Proyecto 1 Data mining

Realizado por:

Nombres

- Augusto Alonso - 181085
- Joohno Molina -
- Mario Sarmientos -

In [1]:

```
!pip install plotly
```

```
Requirement already satisfied: plotly in /
Library/Frameworks/Python.framework/Versio
ns/3.7/lib/python3.7/site-packages (4.14.
3)
Requirement already satisfied: six in /Lib
rary/Frameworks/Python.framework/Versions/
3.7/lib/python3.7/site-packages (from plot
ly) (1.11.0)
Requirement already satisfied: retrying>=
1.3.3 in /Library/Frameworks/Python.framew
ork/Versions/3.7/lib/python3.7/site-packag
es (from plotly) (1.3.3)
```

In [2]:

```
!pip install pyreadstat
!pip install seaborn
!pip install xlrd==1.2.0
!pip install sklearn
```

```
Requirement already satisfied: pyreadstat
in /Library/Frameworks/Python.framework/Ve
rsions/3.7/lib/python3.7/site-packages (1.
1.0)
Requirement already satisfied: pandas>0.2
4.0 in /Library/Frameworks/Python.framewor
k/Versions/3.7/lib/python3.7/site-packages
(from pyreadstat) (1.1.5)
Requirement already satisfied: python-date
util>=2.7.3 in /Library/Frameworks/Python.
framework/Versions/3.7/lib/python3.7/site-
packages (from pandas>0.24.0->pyreadstat)
(2.7.3)
Requirement already satisfied: numpy>=1.1
5.4 in /Library/Frameworks/Python.framewor
k/Versions/3.7/lib/python3.7/site-packages
(from pandas>0.24.0->pyreadstat) (1.20.2)
Requirement already satisfied: pytz>=2017.
2 in /Library/Frameworks/Python.framework/
Versions/3.7/lib/python3.7/site-packages
(from pandas>0.24.0->pyreadstat) (2019.3)
Requirement already satisfied: six>=1.5 in
/Library/Frameworks/Python.framework/Versi
ons/3.7/lib/python3.7/site-packages (from
python-dateutil>=2.7.3->pandas>0.24.0->pyr
eadstat) (1.11.0)
Requirement already satisfied: seaborn in
/Library/Frameworks/Python.framework/Versi
ons/3.7/lib/python3.7/site-packages (0.11.
1)
Requirement already satisfied: scipy>=1.0
in /Library/Frameworks/Python.framework/Ve
rsions/3.7/lib/python3.7/site-packages (fr
om seaborn) (1.6.2)
Requirement already satisfied: matplotlib>
```

```
=2.2 in /Library/Frameworks/Python.framewo
rk/Versions/3.7/lib/python3.7/site-package
s (from seaborn) (3.0.0)
Requirement already satisfied: pandas>=0.2
3 in /Library/Frameworks/Python.framework/
Versions/3.7/lib/python3.7/site-packages
(from seaborn) (1.1.5)
Requirement already satisfied: numpy>=1.15
in /Library/Frameworks/Python.framework/Ve
rsions/3.7/lib/python3.7/site-packages (fr
om seaborn) (1.20.2)
Requirement already satisfied: pyparsing!=
2.0.4,!=2.1.2,!=2.1.6,>=2.0.1 in /Library/
Frameworks/Python.framework/Versions/3.7/l
ib/python3.7/site-packages (from matplotli
b>=2.2->seaborn) (2.2.2)
Requirement already satisfied: python-date
util>=2.1 in /Library/Frameworks/Python.fr
amework/Versions/3.7/lib/python3.7/site-pa
ckages (from matplotlib>=2.2->seaborn) (2.
7.3)
Requirement already satisfied: cycler>=0.1
0 in /Library/Frameworks/Python.framework/
Versions/3.7/lib/python3.7/site-packages
(from matplotlib>=2.2->seaborn) (0.10.0)
Requirement already satisfied: kiwisolver>
=1.0.1 in /Library/Frameworks/Python.frame
work/Versions/3.7/lib/python3.7/site-packa
ges (from matplotlib>=2.2->seaborn) (1.0.
1)
Requirement already satisfied: six in /Lib
rary/Frameworks/Python.framework/Versions/
3.7/lib/python3.7/site-packages (from cycl
er>=0.10->matplotlib>=2.2->seaborn) (1.11.
0)
Requirement already satisfied: setuptools
in /Library/Frameworks/Python.framework/Ve
rsions/3.7/lib/python3.7/site-packages (fr
om kiwisolver>=1.0.1->matplotlib>=2.2->sea
born) (39.0.1)
Requirement already satisfied: pytz>=2017.
2 in /Library/Frameworks/Python.framework/
```

```
Versions/3.7/lib/python3.7/site-packages
(from pandas>=0.23->seaborn) (2019.3)
Requirement already satisfied: xlrd==1.2.0
in /Library/Frameworks/Python.framework/Ve
rsions/3.7/lib/python3.7/site-packages (1.
2.0)
Requirement already satisfied: sklearn in
/Library/Frameworks/Python.framework/Versi
ons/3.7/lib/python3.7/site-packages (0.0)
Requirement already satisfied: scikit-lear
n in /Library/Frameworks/Python.framework/
Versions/3.7/lib/python3.7/site-packages
(from sklearn) (0.24.1)
Requirement already satisfied: joblib>=0.1
1 in /Library/Frameworks/Python.framework/
Versions/3.7/lib/python3.7/site-packages
(from scikit-learn->sklearn) (1.0.1)
Requirement already satisfied: numpy>=1.1
3.3 in /Library/Frameworks/Python.framewor
k/Versions/3.7/lib/python3.7/site-packages
(from scikit-learn->sklearn) (1.20.2)
Requirement already satisfied: threadpoolc
tl>=2.0.0 in /Library/Frameworks/Python.fr
amework/Versions/3.7/lib/python3.7/site-pa
ckages (from scikit-learn->sklearn) (2.1.
0)
Requirement already satisfied: scipy>=0.1
9.1 in /Library/Frameworks/Python.framewor
k/Versions/3.7/lib/python3.7/site-packages
(from scikit-learn->sklearn) (1.6.2)
```

In [3]:

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import pyreadstat
import numpy as np
import plotly.express as px
```

In [4]:

