        int64
MntGoldProds            int64
NumDealsPurchases       int64
NumWebPurchases         int64
NumCatalogPurchases    int64
NumStorePurchases       int64
NumWebVisitsMonth       int64
AcceptedCmp3            int64
AcceptedCmp4            int64
AcceptedCmp5            int64
AcceptedCmp1            int64
AcceptedCmp2            int64
Complain                int64
Response                int64
Age                     int64
dtype: object
```

```
In [ ]: X = data3 # Renombrando variable para utilizarla en Scikit-Learn
```

```
In [ ]: # Normalizando dataframe
scaler = StandardScaler()
X_std = scaler.fit_transform(X)
```

```
In [ ]: # Importando PCA
```

```
pca = PCA()  
pca.fit(X_std)
```

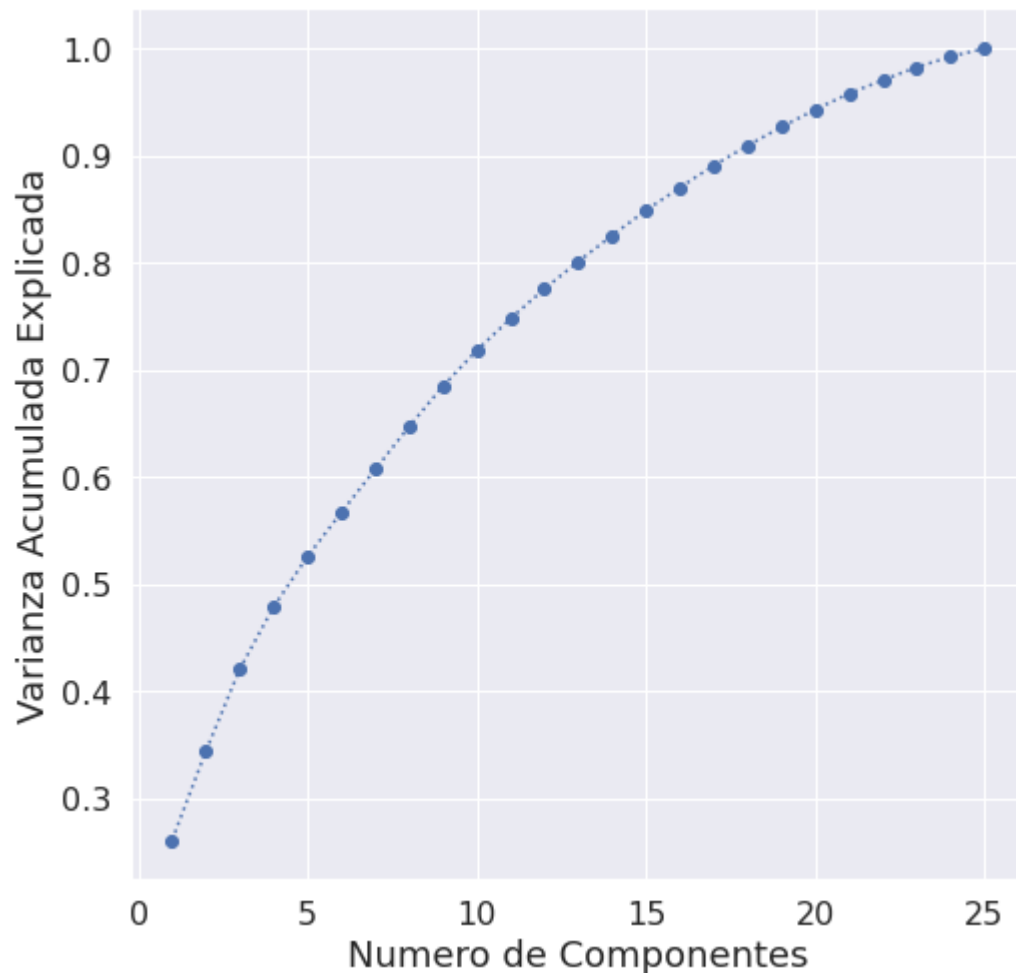
```
Out[ ]: PCA(copy=True, iterated_power='auto', n_components=None, random_state=None,  
          svd_solver='auto', tol=0.0, whiten=False)
```

```
In [ ]: evr = pca.explained_variance_ratio_  
evr
```

```
Out[ ]: array([0.26071465, 0.08314389, 0.07662967, 0.05805226, 0.046901  ,  
              0.04128834, 0.04013911, 0.03993212, 0.03783773, 0.03331178,  
              0.02985145, 0.02775495, 0.02546598, 0.02437783, 0.0234177 ,  
              0.02125227, 0.02067995, 0.01855946, 0.01728875, 0.01629897,  
              0.01506885, 0.01244702, 0.01213377, 0.00975247, 0.00770002])
```

```
In [ ]: # Ploteando grafico de Componentes principales
```

```
fig = plt.figure(figsize=(8,8))  
plt.plot(range(1, len(X.columns)+1), evr.cumsum(), marker='o', linestyle=':')  
plt.xlabel('Numero de Componentes', fontsize=18)  
plt.ylabel('Varianza Acumulada Explicada', fontsize=18)  
plt.xticks(fontsize=16)  
plt.yticks(fontsize=16)  
plt.show()
```



```
In [ ]: # Iteracion para comprobar numero de componentes optimos a utilizar por su nivel de varianza

for i, exp_var in enumerate(evr.cumsum()):
    if exp_var >= 0.8:
        n_comps = i + 1
        break
print("Numero de Componentes Optimos:", n_comps)
pca = PCA(n_components=n_comps)
pca.fit(X_std)
scores_pca = pca.transform(X_std)
```

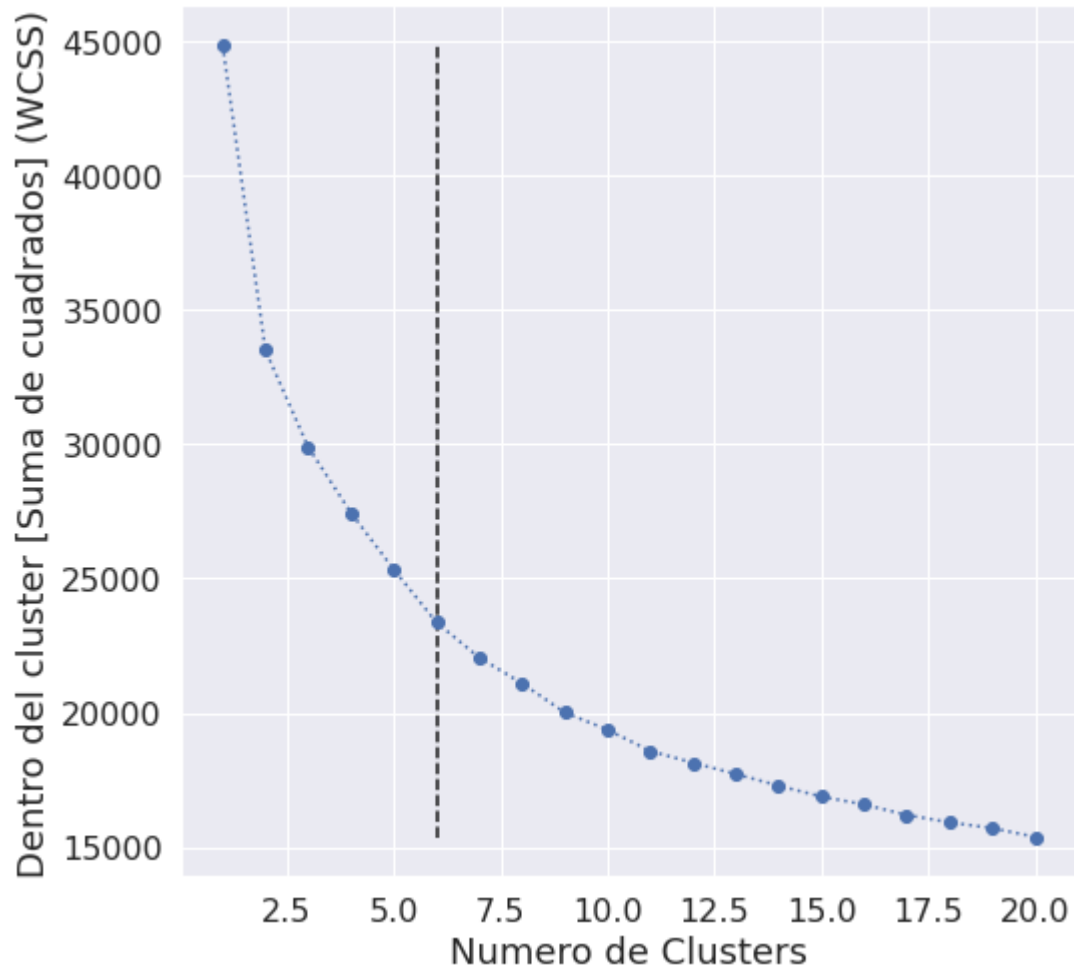
Numero de Componentes Optimos: 13

Algoritmo K-means

```
In [ ]: # Encontrando el punto del codo de la curva de WCSS (dentro de la suma de cuadrados) usando el KneedLocator
wcss = []
max_clusters = 21
for i in range(1, max_clusters):
    kmeans_pca = KMeans(i, init='k-means++', random_state=42)
    kmeans_pca.fit(scores_pca)
    wcss.append(kmeans_pca.inertia_)
n_clusters = KneedLocator([i for i in range(1, max_clusters)], wcss, curve='convex', direction='decreasing').knee
print("Numero de Clusters Optimos:", n_clusters)
```

Numero de Clusters Optimos: 6

```
In [ ]: # Ploteando grafico
fig = plt.figure(figsize=(8,8))
plt.plot(range(1, 21), wcss, marker='o', linestyle=':')
plt.vlines(KneeLocator([i for i in range(1, max_clusters)]), wcss, curve='convex',
           direction='decreasing').knee, ymin=min(wcss), ymax=max(wcss),
           linestyles='dashed')
plt.xlabel('Numero de Clusters', fontsize=18)
plt.ylabel('Dentro del cluster [Suma de cuadrados] (WCSS)', fontsize=18)
plt.xticks(fontsize=16)
plt.yticks(fontsize=16)
plt.show()
```



Analisis y Visualización

```
In [ ]: # Creando la optimizacion de parametros con PCA y K-Means
kmeans_pca = KMeans(n_clusters=n_clusters, init='k-means++', random_state=42)
kmeans_pca.fit(scores_pca);
```

```
In [ ]: # Etiquetando cada uno de los datos dentro del cluster respectivo
df_seg_pca_kmeans = pd.concat([pd.DataFrame(X.reset_index(drop=True)), pd.DataFrame(scores_pca)], axis=1)
df_seg_pca_kmeans.columns.values[(-1*n_comps):] = ["Component " + str(i+1) for i in range(n_comps)]
df_seg_pca_kmeans['Cluster'] = kmeans_pca.labels_
df_seg_pca_kmeans.head()
```

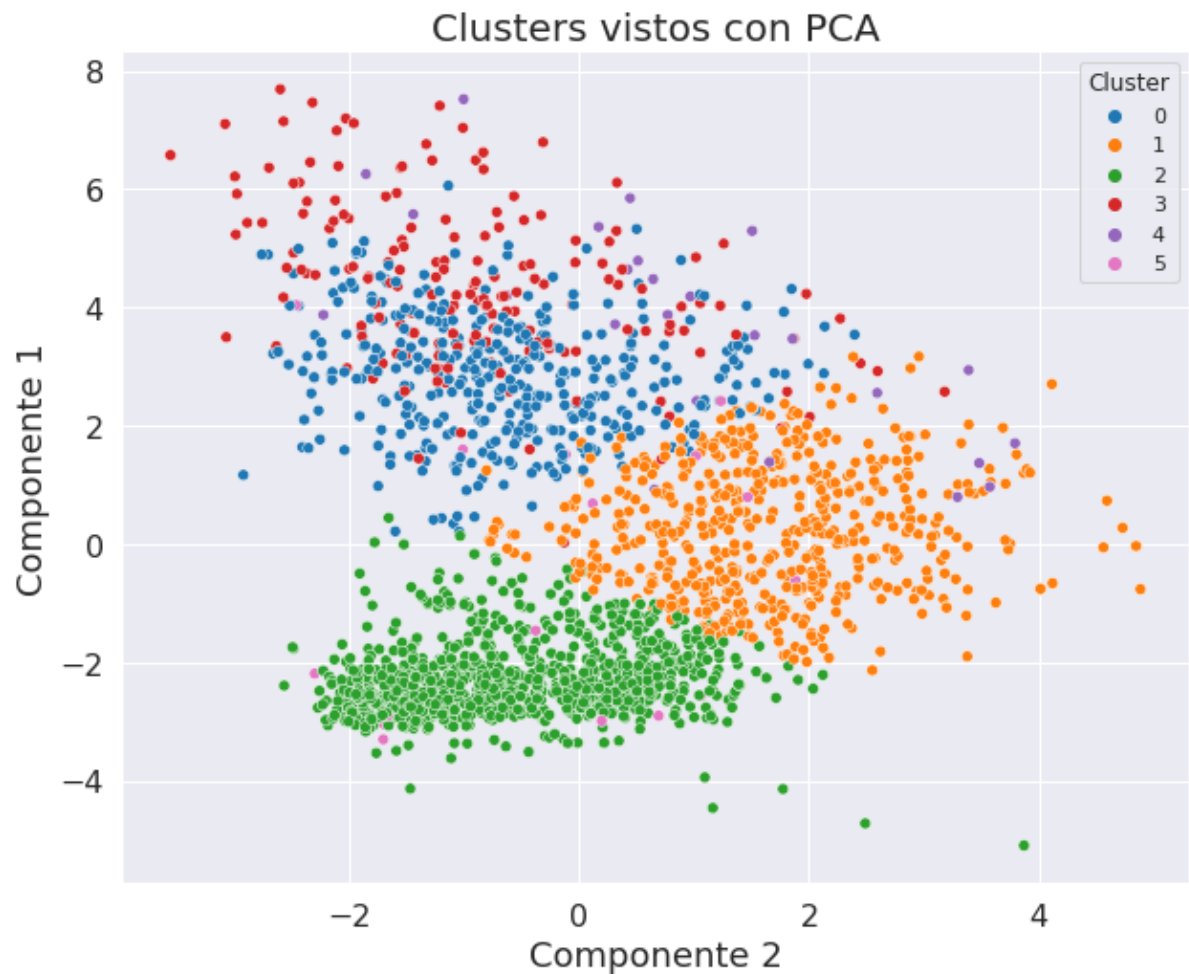
Out[]:

	Education	Marital_Status	Income	Kidhome	Teenhome	Recency	MntWines	MntFruits	Mntl
0	2	4	58138.0	0	0	58	635	88	
1	2	4	46344.0	1	1	38	11	1	
2	2	5	71613.0	0	0	26	426	49	
3	2	5	26646.0	1	0	26	11	4	
4	4	3	58293.0	1	0	94	173	43	

Creando visualizacion de los datos con PCA