```
a = range(2009,2020)
b = "Data/defunciones"
lista = []
nombres = ['Depreg', 'Mupreg', 'Mesreg', 'Añoreg', 'Depo
       'Sexo', 'Diaocu', 'Mesocu', 'Añoocu', 'Edadif', '
       'Ecidif', 'Ocudif', 'Dnadif', 'Mnadif', 'Nacdif',
       'Caudef', 'Asist', 'Ocur', 'Cerdef', 'year']
palabras = ['mupreg', 'mupocu','añoocu', 'Escodif', 'mna
       'caudef.descrip']
```

In [5]:

```
for k in a:
    path = b + str(k) + ".sav"
    df, meta = pyreadstat.read_sav(path)
    columnas = df.columns
    columnas = list(columnas)
    for i in columnas:
        if i in palabras:
            for j in nombres:

                if i.lower() == j.lower():
                    df = df.rename(columns={i: j})
    df["year"] = k
    lista.append(df)
```

In [6]:

```
defunciones = pd.concat(lista)
defunciones.shape
defunciones.columns
```

Out[6]:

```
Index(['Depreg', 'Mupreg', 'Mesreg', 'Añor
eg', 'Depocu', 'Mupocu', 'Areag',
       'Sexo', 'Diaocu', 'Mesocu', 'Añooc
u', 'Edadif', 'Perdif', 'Getdif',
       'Ecidif', 'Ocudif', 'Dnadif', 'Mnad
if', 'Nacdif', 'Dredif', 'Mredif',
       'Caudef', 'Asist', 'Ocur', 'Cerde
f', 'year', 'Escodif', 'Pnadif',
       'Predif', 'Puedif', 'Ciuodif', 'cau
def.descrip'],
      dtype='object')
```

In [7]:

```
defunciones.head()
```

Out[7]:

	Depreg	Mupreg	Mesreg	Añoreg	Depocu	Mupocu	A
0	5.0	0505	1.0	9.0	5.0	0505	
1	1.0	0101	9.0	9.0	1.0	0101	
2	22.0	2206	9.0	9.0	22.0	2206	
3	2.0	0201	12.0	9.0	2.0	0201	
4	1.0	0101	5.0	9.0	1.0	0101	

5 rows × 32 columns

In [8]:

```
defunciones = defunciones[defunciones.columns[:-6]]
defunciones.head()
```

Out[8]:

	Depreg	Mupreg	Mesreg	Añoreg	Depocu	Mupocu	A
0	5.0	0505	1.0	9.0	5.0	0505	
1	1.0	0101	9.0	9.0	1.0	0101	
2	22.0	2206	9.0	9.0	22.0	2206	
3	2.0	0201	12.0	9.0	2.0	0201	
4	1.0	0101	5.0	9.0	1.0	0101	

5 rows × 26 columns

Observamos que tenemos algunas columnas con data perdida

In [9]:

```
defunciones.isna().any()
```

Out[9]:

```
Depreg    False
Mupreg    False
Mesreg    False
Añoreg    False
Depocu    False
Mupocu    False
Areag      True
Sexo      False
Diaocu    False
Mesocu    False
Añoocu     True
Edadif    False
Perdif    False
Getdif     True
Ecidif    False
Ocudif     True
Dnadif    False
Mnadif    False
Nacdif    False
Dredif    False
Mredif    False
Caudef    False
Asist     False
Ocur      False
Cerdef    False
year      False
dtype: bool
```

```
Se decidio llenar los vacios con 0 Puesto que si la
columna es categórica entonces no habría problema
porque a la hora de contar no se categorizaría al
elemento
```

In [10]:

```
data = defunciones.fillna(0)
data.head()
```

Out[10]:

	Depreg	Mupreg	Mesreg	Añoreg	Depocu	Mupocu	A
0	5.0	0505	1.0	9.0	5.0	0505	
1	1.0	0101	9.0	9.0	1.0	0101	
2	22.0	2206	9.0	9.0	22.0	2206	
3	2.0	0201	12.0	9.0	2.0	0201	
4	1.0	0101	5.0	9.0	1.0	0101	

5 rows × 26 columns

In [11]:

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 852835 entries, 0 to 83070
Data columns (total 26 columns):
 #   Column  Non-Null Count   Dtype  
---  ------  --------------   -----  
 0   Depreg  852835 non-null  float64
 1   Mupreg  852835 non-null  object 
 2   Mesreg  852835 non-null  float64
 3   Añoreg  852835 non-null  float64
 4   Depocu  852835 non-null  float64
 5   Mupocu  852835 non-null  object 
 6   Areag   852835 non-null  float64
 7   Sexo    852835 non-null  float64
 8   Diaocu  852835 non-null  float64
 9   Mesocu  852835 non-null  float64
 10  Añoocu  852835 non-null  float64
 11  Edadif  852835 non-null  float64
 12  Perdif  852835 non-null  float64
 13  Getdif  852835 non-null  float64
 14  Ecidif  852835 non-null  float64
 15  Ocudif  852835 non-null  object 
 16  Dnadif  852835 non-null  float64
 17  Mnadif  852835 non-null  object 
 18  Nacdif  852835 non-null  float64
 19  Dredif  852835 non-null  float64
 20  Mredif  852835 non-null  object 
 21  Caudef  852835 non-null  object 
 22  Asist   852835 non-null  float64
 23  Ocur    852835 non-null  float64
 24  Cerdef  852835 non-null  float64
 25  year    852835 non-null  int64  
dtypes: float64(19), int64(1), object(6)
memory usage: 175.7+ MB
```

In [12]:

```
data.describe()
```

Out[12]:

	Depreg	Mesreg	Añoreg	
count	852835.000000	852835.000000	852835.000000	8528
mean	8.661829	6.449960	1677.597797	
std	6.703955	3.447516	750.227793	
min	1.000000	1.000000	9.000000	
25%	1.000000	3.000000	2011.000000	
50%	9.000000	6.000000	2014.000000	
75%	14.000000	9.000000	2017.000000	
max	22.000000	12.000000	2019.000000	

In [13]:

```
dic = pd.read_excel("Data/diccionario.xlsx",sheet_name="
dic["Valor"].unique()
```

Out[13]:

```
array(['Departamento de registro', nan, 'M
unicipio de registro',
       'Mes de registro', 'Año de registr
o', 'Departamento de ocurrencia',
       'Municipio de ocurrencia', 'Área ge
ográfica de ocurrencia',
       'Sexo del difunto(a)', 'Día de ocur
rencia', 'Mes de ocurrencia',
       'Edad del difunto(a)', 'Periodo de
edad del difunto(a)',
       'Grupo étnico del difunto(a)', 'Est
ado civil del difunto(a)',
       'Ocupación del difunto(a)',
       'Departamento de nacimiento del dif
unto(a)',
       'Municipio de nacimiento del difunt
o(a)',
       'Nacionalidad del difunto(a)',
       'Departamento de residencia del dif
unto(a)',
       'Municipio de residencia del difunt
o(a)', 'Causa de defuncion',
       'Asistencia recibida', 'Sitio de oc
urrencia', 'Quien certifica'],
      dtype=object)
```

```
Notese que los departamentos que tienen más fallecidos
son Guatemala, Quetzaltenango, Altaverapaz y San Marcos
```

In [14]:

```
a = data["Depocu"].value_counts()
a[:5]
```

Out[14]:

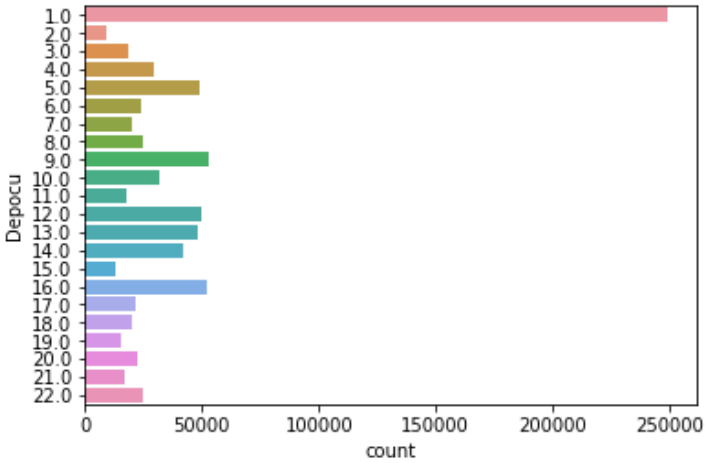
```
1.0     249554
9.0      52543
16.0     52122
12.0     50036
5.0      48986
Name: Depocu, dtype: int64
```

In [15]:

```
sns.countplot(y=data["Depocu"])
```

Out[15]:

```
<matplotlib.axes._subplots.AxesSubplot at
0x7ff4646ed0b8>
```



Por otro lado los municipios donde más gente fallece es Guatemala,Quetzaltenango, Escuintla y Mazatenango

In [16]:

```
a = data["Mupocu"].value_counts()
a[:5]
```

Out[16]:

```
0101    169866
0108     20203
0501     19938
0901     18712
1601     13763
Name: Mupocu, dtype: int64
```

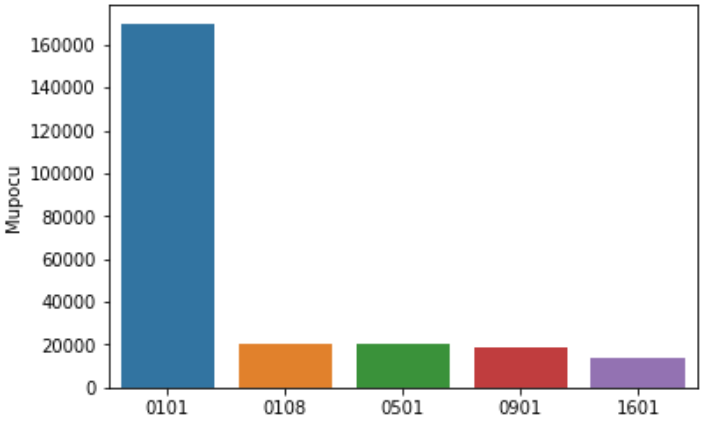
In [17]:

```
b = a[:5]
sns.barplot(b.index,b)
```

```
/Library/Frameworks/Python.framework/Versi
ons/3.7/lib/python3.7/site-packages/seabor
n/_decorators.py:43: FutureWarning: Pass t
he following variables as keyword args: x,
y. From version 0.12, the only valid posit
ional argument will be `data`, and passing
other arguments without an explicit keywor
d will result in an error or misinterpreta
tion.
  FutureWarning
```

Out[17]:

```
<matplotlib.axes._subplots.AxesSubplot at
0x7ff46768eba8>
```



Los meses donde la gente fallecio mas fuel mes de Enero, Julio y Mayo. Aunque no hay mucha diferencia con el resto. Yo no la consideraría una variable para un modelo puesto que apenas hay alguna diferencia.

In [18]:

```
a = data["Mesocu"].value_counts()
a[:5]
```

Out[18]:

```
7.0     73479
1.0     72792
3.0     72758
12.0    72659
8.0     72119
4.0     71605
10.0    71602
5.0     71560
Name: Mesocu, dtype: int64
```

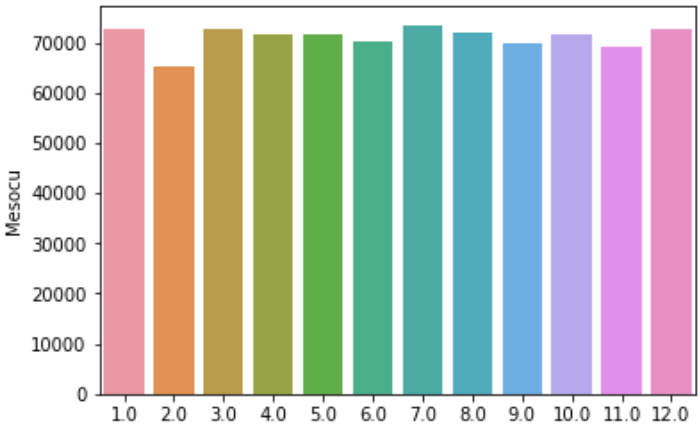
In [19]:

```
sns.barplot(a.index,a)
```

```
/Library/Frameworks/Python.framework/Versi
ons/3.7/lib/python3.7/site-packages/seabor
n/_decorators.py:43: FutureWarning: Pass t
he following variables as keyword args: x,
y. From version 0.12, the only valid posit
ional argument will be `data`, and passing
other arguments without an explicit keywor
d will result in an error or misinterpreta
tion.
  FutureWarning
```

Out[19]:

```
<matplotlib.axes._subplots.AxesSubplot at
0x7ff45c6d59b0>
```



Si ignoramos la categoría donde se llenaron los datos no disponibles vemos que ha ido en aumento el número de defunciones en Guatemala,

en este periodo han aumentado más de 10000 muertes

In [20]:

```
a = data["Añoocu"].value_counts()
a
```

Out[20]:

```
0.0       227103
2017.0    163452
9.0       143414
2018.0     83071
2016.0     82565
2015.0     80876
2011.0     72354
Name: Añoocu, dtype: int64
```