```
In [ ]: # Creando visualizacion de los datos con PCA

x = df_seg_pca_kmeans['Component 2']
y = df_seg_pca_kmeans['Component 1']
fig = plt.figure(figsize=(10, 8))
sns.scatterplot(x, y, hue=df_seg_pca_kmeans['Cluster'], palette = ['tab:blue',
'tab:orange', 'tab:green', 'tab:red', 'tab:purple', 'tab:pink'])
plt.title('Clusters vistos con PCA', fontsize=20)
plt.xlabel("Componente 2", fontsize=18)
plt.ylabel("Componente 1", fontsize=18)
plt.xticks(fontsize=16)
plt.yticks(fontsize=16)
plt.show();
```



Prueba del PCA Kmeans con la Columna de Compra Total

```
In [ ]: data2['Total_spend'] = data2['MntWines']+data2['MntFruits']+data2['MntMeatProd
ucts']+data2['MntFishProducts']+data2['MntSweetProducts']+data2['MntGoldProds'
]
# se crea una columna que represente el total de compras.
```

```
In [ ]: data2['Total_Sons'] = data2['Kidhome']+data2['Teenhome']
```

```
In [ ]: data2.head(3)
```

```
Out[ ]:
```

	ID	Year_Birth	Education	Marital_Status	Income	Kidhome	Teenhome	Dt_Customer	Rec
0	5524	1957	2	4	58138.0	0	0	2012-04-09	
1	2174	1954	2	4	46344.0	1	1	2014-08-03	
2	4141	1965	2	5	71613.0	0	0	2013-08-21	

```
In [ ]: data3 = data2.iloc[:, [2,3,4,31,8,15,16,17,18,19,25,28,29,30]]
```

```
In [ ]: data3.head(3)
```

```
Out[ ]:
```

	Education	Marital_Status	Income	Total_Sons	Recency	NumDealsPurchases	NumWebPurch:
0	2	4	58138.0	0	58		3
1	2	4	46344.0	2	38		2
2	2	5	71613.0	0	26		1

```
In [ ]: X = data3 # Renombrando variable para utilizarla en Scikit-Learn
```

```
In [ ]: # Normalizando dataframe
scaler = StandardScaler()
X_std = scaler.fit_transform(X)
```

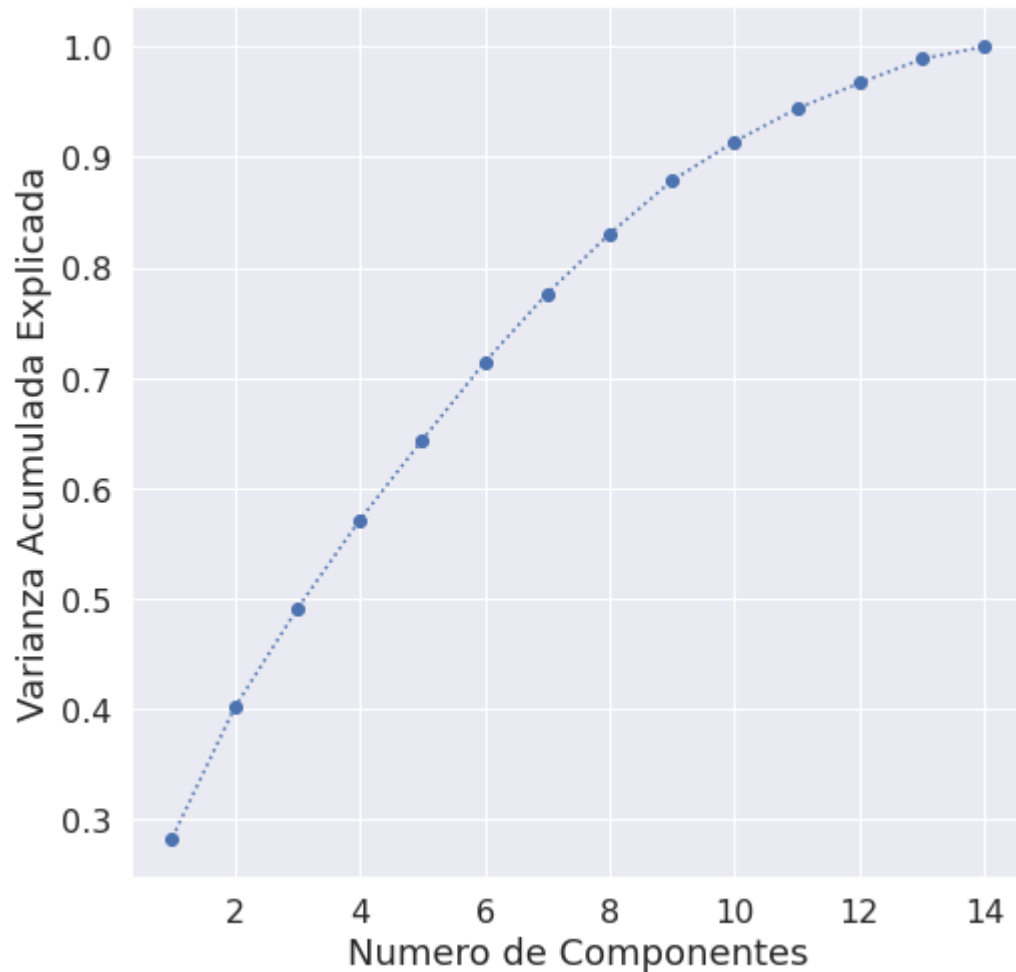
```
In [ ]: # Importando PCA
pca = PCA()
pca.fit(X_std)
```

```
Out[ ]: PCA(copy=True, iterated_power='auto', n_components=None, random_state=None,
        svd_solver='auto', tol=0.0, whiten=False)
```

```
In [ ]: evr = pca.explained_variance_ratio_
evr
```

```
Out[ ]: array([0.28354244, 0.11790765, 0.08929459, 0.08051423, 0.07234054,
               0.07043201, 0.06240103, 0.05387205, 0.04827001, 0.03537724,
               0.02983023, 0.02308819, 0.02174441, 0.01138538])
```

```
In [ ]: # Ploteando grafico de Componentes principales
fig = plt.figure(figsize=(8,8))
plt.plot(range(1, len(X.columns)+1), evr.cumsum(), marker='o', linestyle=':')
plt.xlabel('Numero de Componentes', fontsize=18)
plt.ylabel('Varianza Acumulada Explicada', fontsize=18)
plt.xticks(fontsize=16)
plt.yticks(fontsize=16)
plt.show()
```



```
In [ ]: # Iteracion para comprobar numero de componentes optimos a utilizar por su nivel de varianza

for i, exp_var in enumerate(evr.cumsum()):
    if exp_var >= 0.8:
        n_comps = i + 1
        break
print("Numero de Componentes Optimos:", n_comps)
pca = PCA(n_components=n_comps)
pca.fit(X_std)
scores_pca = pca.transform(X_std)
```

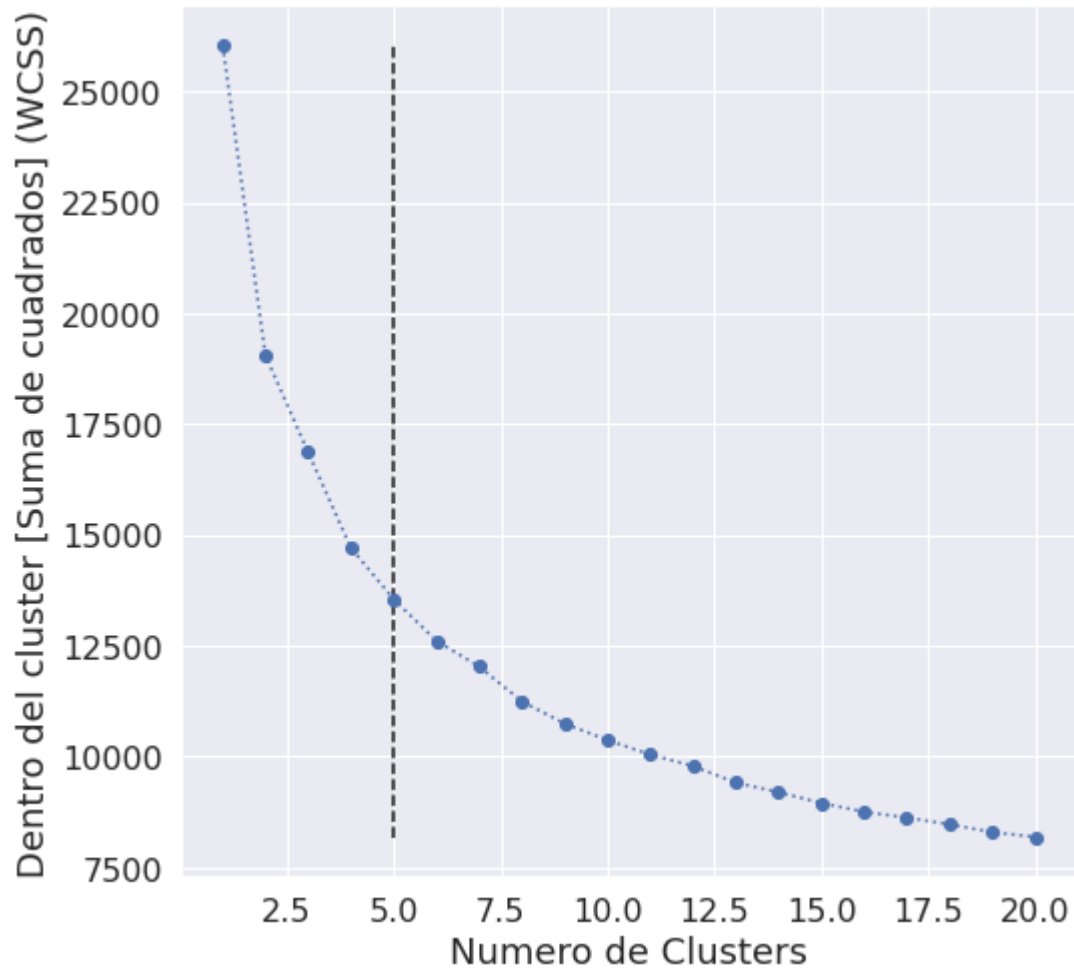
Numero de Componentes Optimos: 8

Algoritmo K-means