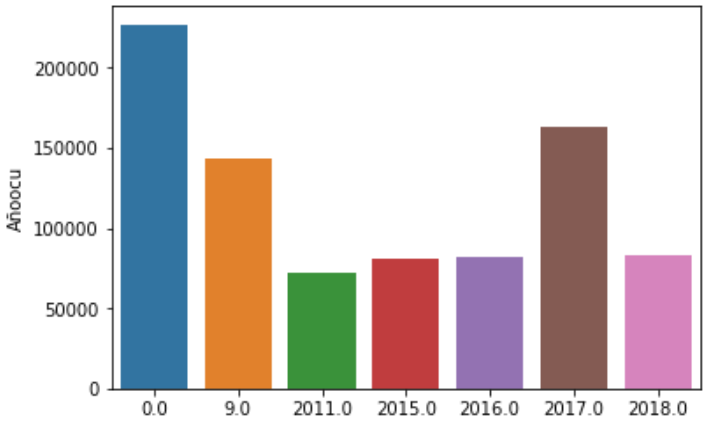
In [21]:

```
sns.barplot(a.index,a)
```

```
/Library/Frameworks/Python.framework/Versi
ons/3.7/lib/python3.7/site-packages/seabor
n/_decorators.py:43: FutureWarning: Pass t
he following variables as keyword args: x,
y. From version 0.12, the only valid posit
ional argument will be `data`, and passing
other arguments without an explicit keywor
d will result in an error or misinterpreta
tion.
  FutureWarning
```

Out[21]:

```
<matplotlib.axes._subplots.AxesSubplot at
0x7ff45d799668>
```



Claramente vemos como es que los hombres murieron más que las mujeres, sin embargo la proporción es bastante similar

In [22]:

```
a = data["Sexo"].value_counts()
a
```

Out[22]:

```
1.0    481148
2.0    371687
Name: Sexo, dtype: int64
```

In [23]:

```
plt.pie(a,labels=a.index,autopct='%1.2f%%')
plt.title("Hombres y mujeres fallecidos entre 2009 a 201
```

Out[23]:

```
Text(0.5, 1.0, 'Hombres y mujeres fallecid
os entre 2009 a 2019')
```

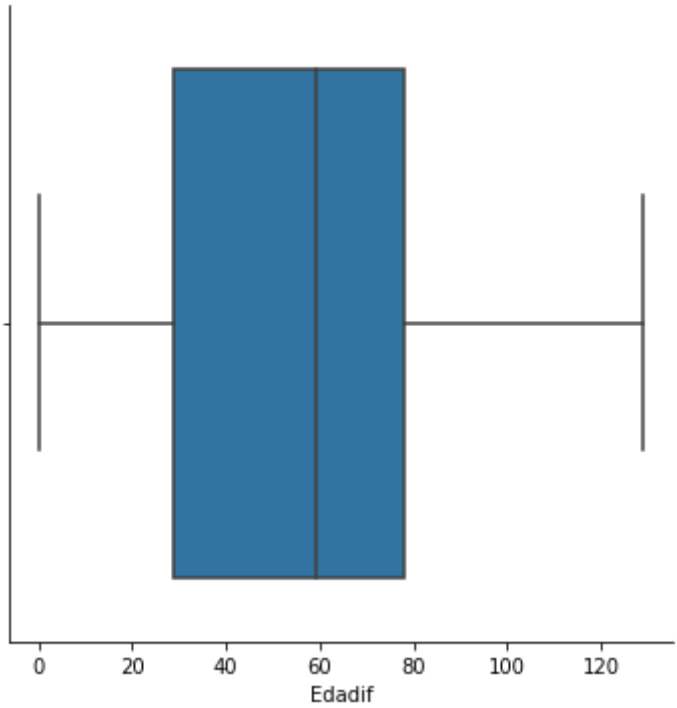
Vemos que ha medida que las personas van aumentando de edad también aumenta la probabilidad de fallecer puesto que el histogram tiene una asimetría hacia la izquierda. Sin embargo hasta la izquierda tenemos un gran número de defunciones que son los niños menores de un año

In [24]:

```
sns.catplot(x = "Edadif",
            data = data,
            kind = "box",
            sym = "", # to ommit the points that are so f
            )
data.Edadif.quantile([0.25,0.5,0.75])
```

Out[24]:

```
0.25    29.0
0.50    59.0
0.75    78.0
Name: Edadif, dtype: float64
```

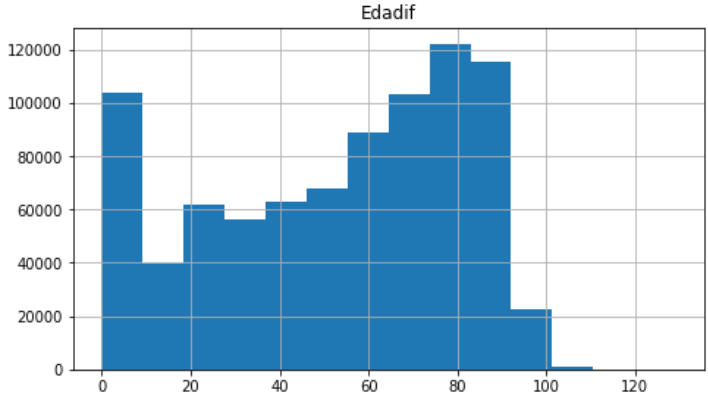


In [25]:

```
from matplotlib import rcParams
rcParams['figure.figsize'] = 7.7,4.27
# Los datos atipicos que removimos aca fueron porque hub
data = data[data["Edadif"] < 200]
data.hist('Edadif',bins=14)
```

Out[25]:

```
array([[<matplotlib.axes._subplots.AxesSub
plot object at 0x7ff474f46278>]],
      dtype=object)
```



Vemos que las personas que no son del grupo índigena fallecieron más que las personas que si pertenecen a este grupo, pero esta variable no la tomaria en cuenta debido a la poca cantidad de datos

In [26]:

```
a = data["Getdif"].value_counts()
a
```

Out[26]:

```
0.0    560119
9.0    103521
2.0     95016
1.0     87019
Name: Getdif, dtype: int64
```

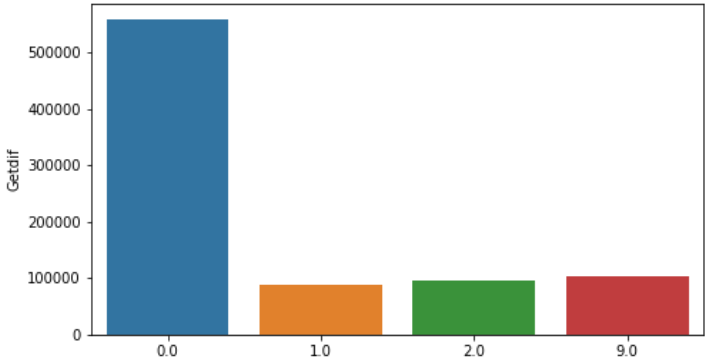
In [27]:

```
sns.barplot(a.index,a)
```

```
/Library/Frameworks/Python.framework/Versi
ons/3.7/lib/python3.7/site-packages/seabor
n/_decorators.py:43: FutureWarning: Pass t
he following variables as keyword args: x,
y. From version 0.12, the only valid posit
ional argument will be `data`, and passing
other arguments without an explicit keywor
d will result in an error or misinterpreta
tion.
  FutureWarning
```

Out[27]:

```
<matplotlib.axes._subplots.AxesSubplot at
0x7ff42c5503c8>
```



Vemos que las personas que están solteras fueron las más fallecieron a lo largo de este período

In [28]:

```
a = data['Ecidif'].value_counts()
a
```

Out[28]:

```
1.0    536530
2.0    292669
9.0      8519
3.0      7957
Name: Ecidif, dtype: int64
```

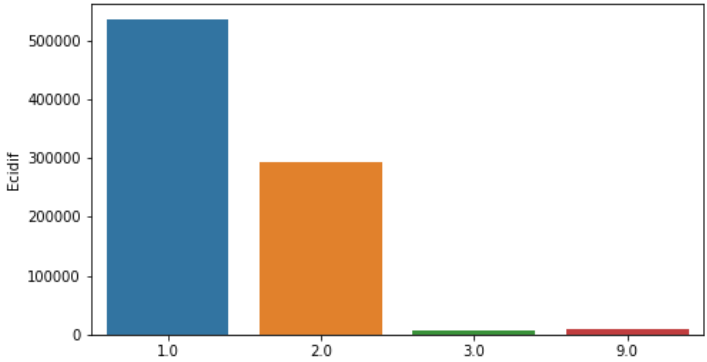
In [29]:

```
sns.barplot(a.index,a)
```

```
/Library/Frameworks/Python.framework/Versi
ons/3.7/lib/python3.7/site-packages/seabor
n/_decorators.py:43: FutureWarning: Pass t
he following variables as keyword args: x,
y. From version 0.12, the only valid posit
ional argument will be `data`, and passing
other arguments without an explicit keywor
d will result in an error or misinterpreta
tion.
  FutureWarning
```

Out[29]:

```
<matplotlib.axes._subplots.AxesSubplot at
0x7ff45da6de10>
```



Las ocupaciones de las personas que más fallecieron fueron las de trabajos domésticos no remunerados, peones de explotaciones agrícolas, agricultores y trabajadores calificados de cultivos extensivos, Estudiante

In [30]:

```
a = data['Ocudif'].value_counts()
a[1:6]
```

Out[30]:

```
9711    81831
9211    54481
9714    43132
6111    23204
9712    18713
Name: Ocudif, dtype: int64
```

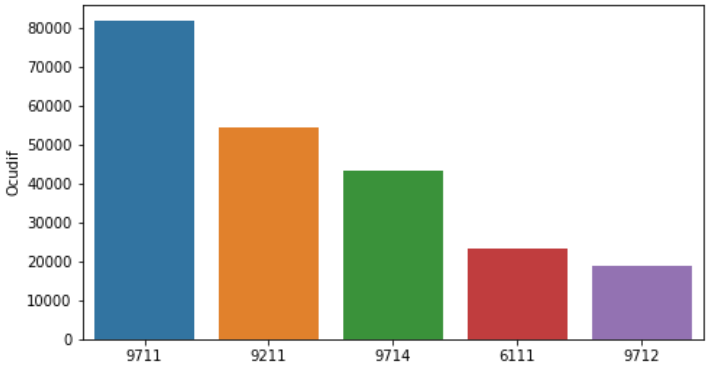
In [31]:

```
b = a[1:6]
sns.barplot(b.index,b)
```

```
/Library/Frameworks/Python.framework/Versi
ons/3.7/lib/python3.7/site-packages/seabor
n/_decorators.py:43: FutureWarning: Pass t
he following variables as keyword args: x,
y. From version 0.12, the only valid posit
ional argument will be `data`, and passing
other arguments without an explicit keywor
d will result in an error or misinterpreta
tion.
  FutureWarning
```

Out[31]:

```
<matplotlib.axes._subplots.AxesSubplot at
0x7ff42cc9d0b8>
```



Los departamentos donde nacieron la mayoria de los difuntos son Guatemala, San Marcos, Alta Verapaz y Huehuetenango

In [32]:

```
a = data['Dnadif'].value_counts()
a[:9]
```

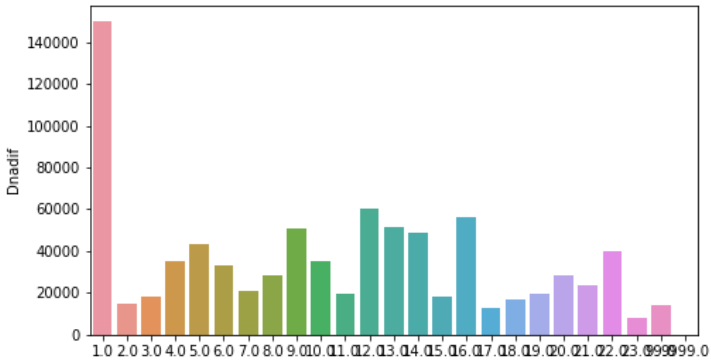
Out[32]:

```
1.0     150096
12.0     60023
16.0     56215
13.0     51176
9.0      50694
Name: Dnadif, dtype: int64
```

In [33]:

```
g = sns.barplot(a.index,a)
```

```
/Library/Frameworks/Python.framework/Versi
ons/3.7/lib/python3.7/site-packages/seabor
n/_decorators.py:43: FutureWarning: Pass t
he following variables as keyword args: x,
y. From version 0.12, the only valid posit
ional argument will be `data`, and passing
other arguments without an explicit keywor
d will result in an error or misinterpreta
tion.
  FutureWarning
```



Las causas principales de fallecimiento fueron: Infarto agudo del miocardio, sin otra especificación, Neumonía, no especificada, Diabetes mellitus no especificada, sin mención de complicación, Muerte sin asistencia, Exposición a factores no especificados, causando otras lesiones y las no especificadas.