```
In [ ]: # Encontrando el punto del codo de la curva de WCSS (dentro de la suma de cuadrados) usando el KneedLocator
wcss = []
max_clusters = 21
for i in range(1, max_clusters):
    kmeans_pca = KMeans(i, init='k-means++', random_state=42)
    kmeans_pca.fit(scores_pca)
    wcss.append(kmeans_pca.inertia_)
n_clusters = KneedLocator([i for i in range(1, max_clusters)], wcss, curve='convex', direction='decreasing').knee
print("Numero de Clusters Optimos:", n_clusters)
```

Numero de Clusters Optimos: 5

```
In [ ]: # Ploteando grafico
fig = plt.figure(figsize=(8,8))
plt.plot(range(1, 21), wcss, marker='o', linestyle=':')
plt.vlines(KneeLocator([i for i in range(1, max_clusters)]), wcss, curve='convex',
            direction='decreasing').knee, ymin=min(wcss), ymax=max(wcss),
            linestyles='dashed')
plt.xlabel('Numero de Clusters', fontsize=18)
plt.ylabel('Dentro del cluster [Suma de cuadrados] (WCSS)', fontsize=18)
plt.xticks(fontsize=16)
plt.yticks(fontsize=16)
plt.show()
```



```
In [ ]: # Creando la optimizacion de parametros con PCA y K-Means
kmeans_pca = KMeans(n_clusters=n_clusters, init='k-means++', random_state=42)
kmeans_pca.fit(scores_pca);
```

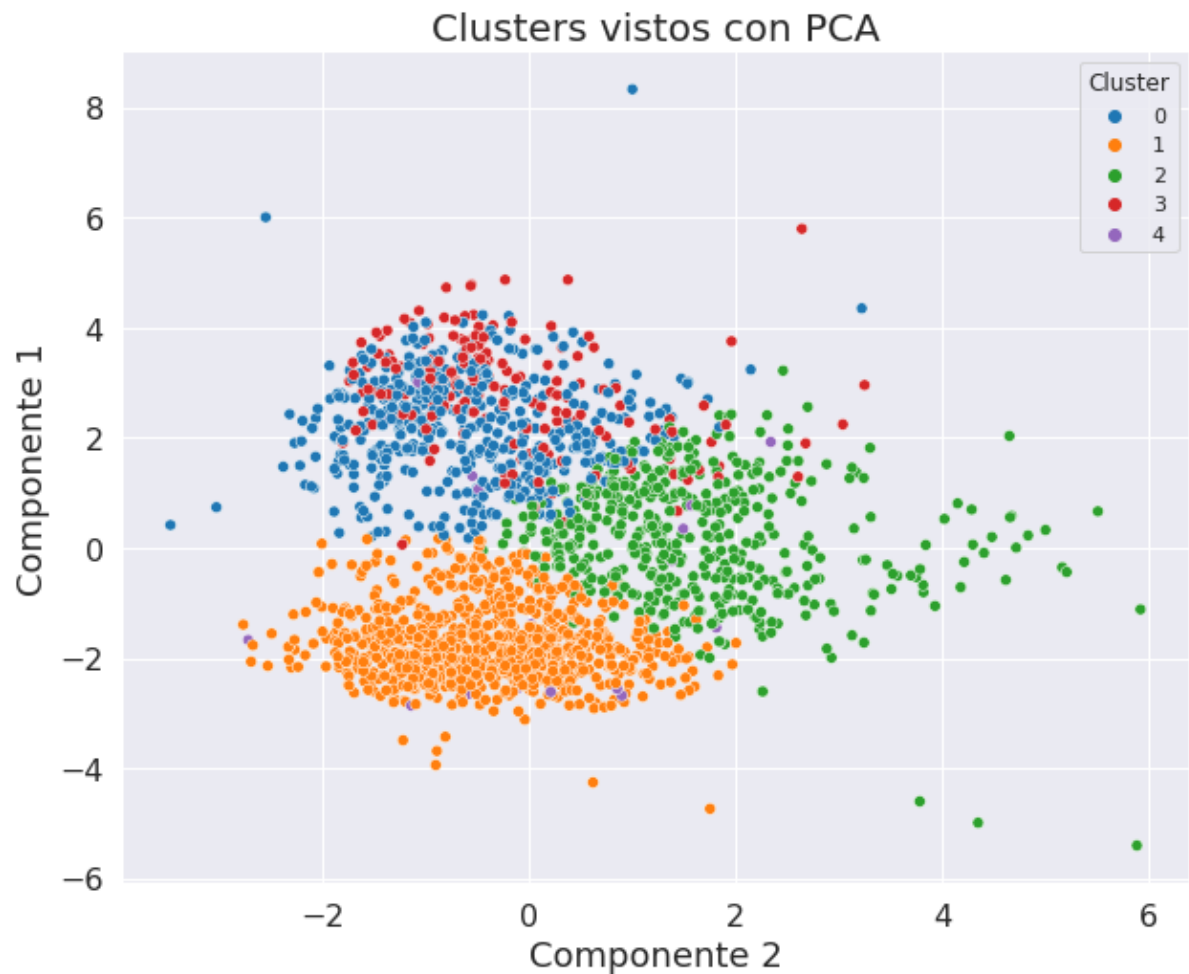
```
In [ ]: # Etiquetando cada uno de los datos dentro del cluster respectivo
df_seg_pca_kmeans = pd.concat([pd.DataFrame(X.reset_index(drop=True)), pd.DataFrame(scores_pca)], axis=1)
df_seg_pca_kmeans.columns.values[(-1*n_comps):] = ["Component " + str(i+1) for i in range(n_comps)]
df_seg_pca_kmeans['Cluster'] = kmeans_pca.labels_
df_seg_pca_kmeans.head()
```

Out[]:

	Education	Marital_Status	Income	Total_Sons	Recency	NumDealsPurchases	NumWebPurchases
0	2	4	58138.0	0	58	3	
1	2	4	46344.0	2	38	2	
2	2	5	71613.0	0	26	1	
3	2	5	26646.0	1	26	2	
4	4	3	58293.0	1	94	5	

```
In [ ]: # Creando visualizacion de los datos con PCA

x = df_seg_pca_kmeans['Component 2']
y = df_seg_pca_kmeans['Component 1']
fig = plt.figure(figsize=(10, 8))
sns.scatterplot(x, y, hue=df_seg_pca_kmeans['Cluster'], palette = ['tab:blue',
'tab:orange', 'tab:green', 'tab:red', 'tab:purple'])
plt.title('Clusters vistos con PCA', fontsize=20)
plt.xlabel("Componente 2", fontsize=18)
plt.ylabel("Componente 1", fontsize=18)
plt.xticks(fontsize=16)
plt.yticks(fontsize=16)
plt.show();
```



Se observa que el componente2 se presenta el cluster 1 desde -2 a 1.8, y en el componenete 1 el Cluster 1, desde -2 a 0.

El cluster 2, en lo que respecta al componente 2 desde 0 a llegar cercananamente a 4, y en lo que respecta al compente1 desde -2 a 2.

Los Clusters 3 y 4 se observan de forma sobrepuestas o mezclada.

Revision de Outliers- Valores Atipicos

```
In [ ]: data2.columns
```

```
Out[ ]: Index(['ID', 'Year_Birth', 'Education', 'Marital_Status', 'Income', 'Kidhome',  
              'Teenhome', 'Dt_Customer', 'Recency', 'MntWines', 'MntFruits',  
              'MntMeatProducts', 'MntFishProducts', 'MntSweetProducts',  
              'MntGoldProds', 'NumDealsPurchases', 'NumWebPurchases',  
              'NumCatalogPurchases', 'NumStorePurchases', 'NumWebVisitsMonth',  
              'AcceptedCmp3', 'AcceptedCmp4', 'AcceptedCmp5', 'AcceptedCmp1',  
              'AcceptedCmp2', 'Complain', 'Z_CostContact', 'Z_Revenue', 'Response',  
              'Age', 'Total_spend', 'Total_Sons'],  
              dtype='object')
```

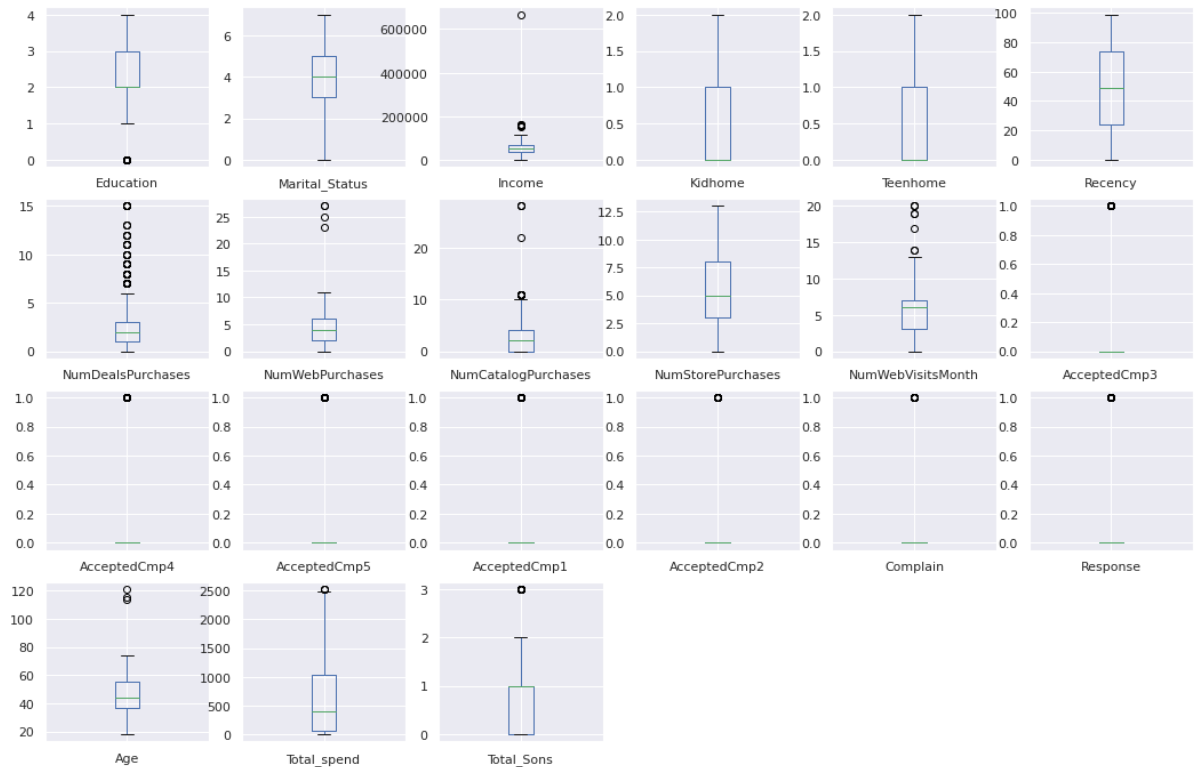
```
In [ ]: data3 = data2.iloc[:, [2, 3, 4, 5, 6, 8, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 28, 29, 30, 31]]  
        # se procede a realizar una seleccion de caracteristicas
```

```
In [ ]: data3.columns
```

```
Out[ ]: Index(['Education', 'Marital_Status', 'Income', 'Kidhome', 'Teenhome',  
              'Recency', 'NumDealsPurchases', 'NumWebPurchases',  
              'NumCatalogPurchases', 'NumStorePurchases', 'NumWebVisitsMonth',  
              'AcceptedCmp3', 'AcceptedCmp4', 'AcceptedCmp5', 'AcceptedCmp1',  
              'AcceptedCmp2', 'Complain', 'Response', 'Age', 'Total_spend',  
              'Total_Sons'],  
              dtype='object')
```

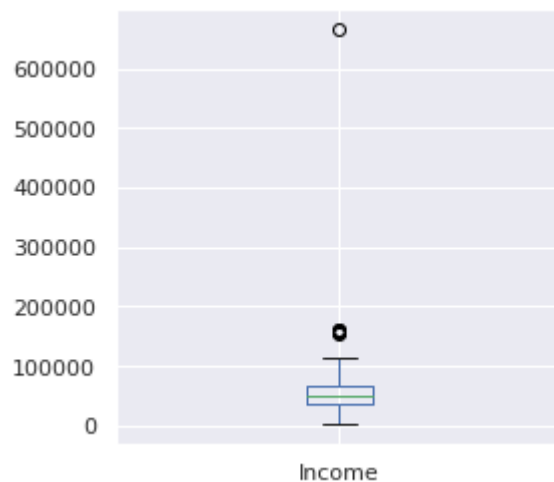


```
In [ ]: data3.plot(kind='box',subplots=True, layout=(6,6), sharex=False, sharey=False)
sns.set(rc={'figure.figsize':(18,18)})
plt.show()    # Representación de las características como vienen en el daset
              original, sin tratamiento de los outliers - atípicos
```



Característica Income

```
In [ ]: data3['Income'].plot(kind='box')
sns.set(rc={'figure.figsize':(4,4)})
plt.show()
```



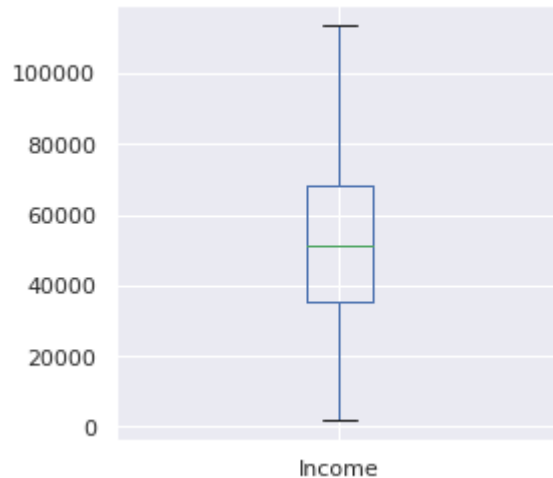
```
In [ ]: quantiles1 = np.percentile(data3['Income'], [25,50,75])
quantiles1
```

```
Out[ ]: array([35538.75, 51381.5 , 68289.75])
```

```
In [ ]: # Analizando desde el punto de dispersion de Los datos
median1 = quantiles1[1]
IQR1 = quantiles1[2]-quantiles1[0]
sigma1 = 0.75*IQR1
```

```
In [ ]: data3 = data3.query("(Income > @median1 - 4*@sigma1) & (Income < @median1 + 4*
@sigma1)")
```

```
In [ ]: data3['Income'].plot(kind='box')
sns.set(rc={'figure.figsize':(4,4)})
plt.show()
```



```
In [ ]: data3.columns
```

```
Out[ ]: Index(['Education', 'Marital_Status', 'Income', 'Kidhome', 'Teenhome',
               'Recency', 'NumDealsPurchases', 'NumWebPurchases',
               'NumCatalogPurchases', 'NumStorePurchases', 'NumWebVisitsMonth',
               'AcceptedCmp3', 'AcceptedCmp4', 'AcceptedCmp5', 'AcceptedCmp1',
               'AcceptedCmp2', 'Complain', 'Response', 'Age', 'Total_spend',
               'Total_Sons'],
              dtype='object')
```

```
In [ ]: data3.head(3)
```

```
Out[ ]:
```

	Education	Marital_Status	Income	Kidhome	Teenhome	Recency	NumDealsPurchases	Num
0	2	4	58138.0	0	0	58	3	
1	2	4	46344.0	1	1	38	2	
2	2	5	71613.0	0	0	26	1	

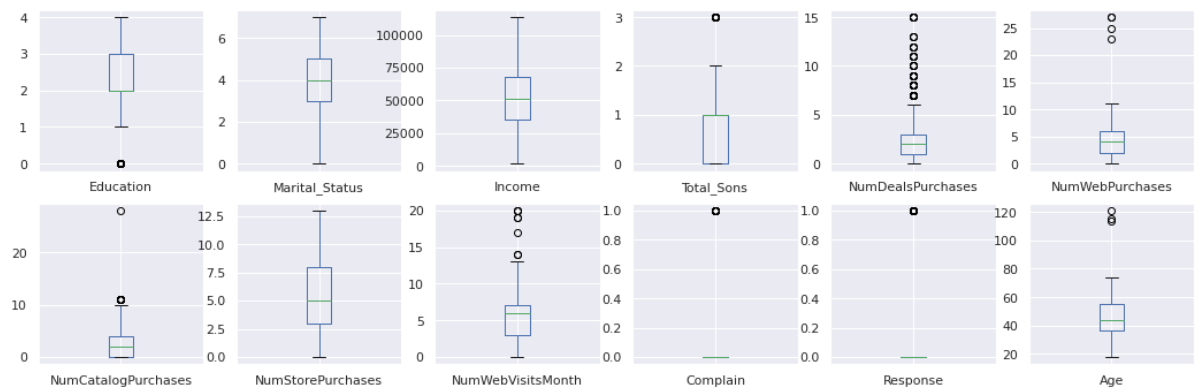
```
In [ ]: data4 = data3.iloc[:,[0,1,2,20,6,7,8,9,10,16,17,18]]
```

```
In [ ]: data4.head(3)
```

```
Out[ ]:
```

	Education	Marital_Status	Income	Total_Sons	NumDealsPurchases	NumWebPurchases	Nurr
0	2	4	58138.0	0	3	8	
1	2	4	46344.0	2	2	1	
2	2	5	71613.0	0	1	8	

```
In [ ]: data4.plot(kind='box',subplots=True, layout=(6,6), sharex=False, sharey=False)
sns.set(rc={'figure.figsize':(18,18)})
plt.show()
```



Prueba de Clusters despues de Limpiar Outliers

```
In [ ]: data4.head(3)
```

```
Out[ ]:
```

	Education	Marital_Status	Income	Total_Sons	NumDealsPurchases	NumWebPurchases	Nurr
0	2	4	58138.0	0	3	8	
1	2	4	46344.0	2	2	1	
2	2	5	71613.0	0	1	8	

```
In [ ]: X = data4 # Renombrando variable para utilizarla en Scikit-Learn
```

```
In [ ]: # Normalizando dataframe
scaler = StandardScaler()
X_std = scaler.fit_transform(X)
```

```
In [ ]: # Importando PCA
```

```
pca = PCA()  
pca.fit(X_std)
```

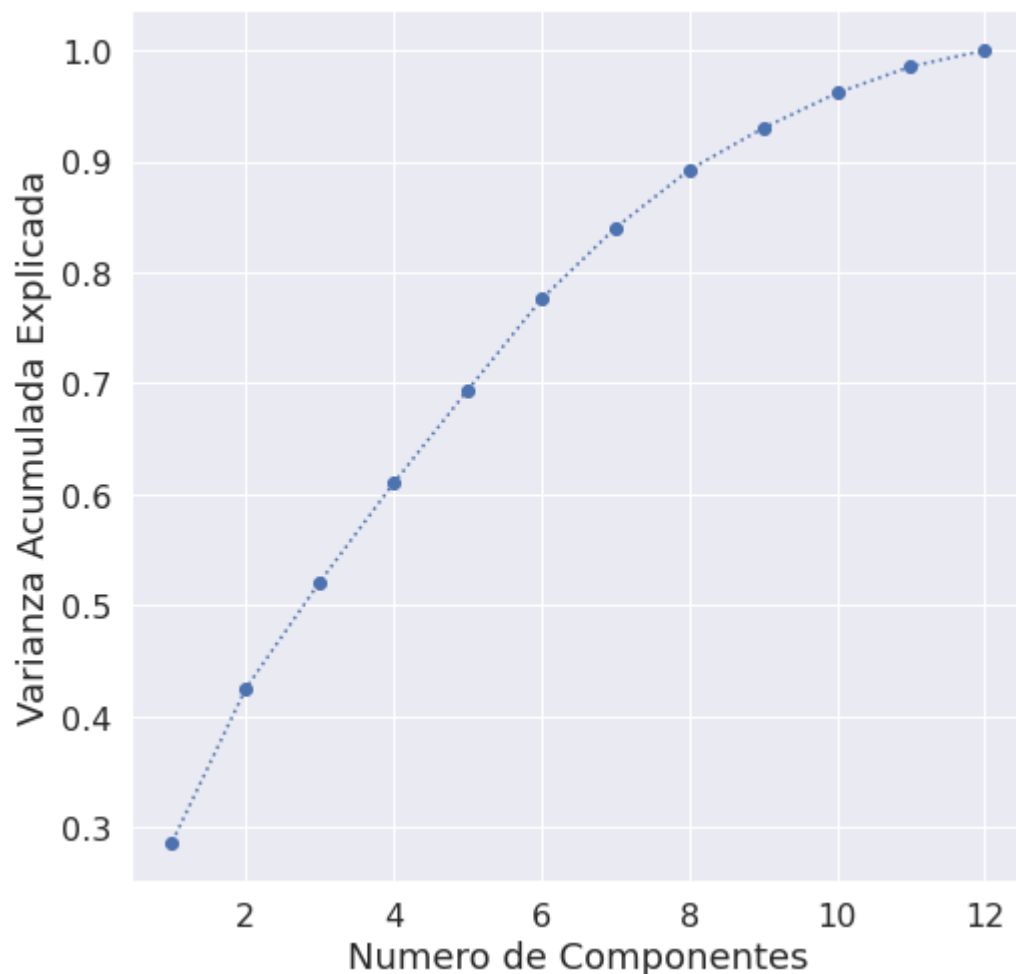
```
Out[ ]: PCA(copy=True, iterated_power='auto', n_components=None, random_state=None,  
          svd_solver='auto', tol=0.0, whiten=False)
```

```
In [ ]: evr = pca.explained_variance_ratio_  
evr
```

```
Out[ ]: array([0.2868485 , 0.13852953, 0.09579883, 0.08901062, 0.08359502,  
              0.08211277, 0.06410803, 0.05230884, 0.03762678, 0.03161737,  
              0.02395371, 0.01448999])
```

```
In [ ]: # Ploteando grafico de Componentes principales
```

```
fig = plt.figure(figsize=(8,8))  
plt.plot(range(1, len(X.columns)+1), evr.cumsum(), marker='o', linestyle=':')  
plt.xlabel('Numero de Componentes', fontsize=18)  
plt.ylabel('Varianza Acumulada Explicada', fontsize=18)  
plt.xticks(fontsize=16)  
plt.yticks(fontsize=16)  
plt.show()
```



```
In [ ]: # Iteracion para comprobar numero de componentes optimos a utilizar por su nivel de varianza

for i, exp_var in enumerate(evr.cumsum()):
    if exp_var >= 0.8:
        n_comps = i + 1
        break
print("Numero de Componentes Optimos:", n_comps)
pca = PCA(n_components=n_comps)
pca.fit(X_std)
scores_pca = pca.transform(X_std)
```

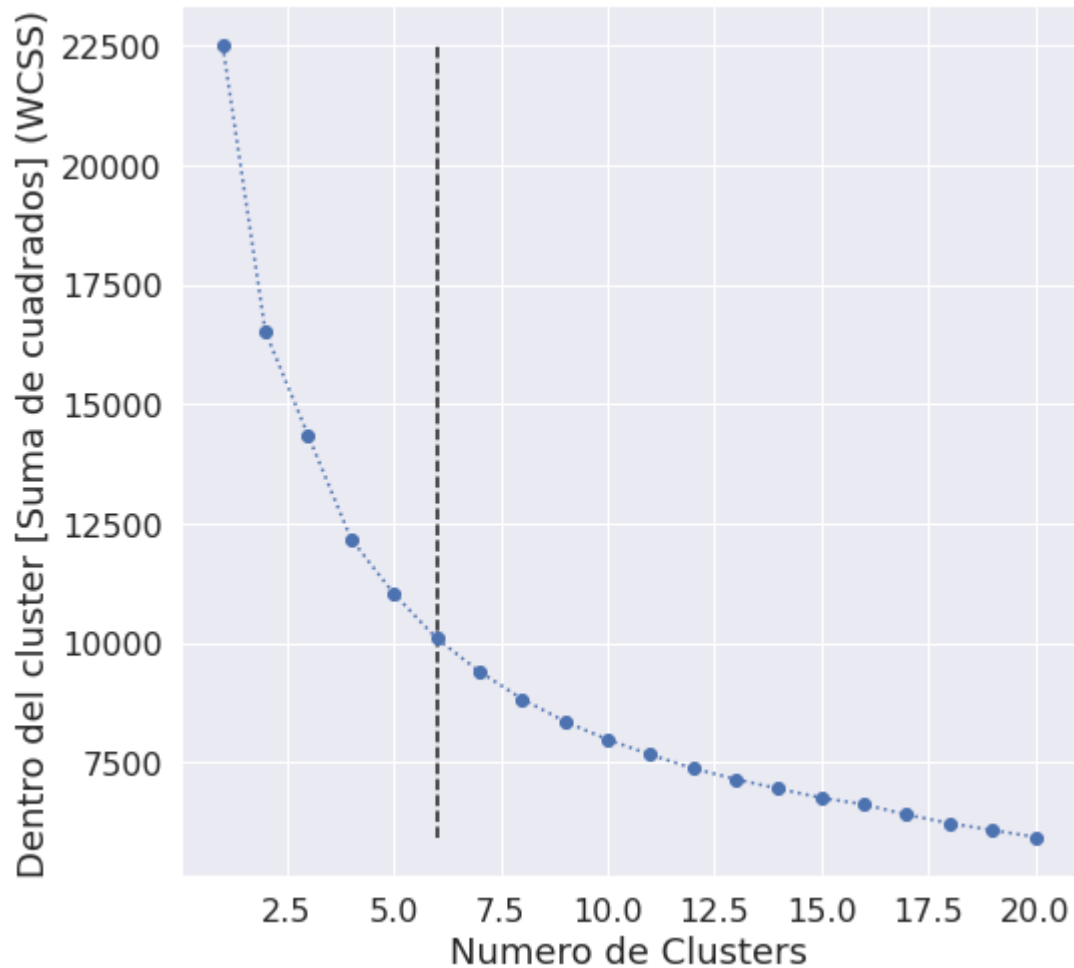
Numero de Componentes Optimos: 7

Algoritmo K-means