In [34]:

```
a = data['Caudef'].value_counts()
a[:5]
```

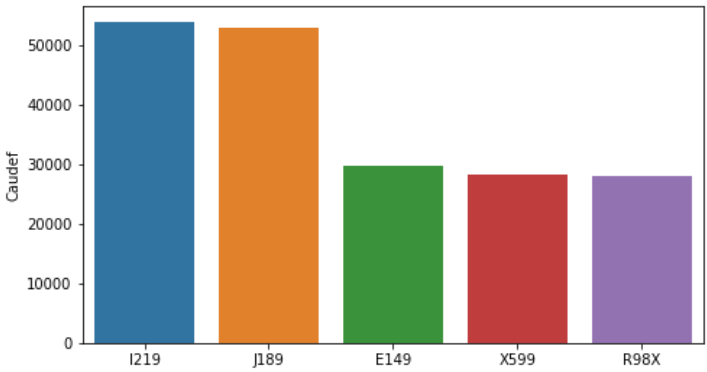
Out[34]:

```
I219    54027
J189    53038
E149    29872
X599    28308
R98X    28099
Name: Caudef, dtype: int64
```

In [35]:

```
b = a[:5]
g = sns.barplot(b.index,b)
```

```
/Library/Frameworks/Python.framework/Versi
ons/3.7/lib/python3.7/site-packages/seabor
n/_decorators.py:43: FutureWarning: Pass t
he following variables as keyword args: x,
y. From version 0.12, the only valid posit
ional argument will be `data`, and passing
other arguments without an explicit keywor
d will result in an error or misinterpreta
tion.
  FutureWarning
```



La gran mayoría no recibio ninguna asistencia, aunque su proporción es bastante parecida con los que recibieron atención médica, muy por atrás encontramos la asistencia empírica y los otros tipos de asistencia.

In [36]:

```
a = data['Asist'].value_counts()
a
```

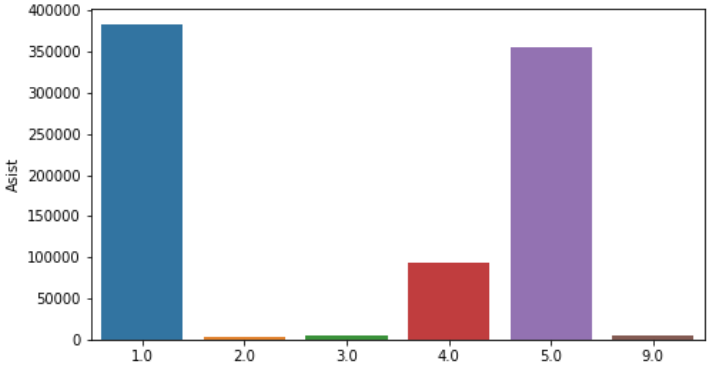
Out[36]:

```
1.0    383618
5.0    355526
4.0     93984
9.0      5056
3.0      4361
2.0      3130
Name: Asist, dtype: int64
```

In [37]:

```
g = sns.barplot(a.index,a)
```

```
/Library/Frameworks/Python.framework/Versi
ons/3.7/lib/python3.7/site-packages/seabor
n/_decorators.py:43: FutureWarning: Pass t
he following variables as keyword args: x,
y. From version 0.12, the only valid posit
ional argument will be `data`, and passing
other arguments without an explicit keywor
d will result in an error or misinterpreta
tion.
  FutureWarning
```



In [38]:

```
data['Asist']
```

Out[38]:

```
0         1.0
1         4.0
2         4.0
3         4.0
4         4.0
         ...
83066     1.0
83067     1.0
83068     5.0
83069     9.0
83070     1.0
Name: Asist, Length: 845675, dtype: float64
```

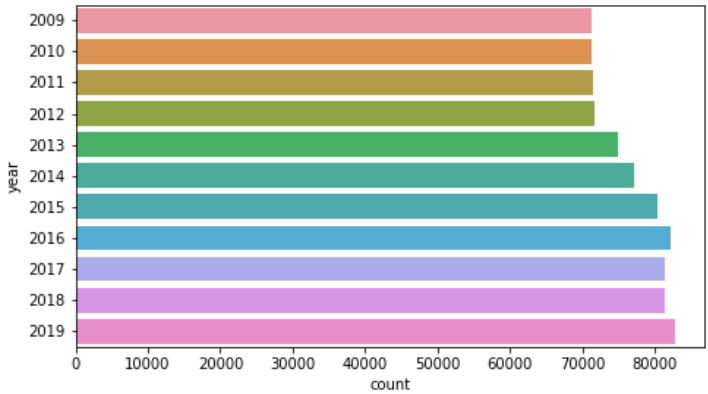
Podemos observar que a lo largo de los años la cantidad de defunciones ha ido en aumento, podemos observar que en este periodo de 10 años han aumentado las defunciones aproximadamente

In [39]:

```
sns.countplot(y=data["year"])
```

Out[39]:

```
<matplotlib.axes._subplots.AxesSubplot at
0x7ff45d949438>
```



Cruzando variables

En la siguiente tabla cruzamos el departamento donde se registro el difunto y el departamento de residia el difunto, encontramos por ejemplo que son más de 50000 personas las que vivían en el departamento de Guatemala pero sin embargo fallecieron fuera de este departamento

In [40]:

```
pd_crosstab = pd.crosstab(data["Depreg"], data["Dredif"]
pd_crosstab
```

2.0	203	8188	3	5	16	8	2	2	6
3.0	780	13	14272	579	228	59	15	17	24
4.0	304	2	261	26264	53	19	40	28	35
5.0	1108	35	120	182	39075	641	60	31	90
6.0	750	15	13	24	168	18842	6	15	6
7.0	109	6	15	62	21	4	16825	62	20
8.0	92	2	6	7	9	7	50	22422	212
9.0	304	8	13	30	101	9	287	1396	41876
10.0	207	4	19	35	230	18	633	24	221
11.0	85	3	5	7	57	9	5	14	314
12.0	274	2	2	9	30	7	6	23	212
13.0	93	3	5	5	15	1	12	30	110

In [41]:

```
# Creamos una columna clasificando a los distintos grupo
criteria = [data['Edadif'].between(0, 1),data['Edadif'].
            data['Edadif'].between(35.1, 50),data['Edadi
values = ["Menos de un año", "Niño", "Adolescente","Jove

data['Edadrange'] = np.select(criteria, values, 0)
```

Ahora cruzamos los grupos de edad y el sexo donde vemos que las mujeres de edad avanzada fueron las personas que más fallecieron en este período de tiempo seguidas de los hombres.