```
In [ ]: # Encontrando el punto del codo de la curva de WCSS (dentro de la suma de cuadrados) usando el KneedLocator
wcss = []
max_clusters = 21
for i in range(1, max_clusters):
    kmeans_pca = KMeans(i, init='k-means++', random_state=42)
    kmeans_pca.fit(scores_pca)
    wcss.append(kmeans_pca.inertia_)
n_clusters = KneedLocator([i for i in range(1, max_clusters)], wcss, curve='convex', direction='decreasing').knee
print("Numero de Clusters Optimos:", n_clusters)
```

Numero de Clusters Optimos: 6

```
In [ ]: # Ploteando grafico
fig = plt.figure(figsize=(8,8))
plt.plot(range(1, 21), wcss, marker='o', linestyle=':')
plt.vlines(KneeLocator([i for i in range(1, max_clusters)]), wcss, curve='convex',
            direction='decreasing').knee, ymin=min(wcss), ymax=max(wcss),
            linestyles='dashed')
plt.xlabel('Numero de Clusters', fontsize=18)
plt.ylabel('Dentro del cluster [Suma de cuadrados] (WCSS)', fontsize=18)
plt.xticks(fontsize=16)
plt.yticks(fontsize=16)
plt.show()
```



```
In [ ]: # Creando la optimizacion de parametros con PCA y K-Means
kmeans_pca = KMeans(n_clusters=n_clusters, init='k-means++', random_state=42)
kmeans_pca.fit(scores_pca);
```

```
In [ ]: # Etiquetando cada uno de los datos dentro del cluster respectivo
df_seg_pca_kmeans = pd.concat([pd.DataFrame(X.reset_index(drop=True)), pd.DataFrame(scores_pca)], axis=1)
df_seg_pca_kmeans.columns.values[(-1*n_comps):] = ["Component " + str(i+1) for i in range(n_comps)]
df_seg_pca_kmeans['Cluster'] = kmeans_pca.labels_
df_seg_pca_kmeans.head()
```

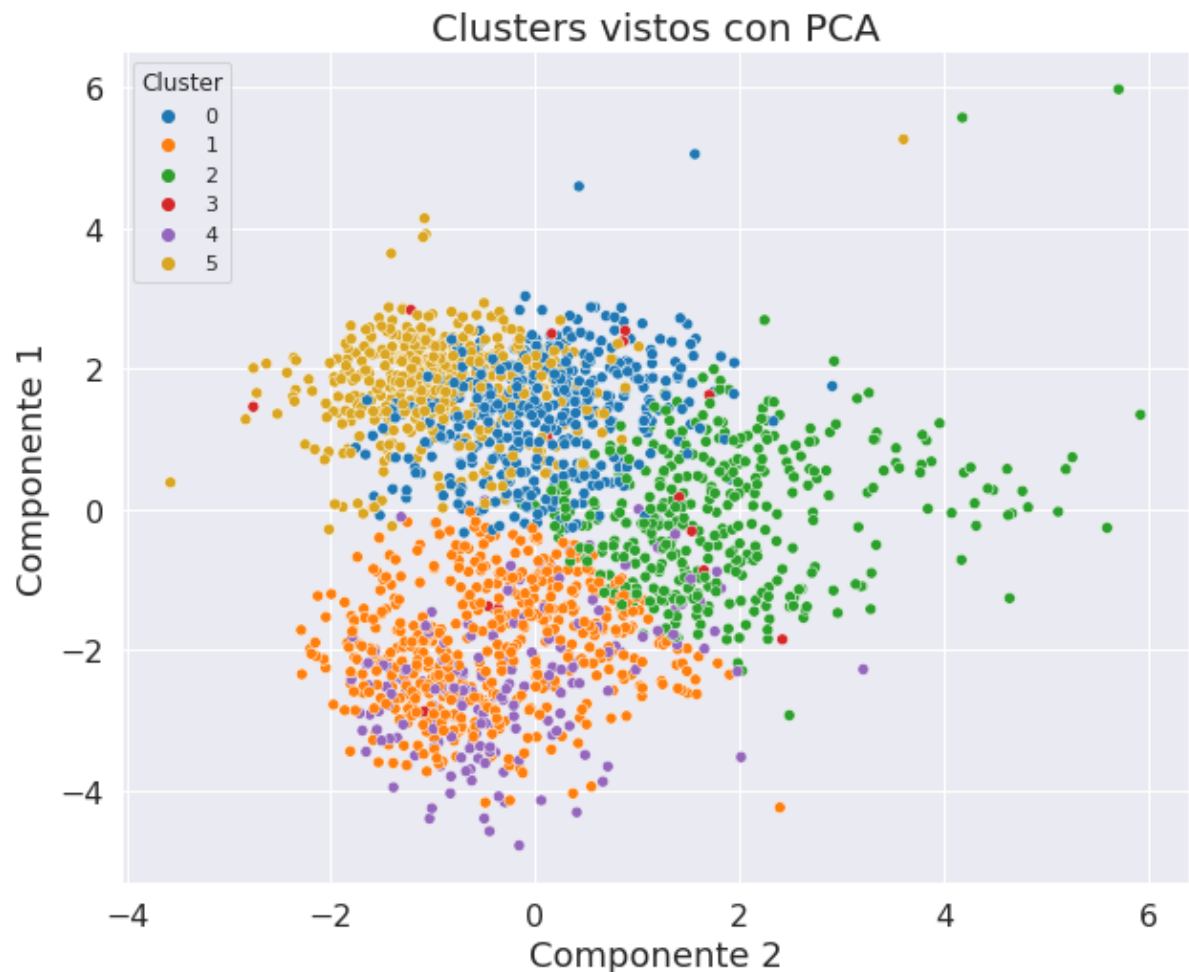
Out[]:

	Education	Marital_Status	Income	Total_Sons	NumDealsPurchases	NumWebPurchases	Nurr
0	2	4	58138.0	0	3	8	
1	2	4	46344.0	2	2	1	
2	2	5	71613.0	0	1	8	
3	2	5	26646.0	1	2	2	
4	4	3	58293.0	1	5	5	

Creando visualizacion de los datos con PCA

```
In [ ]: # Creando visualizacion de los datos con PCA

x = df_seg_pca_kmeans['Component 2']
y = df_seg_pca_kmeans['Component 1']
fig = plt.figure(figsize=(10, 8))
sns.scatterplot(x, y, hue=df_seg_pca_kmeans['Cluster'], palette = ['tab:blue',
'tab:orange', 'tab:green', 'tab:red', 'tab:purple', 'goldenrod'])
plt.title('Clusters vistos con PCA', fontsize=20)
plt.xlabel("Componente 2", fontsize=18)
plt.ylabel("Componente 1", fontsize=18)
plt.xticks(fontsize=16)
plt.yticks(fontsize=16)
plt.show();
```



```
In [ ]: # Los componentes 1 y 2
# el cluster 0 se encuentra en medio del cluster 1 y 2 , y el cluster 0 se encue
ntra en el componente2 entre -1 a 1.8, y en el componente1 entre 0 a 3
# el Cluster 1 se encuentra en el componente2 desde -2 a 1.5, y en el componen
te1 desde -3 a 0 .
# el cluster 2 se encuentra en el componente2 desde 0 a 3 (alrededor) de forma
dispersa
# el cluster 3 no se observa
# el cluster 4 se encuentra disperso entre el cluster 1
# el cluster 5 se encuentra en el componente 2 entre -2 a 0, y en el component
e1 desde 1 a 3.
```



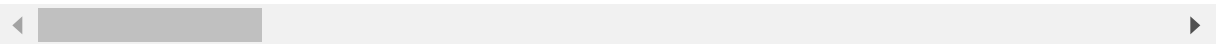
```
In [ ]: # Marcando cada uno de los datos con su respectivo cluster en el dataset original

data1['Cluster'] = df_seg_pca_kmeans['Cluster']
data1
```

Out[]:

	ID	Year_Birth	Education	Marital_Status	Income	Kidhome	Teenhome	Dt_Customer
0	5524	1957	Graduation	Single	58138.0	0	0	2012-04-09
1	2174	1954	Graduation	Single	46344.0	1	1	2014-08-03
2	4141	1965	Graduation	Together	71613.0	0	0	2013-08-21
3	6182	1984	Graduation	Together	26646.0	1	0	2014-10-02
4	5324	1981	PhD	Married	58293.0	1	0	2014-01-19
...
2235	10870	1967	Graduation	Married	61223.0	0	1	2013-06-13
2236	4001	1946	PhD	Together	64014.0	2	1	2014-10-06
2237	7270	1981	Graduation	Divorced	56981.0	0	0	2014-01-25
2238	8235	1956	Master	Together	69245.0	0	1	2014-01-24
2239	9405	1954	PhD	Married	52869.0	1	1	2012-10-15

2240 rows × 35 columns



```
In [ ]: # Comprobando cantidad de datos en cada Cluster

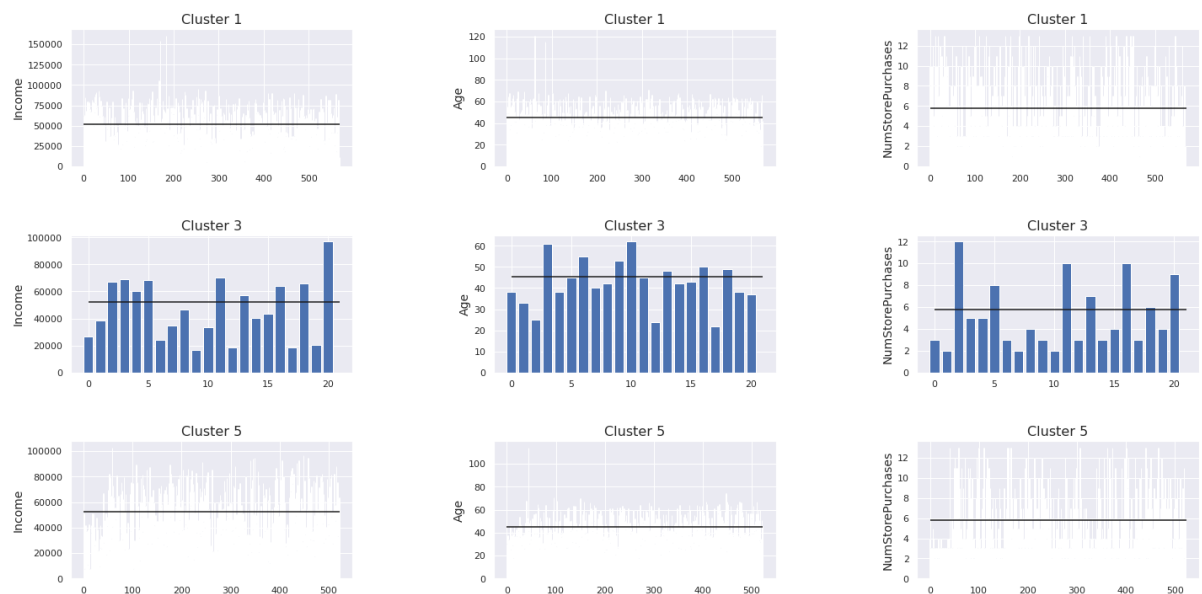
pd.value_counts(data1['Cluster'])
```

```
Out[ ]: 1.0    569
        5.0    523
        0.0    500
        2.0    424
        4.0    195
        3.0     21
        Name: Cluster, dtype: int64
```

```
In [ ]: # Visualizando características generales de cada cluster
```

```
clusters = [1, 3, 5]
features = ["Income", "Age", "NumStorePurchases"]
#colors = ['tab:green', 'tab:olive', 'tab:cyan']
dim = len(clusters)

fig, axes = plt.subplots(dim, dim, figsize=(24, 12))
i = 0
test_cluster = data1.loc[data1['Cluster'] == clusters[0]]
for ax in axes.flatten():
    if i % dim == 0 and i != 0:
        test_cluster = data1.loc[data1['Cluster'] == clusters[i // dim]]
    col = features[i % dim]
    y = test_cluster[col]
    x = [i for i in range(len(y))]
    ax.bar(x, y) #colors[i//dim]
    ax.set_ylabel(col, fontsize=14)
    ax.set_title("Cluster " + str(clusters[i // dim]), fontsize=16)
    ax.hlines(np.mean(data1[col]), 0, len(y))
    plt.subplots_adjust(wspace=.5, hspace=.5)
    i += 1
```



```
In [ ]: # se obser el el cuadro anterior que en el Cluster 3 parece estar uno de los grupos de mayor nivel de ingreso, en comparación con cluster 5 y Cluster1
# se puede ver que el cluster 3 indica tener el grupo de clientes de mayor edad en comparación a los cluster 5 y cluster 1
# en lo que respecta al número de compras en las tiendas (NumStorePurchases) se observan en comportamiento similar en los clusters.
```

```
In [ ]: # Visualizando datos de cada Cluster a nivel de filtrado
```

```
preview = data1.loc[data1['Cluster'] == 5]  
print(len(preview))  
preview.head(n=30)
```

```
# se observa la distribucion de las características según el cluster5, en donde se pueden ver los valores que poseen cada una según se integran al cluster.  
# el cluster5 presenta la información del comportamiento de compra de aquellos que el algoritmo los identifica como tales.
```

Out[]:

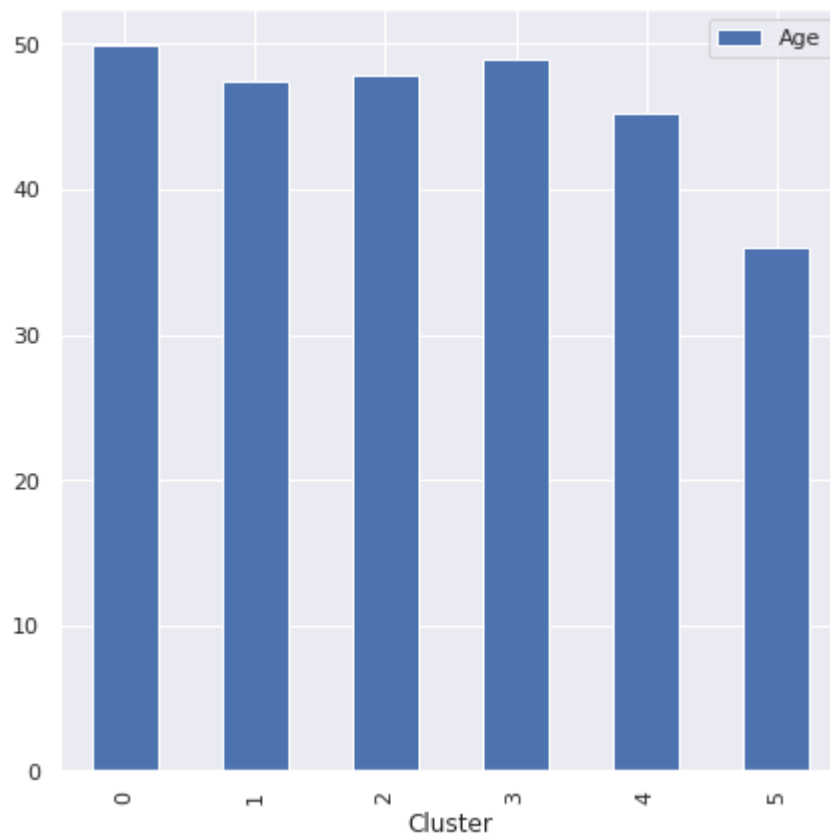
	ID	Year_Birth	Education	Marital_Status	Income	Kidhome	Teenhome	Dt_Customer
3	6182	1984	Graduation	Together	26646.0	1	0	2014-10-02
10	1994	1983	Graduation	Married	51381.5	1	0	2013-11-15
11	387	1976	Basic	Married	7500.0	0	0	2012-11-13
14	2569	1987	Graduation	Married	17323.0	0	0	2012-10-10
16	9736	1980	Graduation	Married	41850.0	1	1	2012-12-24
19	2278	1985	2n Cycle	Single	33812.0	1	0	2012-03-11
20	9360	1982	Graduation	Married	37040.0	0	0	2012-08-08
25	7892	1969	Graduation	Single	18589.0	0	0	2013-02-01
28	9422	1989	Graduation	Married	38360.0	1	0	2013-05-31
30	6864	1989	Master	Divorced	10979.0	0	0	2014-05-22
37	10755	1976	2n Cycle	Married	23718.0	1	0	2013-02-09
38	8595	1973	Graduation	Widow	42429.0	0	1	2014-11-02
41	503	1985	Master	Married	20559.0	1	0	2013-12-03
44	2139	1975	Master	Married	7500.0	1	0	2013-02-10
46	9909	1996	2n Cycle	Married	7500.0	0	0	2012-09-11
47	7286	1968	Graduation	Together	41728.0	1	0	2013-05-24
52	1331	1977	Graduation	Single	35790.0	1	0	2013-02-01
57	7437	1988	Graduation	Single	27938.0	1	0	2014-04-28
58	8557	1982	Graduation	Single	51381.5	1	0	2013-06-17
65	8082	1971	Graduation	Married	25721.0	1	0	2013-05-21
71	10629	1973	2n Cycle	Married	51381.5	1	0	2012-09-14
75	5846	1977	Graduation	Divorced	40246.0	1	0	2012-12-19
80	3332	1985	Graduation	Single	29760.0	1	0	2012-08-29
81	2261	1969	Graduation	Married	26304.0	1	0	2013-06-23
82	5346	1973	2n Cycle	Married	23559.0	1	0	2013-03-07
95	7516	1983	Graduation	Married	30096.0	1	0	2014-05-22
100	1473	1960	2n Cycle	Single	47823.0	0	1	2013-07-23
105	8373	1979	Basic	Together	24594.0	1	0	2013-10-12
108	3629	1978	Graduation	Single	38557.0	1	0	2012-12-19
118	5234	1967	2n Cycle	Together	30753.0	1	1	2013-11-07

```
In [ ]: round(df_seg_pca_kmeans.pivot_table('Age', index= 'Cluster', aggfunc= 'mean', fill_value=0),2)
```

Out[]:

	Age
Cluster	
0	49.93
1	47.44
2	47.88
3	48.90
4	45.19
5	35.96

```
In [ ]: round(df_seg_pca_kmeans.pivot_table('Age', index= 'Cluster', aggfunc= 'mean', fill_value=0),2).plot(kind = 'bar')
sns.set(rc={'figure.figsize':(7,7)})
plt.show()
```



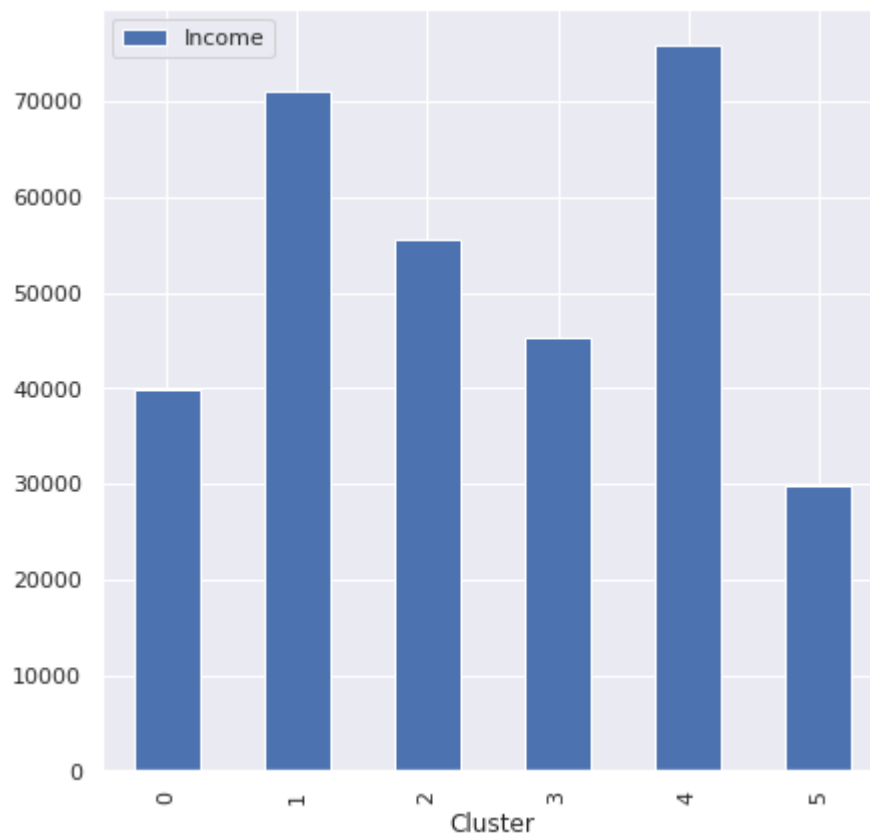
```
In [ ]: # Del gráfico anterior se puede ver que las edades entre los primeros 4 clusters son muy similares, mientras que los últimos dos clusters tienen a los clientes más jóvenes
```

```
In [ ]: round(df_seg_pca_kmeans.pivot_table('Income', index= 'Cluster', aggfunc= 'mean',fill_value=0),2)
```

Out[]:

Income	
Cluster	
0	39826.32
1	70963.72
2	55645.15
3	45242.29
4	75879.98
5	29844.13

```
In [ ]: round(df_seg_pca_kmeans.pivot_table('Income', index= 'Cluster', aggfunc= 'mean',fill_value=0),2).plot(kind = 'bar')
sns.set(rc={'figure.figsize':(7,7)})
plt.show()
```



```
In [ ]: # según se puede ver el cluster 1 y 4 poseen los clientes de mayor ingreso
```

Prueba Modelo Supervisado

Se realiza una prueba para determinar si algunas variables que se utilien pueden predecir si un cliente puede o no reaccionar positivamente a las campañas.


```
In [ ]: data3.columns
```

```
Out[ ]: Index(['Education', 'Marital_Status', 'Income', 'Kidhome', 'Teenhome',  
              'Recency', 'NumDealsPurchases', 'NumWebPurchases',  
              'NumCatalogPurchases', 'NumStorePurchases', 'NumWebVisitsMonth',  
              'AcceptedCmp3', 'AcceptedCmp4', 'AcceptedCmp5', 'AcceptedCmp1',  
              'AcceptedCmp2', 'Complain', 'Response', 'Age', 'Total_spend',  
              'Total_Sons'],  
             dtype='object')
```

```
In [ ]: data3.head(3)
```

```
Out[ ]:
```

	Education	Marital_Status	Income	Kidhome	Teenhome	Recency	NumDealsPurchases	Num
0	2	4	58138.0	0	0	58		3
1	2	4	46344.0	1	1	38		2
2	2	5	71613.0	0	0	26		1

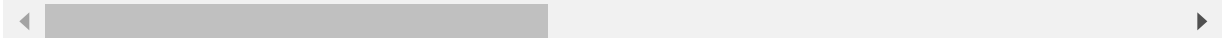


```
In [ ]: data5 = data3.iloc[:,[0,1,2,3,4,5,6,7,8,9,10,11,16,17,18]]
```

```
In [ ]: data5.head(3)
```

```
Out[ ]:
```

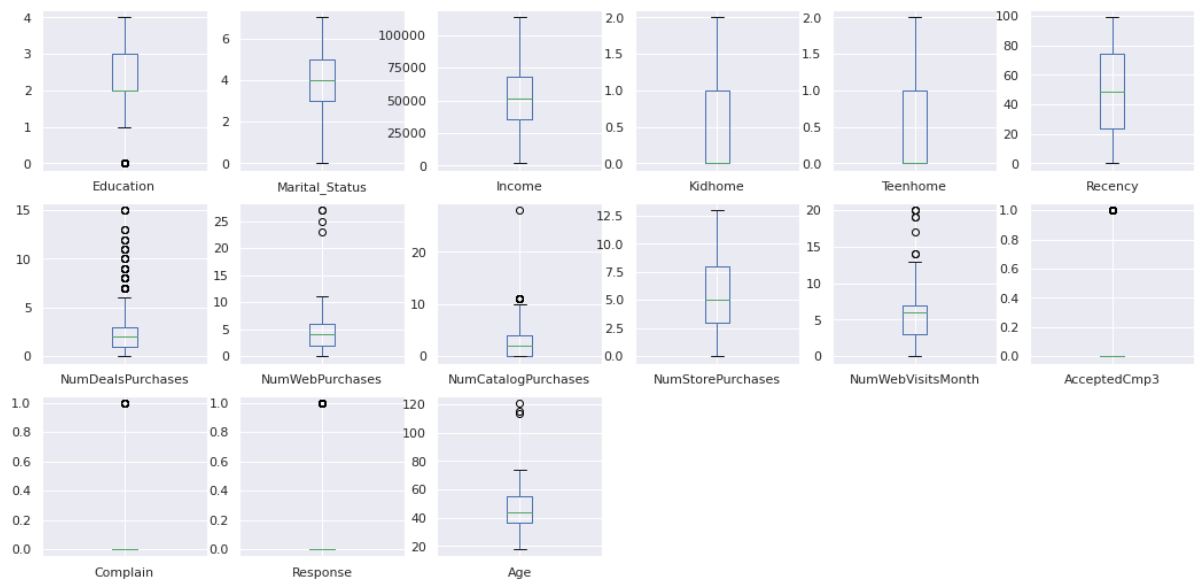
	Education	Marital_Status	Income	Kidhome	Teenhome	Recency	NumDealsPurchases	Num
0	2	4	58138.0	0	0	58		3
1	2	4	46344.0	1	1	38		2
2	2	5	71613.0	0	0	26		1