In [42]:

```
pd_crosstab = pd.crosstab(data["Edadrange"], data["Sexo"
pd_crosstab
```

Out[42]:

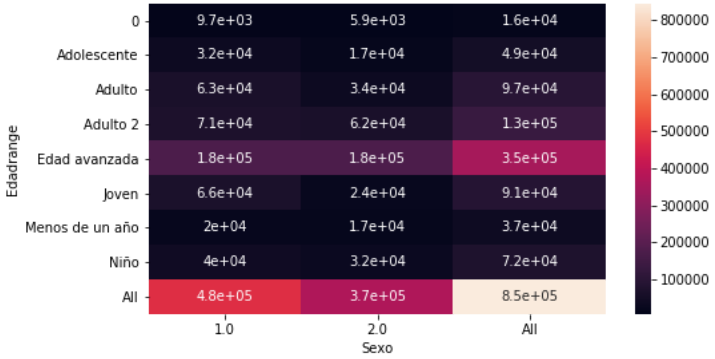
Sexo	1.0	2.0	All
Edadrange			
0	9738	5871	15609
Adolescente	32178	16902	49080
Adulto	62799	34421	97220
Adulto 2	70517	62282	132799
Edad avanzada	175495	176217	351712
Joven	66093	24456	90549
Menos de un año	20047	16770	36817
Niño	39863	32026	71889
All	476730	368945	845675

In [43]:

```
sns.heatmap(pd_crosstab,annot=True)
```

Out[43]:

```
<matplotlib.axes._subplots.AxesSubplot at
0x7ff464846048>
```



Los hombres solteros fueron las personas que más fallecieron en este período seguidos de las mujeres solteras

In [44]:

```
pd_crosstab = pd.crosstab(data["Ecidif"], data["Sexo"],
pd_crosstab
```

Out[44]:

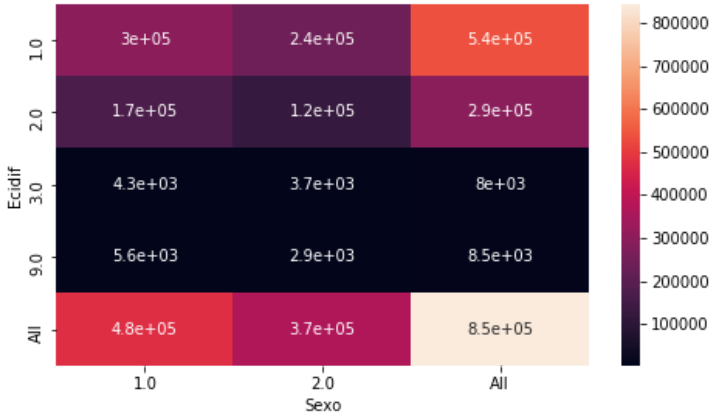
Sexo	1.0	2.0	All
Ecidif			
1.0	296156	240374	536530
2.0	170674	121995	292669
3.0	4275	3682	7957
9.0	5625	2894	8519
All	476730	368945	845675

In [45]:

```
sns.heatmap(pd_crosstab,annot=True)
```

Out[45]:

```
<matplotlib.axes._subplots.AxesSubplot at
0x7ff4649947f0>
```



En la siguiente tabla comparamos la causa de defunción con el departamento donde sucede y vemos que en Guatemala es donde más gente fallece por ataques al corazón, seguido de Quetzaltenango y Alta Verapaz.

Con la Neumonía tenemos a Alta Verapaz como el departamento donde más gente muere por esto, esto puede ser debido a sus temperaturas frías y sus altas temperaturas, seguido de Guatemala y Totonicapán.

In [46]:

```
pd_crosstab = pd.crosstab(data["Caudef"], data["Dredif"]
pd_crosstab.sort_values("All",ascending=False).head(6)
```

Out[46]:

Dredif	1.0	2.0	3.0	4.0	5.0	6.0	7
Caudef							
All	195477	11183	17437	29860	45806	23197	188
I219	12901	1594	996	1560	3542	1869	6
J189	5305	557	724	2645	1718	852	23
E149	8917	478	736	1108	1923	746	5
X599	8151	295	737	1083	1679	710	5
R98X	1613	54	333	491	2507	1058	5

6 rows × 25 columns

A continuación hacemos una tabla del rango de edad y la causa de muerte, donde vemos que las personas mayores fallecieron a causa de enfermedades del corazón, seguida de la diabetes y muerte sin asistencia.

El segundo grupo más afectado son los adultos que se encuentran entre los 50 y 65 años de edad

In [47]:

```
pd_crosstab = pd.crosstab(data["Caudef"], data["Edadrang
pd_crosstab.sort_values("All",ascending=False).head(6)
```

Out[47]:

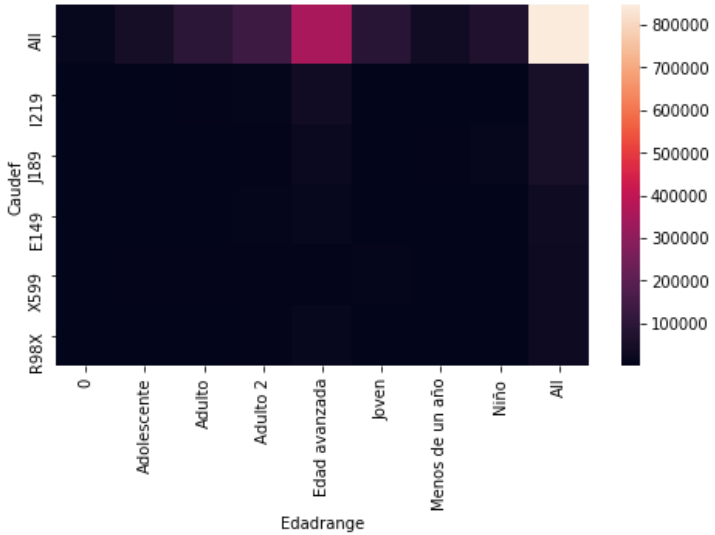
Edadrange	0	Adolescente	Adulto	Adulto 2	Edac avanzada
Caudef					
All	15609	49080	97220	132799	351712
I219	727	347	4103	9010	38048
J189	501	1447	2493	4634	26023
E149	414	71	3272	9255	16198
X599	958	3444	5501	3897	4771
R98X	432	925	2969	4714	14867