```
In [ ]: data5.columns
```

```
Out[ ]: Index(['Education', 'Marital_Status', 'Income', 'Kidhome', 'Teenhome',  
              'Recency', 'NumDealsPurchases', 'NumWebPurchases',  
              'NumCatalogPurchases', 'NumStorePurchases', 'NumWebVisitsMonth',  
              'AcceptedCmp3', 'Complain', 'Response', 'Age'],  
             dtype='object')
```

```
In [ ]: data5.plot(kind='box',subplots=True, layout=(6,6), sharex=False, sharey=False)
sns.set(rc={'figure.figsize':(18,18)})
plt.show()
```



```
In [ ]: data5.iloc[:,[0,1,2,3,4,5,11,12,13,14]]
```

Out[]:

	Education	Marital_Status	Income	Kidhome	Teenhome	Recency	AcceptedCmp3	Compl:
0	2	4	58138.0	0	0	58	0	
1	2	4	46344.0	1	1	38	0	
2	2	5	71613.0	0	0	26	0	
3	2	5	26646.0	1	0	26	0	
4	4	3	58293.0	1	0	94	0	
...	
2235	2	3	61223.0	0	1	46	0	
2236	4	5	64014.0	2	1	56	0	
2237	2	2	56981.0	0	0	91	0	
2238	3	5	69245.0	0	1	8	0	
2239	4	3	52869.0	1	1	40	0	

2232 rows × 10 columns




```
In [ ]: # Importando bibliotecas
from pandas import read_csv
import pandas as pd #manejo y estructuracion de datos y su manipulación
from pandas.plotting import scatter_matrix #diagramas de correlación
from matplotlib import pyplot #Hacer gráficos en python
from sklearn.model_selection import train_test_split #Lograr dividir las muestras
from sklearn.model_selection import cross_val_score #validación cruzada score
from sklearn.model_selection import StratifiedKFold #validación cruzada
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix #matriz de confusión
from sklearn.metrics import accuracy_score #score de validación cruzada
```

```
In [ ]: # Modelos de ML con que se va a trabajar
from sklearn.metrics import accuracy_score #score de validación cruzada
from sklearn.linear_model import LogisticRegression #regresión logística
from sklearn.tree import DecisionTreeClassifier #árboles de decisión
from sklearn.neighbors import KNeighborsClassifier #KNN
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis #Análisis discriminante lineal
from sklearn.naive_bayes import GaussianNB #Gauss Bayesiana
from sklearn.svm import SVC # Maquinas de Soporte Vectorial
from sklearn.model_selection import train_test_split
```

```
In [ ]: # Conjunto de datos de validación dividida
# Con el 80% se crea el modelo y con el 20% se entrena

array = data5.values #Los datos ahora se transforman en un arreglo

X = array[:,[0,1,2,3,4,5,11,12,14]] # se toman los datos, sin la clase de clasificación
y = array[:,13] # se toman los datos

# Se dividen los datos en conjunto de entrenamiento y prueba, se utiliza random_state = 0 para que no dé
# resultados diferentes si se vuelve a correr.
X_train, X_test, Y_train, Y_test = train_test_split(X, y, test_size=0.20, random_state=0)
```

```
In [ ]: # Algoritmos de Comprobación, se guardan en una lista
models = []
models.append(('LR', LogisticRegression(solver='liblinear', multi_class='ovr')))
models.append(('LDA', LinearDiscriminantAnalysis()))
models.append(('KNN', KNeighborsClassifier()))
models.append(('CART', DecisionTreeClassifier()))
models.append(('NB', GaussianNB()))
models.append(('SVC', SVC()))
```

```
In [ ]: #Se debe seleccionar el mejor modelo, ya que ahora se tienen 6 modelos y estimaciones de precisión para cada uno,
#por ello se necesita comparar los modelos entre sí y seleccionar el más preciso.
import warnings
warnings.filterwarnings('ignore')

resultados = []
names = []

# Si se necesita tanto el índice o nombre, así como el elemento, se usa for in
# dice, elemento en lista
for name, model in models:
    kfold = StratifiedKFold(n_splits=10, random_state=1, shuffle=True) # Declaración de la validación cruzada, las características
    cv_resultados = cross_val_score(model, X_train, Y_train, cv=kfold, scoring='accuracy') # genera la precisión de la validación cruzada y la guarda en la variable cv_resultados en lista
    resultados.append(cv_resultados) # genera la precisión de la validación cruzada y la guarda en la variable cv_resultados en matrices, esto para hacer el boxplot.
    names.append(name) # names en matrices
    print('%s: %f (%f)' % (name, cv_resultados.mean(), cv_resultados.std()))

LR: 0.845377 (0.005149)
LDA: 0.853198 (0.021976)
KNN: 0.825755 (0.019314)
CART: 0.797728 (0.030312)
NB: 0.845936 (0.007645)
SVC: 0.842019 (0.002051)
```

```
In [ ]: # Haciendo predicciones y evaluación del dataset

model = LinearDiscriminantAnalysis()
model.fit(X_train, Y_train)
prediccion = model.predict(X_test)
```

```
In [ ]: mc =pd.DataFrame(confusion_matrix(Y_test, prediccion, labels=[0,1]),
                        index = [0,1],
                        columns = [0,1])

# Evaluando Predicciones
print("ROC:", accuracy_score(Y_test, prediccion),sep='\n')
print("")
print("Matriz de Confusión:", mc,sep='\n')
```

ROC:
0.8769574944071589

Matriz de Confusión:

	0	1
0	380	15
1	40	12

```
In [ ]: # La matriz de confusion indica parte de la efectividad del algoritmo de identificación para lo que se le configura
# en este caso el algoritmo identificó 380 positivos verdaderos e identificó 12 negativos verdaderos.
```

```
In [ ]: from sklearn.preprocessing import LabelEncoder
```

```
In [ ]: from sklearn.linear_model import LogisticRegression
classifier = LogisticRegression(random_state=10)
classifier.fit(X_train, Y_train)
```

```
Out[ ]: LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
intercept_scaling=1, l1_ratio=None, max_iter=100,
multi_class='auto', n_jobs=None, penalty='l2',
random_state=10, solver='lbfgs', tol=0.0001, verbose=0,
warm_start=False)
```

```
In [ ]: y_pred = classifier.predict(X_test)
```

```
In [ ]: reaccion_positiva = classifier.predict([[2,5,89000,2,2,12,1,1,42]]) # predicción de reacción o no a la campaña apublicitaria
print(reaccion_positiva)

[0.]
```

```
In [ ]: reaccion_positiva = classifier.predict([[3,4,59000,2,0,38,0,0,45]]) # predicción de reacción o no a la campaña apublicitaria
print(reaccion_positiva)

[0.]
```

```
In [ ]: reaccion_positiva = classifier.predict([[4,4,100000,2,8,44,1,0,28]]) # predicción de reacción o no a la campaña apublicitaria
print(reaccion_positiva)

[0.]
```

```
In [ ]: reaccion_positiva = classifier.predict([[3,2,75000,2,1,8,1,1,28]]) # predicción de reacción o no a la campaña apublicitaria
print(reaccion_positiva) # el modelo supervisado permite determinar si un cliente según algunas características,
# va o no a reaccionar a la campaña

[1.]
```

```
In [ ]: data5.iloc[:,[0,1,2,3,4,5,11,12,13,14]]
```

```
Out[ ]:
```

	Education	Marital_Status	Income	Kidhome	Teenhome	Recency	AcceptedCmp3	Compl:
0	2	4	58138.0	0	0	58	0	
1	2	4	46344.0	1	1	38	0	
2	2	5	71613.0	0	0	26	0	
3	2	5	26646.0	1	0	26	0	
4	4	3	58293.0	1	0	94	0	
...
2235	2	3	61223.0	0	1	46	0	
2236	4	5	64014.0	2	1	56	0	
2237	2	2	56981.0	0	0	91	0	
2238	3	5	69245.0	0	1	8	0	
2239	4	3	52869.0	1	1	40	0	

2232 rows × 10 columns



Conclusiones

Cómo son los Clientes?

En términos de Edad, Ingreso, Estado Civil, Nucleo Familiar Comportamiento de Compra

Entre menor se los días de la ultima compra, mejor respuesta tendrá el cliente a la campaña.

Existe relación entre comprar un producto junto con otro, además aquellos que tienen una mayor cantidad de hijos tienden a realizar compras con descuentos.

La cantidad promedio de hijos por hogar corresponde alrededor de un hijo, la edad promedio de los clientes es de 45 años, la edad que más se repite más veces en los clientes es 38.

El ingreso promedio de los clientes es de \$52 237

El monto promedio que los clientes gastan en las compras totales es de \$ 605.79

Los clientes prefieren en su mayoría realizar las compras en las tiendas o establecimientos físicos, seguido de las compras por la web.

Aquellos que van más a las tiendas corresponde a los que se encuentran casados o en una relación.

Según se observó en nivel de compras totales con el pasar de los años va en descenso.

Por medio de la aplicación de un método supervisado, se puede determinar si un cliente puede o no reaccionar positivamente a las campañas.

Además la creación de Cluster de los clientes, nos permite determinar por rangos de edad o niveles de ingreso algún tipo de conducta de compra, que permita formular una campaña publicitaria por el medio de compra preferido del cliente (tiendas y Web)

Recomendaciones

Se pueden realizar campañas de descuentos para aquellos clientes de familias numerosas, promoviendo las compras.

Además aquellos clientes nuevos se les puede invitar a visitar la pagina web , redes sociales y uso de alguna app para promover la visita y fidelización del cliente.

Tomar información de los clientes adicionales como el género, lo cual permitiría segmentar aun mejor una campaña dirigida a los gustos y preferencias del consumidor.

Aquellos clientes que tienen que ir a la tienda por un producto de su preferencia, podría estimularse la compra de un producto adicional si se le ofrece una regalía o combo. Esto por la correlación que existe entre los productos evaluados.

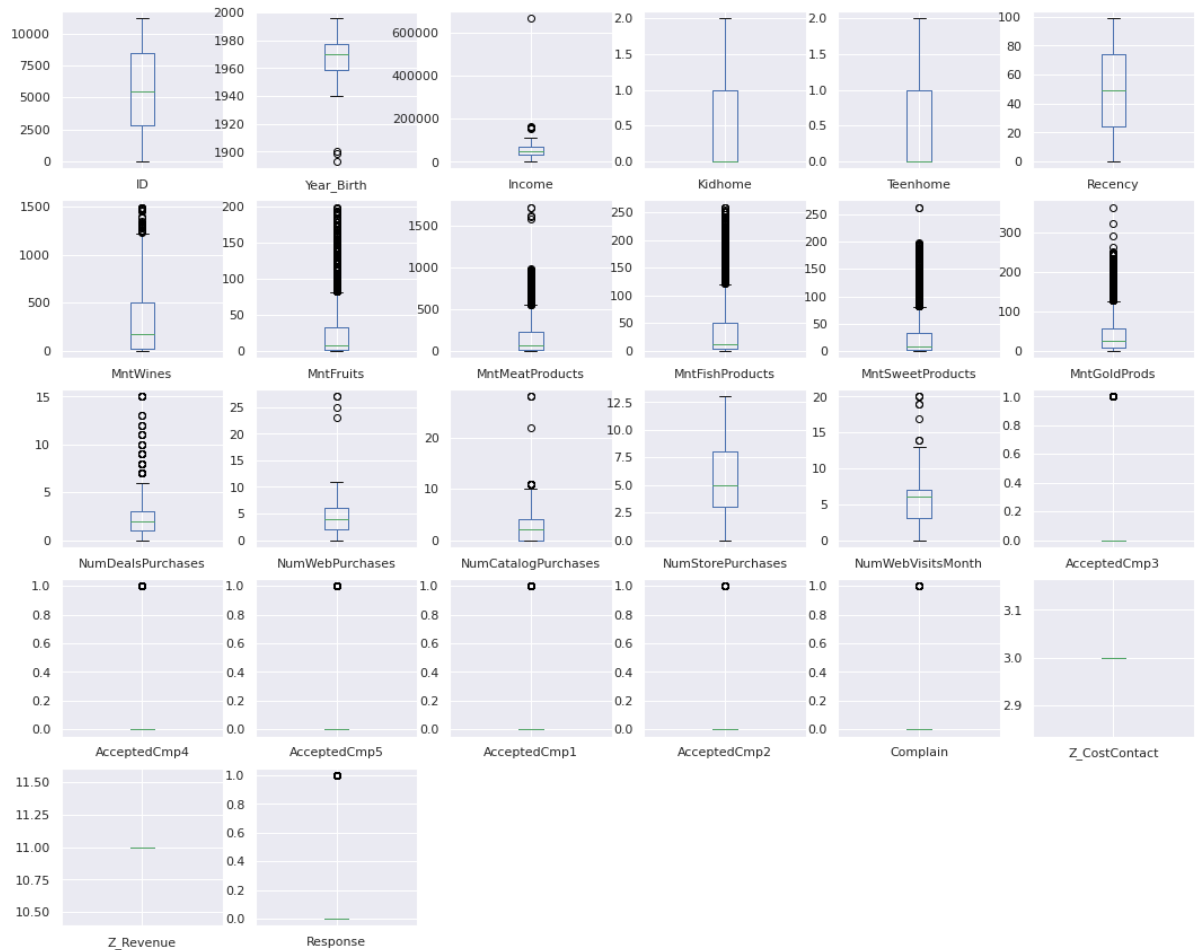
Resumen de las Características

Las variables o dataset utilizados

1. data es el dataset original tal y como se recibe y se le limpia los nulos
2. data1 es el dataset copia del dataset original, utilizado para el proceso de EDA y transformaciones que enriquecen el proceso de análisis
3. data2 con vamos a proceder a convertir las características objeto en numericas
4. data3 la que se utilizó para limpiar los outliers y Kmeans inicial, posteriormente se realiza otro proceso de Kmeans con menos columnas
5. data4 y data5 se utilizan para los procesos de machine learning tienen columnas seleccionadas a criterio para el desarrollo del trabajo

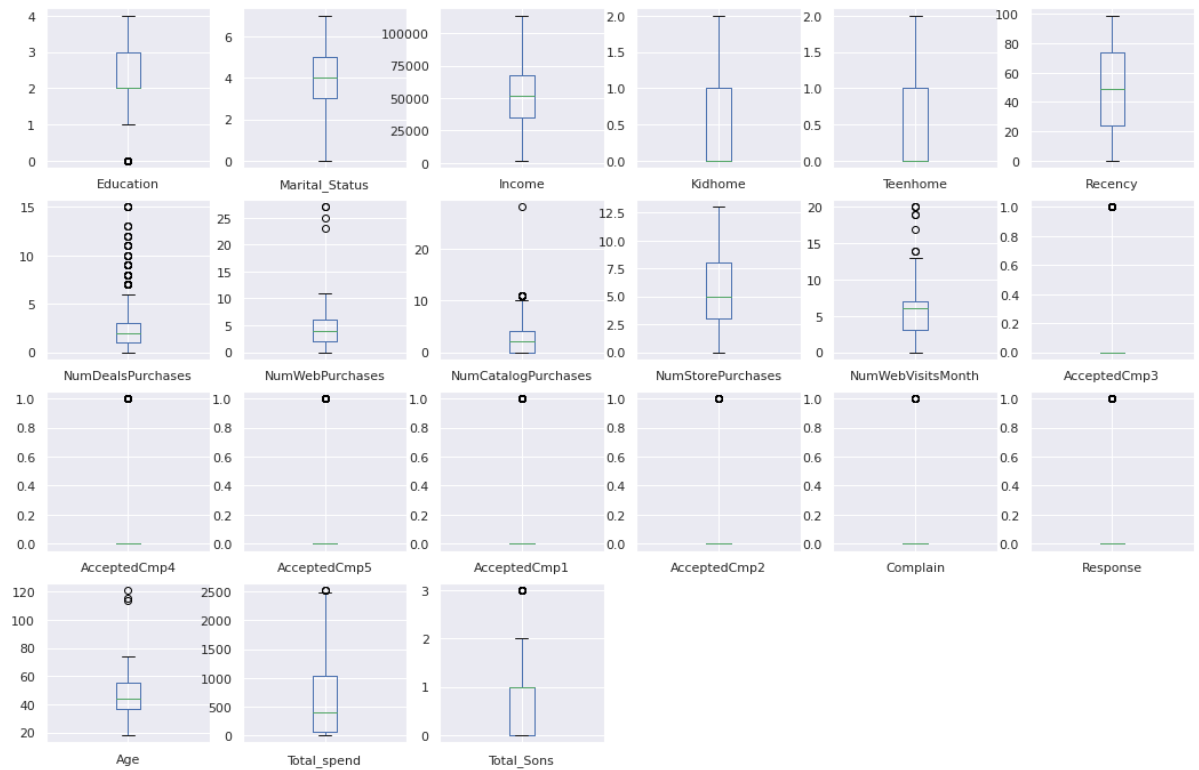
Características