In [48]:

```
a = pd_crosstab.sort_values("All",ascending=False).head(
sns.heatmap(a)
```

Out[48]:

```
<matplotlib.axes._subplots.AxesSubplot at
0x7ff42da990b8>
```



Aplicando Kmeans

In [49]:

```
from sklearn.cluster import KMeans
data = data.select_dtypes(exclude=['object'])
```

In [50]:

```
lista2 = []
for i in range(1,11):
    kmeans = KMeans(n_clusters = i, max_iter = 300)
    kmeans.fit(data)
    lista2.append(kmeans.inertia_)
```

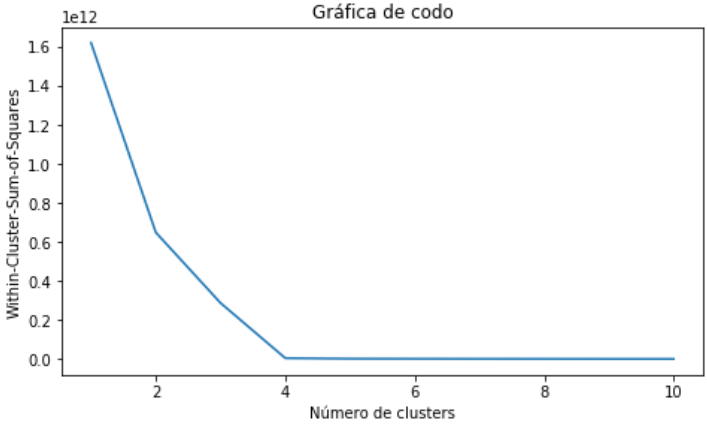
Al hacer la gráfica de codo vemos que el número optimo de clusters para este conjunto de datos es de 4 puesto que es el que menor error cuadrado tiene y práctiamente tiene la misma precisión con más clusters

In [51]:

```
plt.plot(range(1,11),lista2)
plt.title("Gráfica de codo")
plt.xlabel("Número de clusters")
plt.ylabel("Within-Cluster-Sum-of-Squares")
```

Out[51]:

```
Text(0, 0.5, 'Within-Cluster-Sum-of-Square
s')
```



In [52]:

```
# Create a KMeans instance with 3 clusters: model
model = KMeans(n_clusters = 4)

# Fit model to points
model.fit(data)
```

Out[52]:

```
KMeans(n_clusters=4)
```

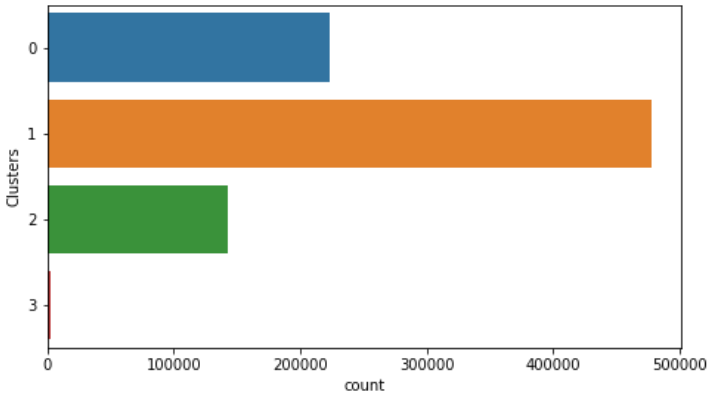
Creado ya el modelo creamos una nueva columna para ver a donde pertenece cada individuo

In [53]:

```
data['Clusters'] = model.labels_
sns.countplot(y=data["Clusters"])
```

Out[53]:

```
<matplotlib.axes._subplots.AxesSubplot at
0x7ff4647e3438>
```



Note que el sexo no es un factor tan importante en los primeros 3 clusters, solamente en el cluster 3 donde la mayoria son hombres

In [54]:

```
pd_crosstab = pd.crosstab(data["Clusters"], data["Sexo"]
pd_crosstab
```

Out[54]:

Sexo	1.0	2.0
Clusters		
0	0.562266	0.437734
1	0.559240	0.440760
2	0.576605	0.423395
3	0.770915	0.229085
All	0.563727	0.436273

En todos los clusters se mantiene casi la misma proporcion en cuanto al departamento donde fallecieron menos en el cluster 3

In [55]:

```
pd_crosstab = pd.crosstab(data["Clusters"], data["Depocu
pd_crosstab
```

Out[55]:

Depocu	1.0	2.0	3.0	4.0	5.0
Clusters					
0	0.291107	0.010598	0.021784	0.035304	0.057093
1	0.289857	0.010812	0.022384	0.035110	0.058710
2	0.299848	0.011265	0.020830	0.034229	0.053205
3	0.755556	0.006209	0.009477	0.003268	0.028105
All	0.293554	0.010815	0.021917	0.034898	0.057247

5 rows × 22 columns

Notemos que los que pertenecen al cluster 2 y 1 tienen una mayor proporcion de personas con edad avanzada que el cluster 0

In [56]:

```
# Creamos una columna clasificando a los distintos grupo
criteria = [data['Edadif'].between(0, 1),data['Edadif'].
            data['Edadif'].between(35.1, 50),data['Edadi
values = ["Menos de un año", "Niño", "Adolescente","Jove

data['Edadrange'] = np.select(criteria, values, 0)

pd_crosstab = pd.crosstab(data["Clusters"], data["Edadra
pd_crosstab
```

Out[56]:

Edadrange	0	Adolescente	Adulto	Adulto 2	av
Clusters					
0	0.018225	0.057650	0.113625	0.157893	0
1	0.018396	0.055920	0.114011	0.160357	0
2	0.019229	0.065130	0.117844	0.145121	0
3	0.009150	0.086275	0.226471	0.130065	0
All	0.018457	0.058036	0.114961	0.157033	0

In [59]:

```
data2 = data.select_dtypes(exclude=['object'])
X = data2.iloc[:,:].values
from sklearn.metrics import silhouette_score
grupos = [KMeans(n_clusters = i, max_iter = 300).fit(X)
```

In [*]:

```
scores = [silhouette_score(X, model.labels_) for model i
scores
```

In [ ]:

```

```