```
In [ ]: data.plot(kind='box',subplots=True, layout=(6,6), sharex=False, sharey=False)
sns.set(rc={'figure.figsize':(18,18)})
plt.show()    # Representación de las características como vienen en el dataset
              original, sin tratamiento de los outliers - atípicos
```

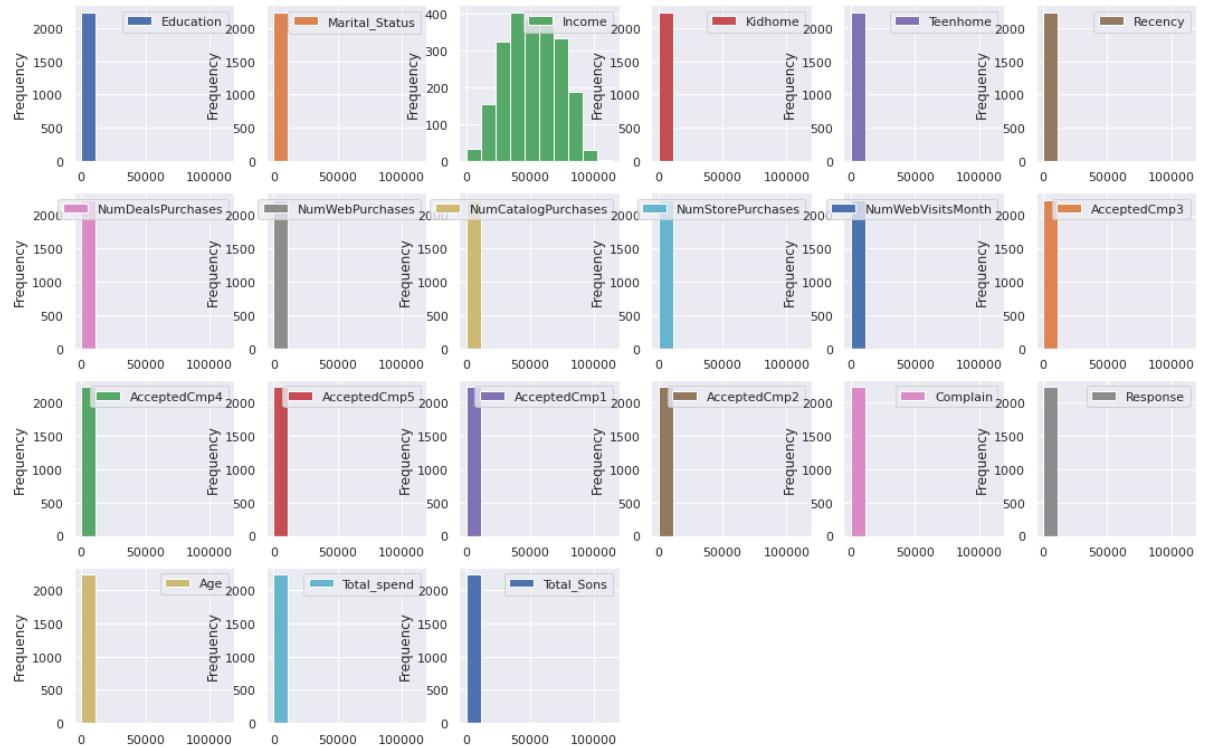


Características posterior del Trabajo de Características y Outliers

```
In [ ]: data3.plot(kind='box',subplots=True, layout=(6,6), sharex=False, sharey=False)
sns.set(rc={'figure.figsize':(18,18)})
plt.show()
```



```
In [ ]: data3.plot(kind='hist',subplots=True, layout=(6,6), sharex=False, sharey=False)
sns.set(rc={'figure.figsize':(18,18)})
plt.show() # presentacion de los histogramas de las características
```




```
In [ ]: data.describe().T
```

```
Out[ ]:
```

	count	mean	std	min	25%	50%	75%
ID	2240.0	5592.159821	3246.662198	0.0	2828.25	5458.5	8427.75
Year_Birth	2240.0	1968.805804	11.984069	1893.0	1959.00	1970.0	1977.00
Income	2240.0	52237.975446	25037.955891	1730.0	35538.75	51381.5	68289.75
Kidhome	2240.0	0.444196	0.538398	0.0	0.00	0.0	1.00
Teenhome	2240.0	0.506250	0.544538	0.0	0.00	0.0	1.00
Recency	2240.0	49.109375	28.962453	0.0	24.00	49.0	74.00
MntWines	2240.0	303.935714	336.597393	0.0	23.75	173.5	504.25
MntFruits	2240.0	26.302232	39.773434	0.0	1.00	8.0	33.00
MntMeatProducts	2240.0	166.950000	225.715373	0.0	16.00	67.0	232.00
MntFishProducts	2240.0	37.525446	54.628979	0.0	3.00	12.0	50.00
MntSweetProducts	2240.0	27.062946	41.280498	0.0	1.00	8.0	33.00
MntGoldProds	2240.0	44.021875	52.167439	0.0	9.00	24.0	56.00
NumDealsPurchases	2240.0	2.325000	1.932238	0.0	1.00	2.0	3.00
NumWebPurchases	2240.0	4.084821	2.778714	0.0	2.00	4.0	6.00
NumCatalogPurchases	2240.0	2.662054	2.923101	0.0	0.00	2.0	4.00
NumStorePurchases	2240.0	5.790179	3.250958	0.0	3.00	5.0	8.00
NumWebVisitsMonth	2240.0	5.316518	2.426645	0.0	3.00	6.0	7.00
AcceptedCmp3	2240.0	0.072768	0.259813	0.0	0.00	0.0	0.00
AcceptedCmp4	2240.0	0.074554	0.262728	0.0	0.00	0.0	0.00
AcceptedCmp5	2240.0	0.072768	0.259813	0.0	0.00	0.0	0.00
AcceptedCmp1	2240.0	0.064286	0.245316	0.0	0.00	0.0	0.00
AcceptedCmp2	2240.0	0.013393	0.114976	0.0	0.00	0.0	0.00
Complain	2240.0	0.009375	0.096391	0.0	0.00	0.0	0.00
Z_CostContact	2240.0	3.000000	0.000000	3.0	3.00	3.0	3.00
Z_Revenue	2240.0	11.000000	0.000000	11.0	11.00	11.0	11.00
Response	2240.0	0.149107	0.356274	0.0	0.00	0.0	0.00



Fuente Primarias

codigo

1. Como extraer año de fecha y hora

<https://www.it-swarm-es.com/es/python/python-pandas-extraer-ano-de-fecha-y-hora-df-ano-df-fecha.-ano-no-funciona/1053538121/> (<https://www.it-swarm-es.com/es/python/python-pandas-extraer-ano-de-fecha-y-hora-df-ano-df-fecha.-ano-no-funciona/1053538121/>)

1. Uso de código visto en la web

PETR KOLAR <https://www.kaggle.com/petrkolar/ml-workflow-0-99-f1> (<https://www.kaggle.com/petrkolar/ml-workflow-0-99-f1>)

MULTICOLLINEARITY (CORRELATION BETWEEN PREDICTOR VARIABLES)

```
cor_matrix = df.corr().abs() cor_matrix.style.background_gradient(sns.light_palette('red', as_cmap=True))
```

Fuentes Secundarias

1. Influencia de la publicidad en el comportamiento de compra de los estudiantes de mercadeo de la Extensión Universitaria de Aguadulce. vol. 5, núm. 1, 2020

<http://portal.amelica.org/ameli/jatsRepo/212/2121146005/html/index.html>
(<http://portal.amelica.org/ameli/jatsRepo/212/2121146005/html/index.html>)

1. Análisis del efecto de la publicidad en la intención de compra y el papel que en esa relación juegan el brand engagement y el brand equity.2018

https://repository.icesi.edu.co/biblioteca_digital/bitstream/10906/84330/1/TG02186.pdf
(https://repository.icesi.edu.co/biblioteca_digital/bitstream/10906/84330/1/TG02186.pdf)

```
In [1]: import pip
```