Energy Usage Chicago 2010 - Ensayos3-final-topdf

January 6, 2021

1 Consumo de energía en Chicago en 2010

Alberto Ramos Sánchez

08/01/2021

1.1 Contenido

Dependencias

[1]:
```
import pandas as pd
import numpy as np
```

```
from scipy.stats import zscore

import seaborn as sns
import plotly.express as px
import matplotlib.pyplot as plt
import plotly.graph_objects as go
from plotly.subplots import make_subplots

from yellowbrick.cluster import silhouette_visualizer

import datetime

from sklearn import preprocessing

from scipy.stats import zscore

from dataloader import DataLoader

from clustering import KMeansCluster, CMeansCluster

from visualization import tsne

from pathlib import Path

images_path = Path("./images2")

np.random.seed(42)
```

1.2 Exploración de los datos

Leemos de fichero los datos previamente preprocesados

[2]:
```
df = pd.read_csv('energy-usage-2010-clean.csv')
df
```

[2]:

	COMMUNITY AREA NAME	CENSUS BLOCK	BUILDING TYPE	BUILDING_SUBTYPE
0	Archer Heights	1.703157e+14	Residential	Multi < 7
1	Ashburn	1.703170e+14	Residential	Multi 7+
2	Auburn Gresham	1.703171e+14	Commercial	Multi < 7
3	Austin	1.703125e+14	Commercial	Multi < 7
4	Austin	1.703125e+14	Commercial	Multi < 7
…	…	…	…	…
67046	Woodlawn	1.703184e+14	Residential	Single Family
67047	Woodlawn	1.703184e+14	Commercial	Multi < 7
67048	Woodlawn	1.703184e+14	Residential	Multi < 7
67049	Woodlawn	1.703184e+14	Residential	Single Family
67050	Woodlawn	1.703184e+14	Residential	Multi < 7

```
       KWH JANUARY 2010  KWH FEBRUARY 2010  KWH MARCH 2010  KWH APRIL 2010  \
0                   NaN                NaN             NaN             NaN
1                7334.0             7741.0          4214.0          4284.0
2                   NaN                NaN             NaN             NaN
3                   NaN                NaN             NaN             NaN
4                   NaN                NaN             NaN             NaN
…                    …                  …               …               …
67046            2705.0             1318.0          1582.0          1465.0
67047            1005.0             1760.0          1521.0          1832.0
67048            3567.0             3031.0          2582.0          2295.0
67049            1208.0             1055.0          1008.0          1109.0
67050            2717.0             3057.0          2695.0          3793.0

       KWH MAY 2010  KWH JUNE 2010  …  TOTAL POPULATION  TOTAL UNITS  \
0               NaN            NaN  …              89.0         24.0
1            2518.0         4273.0  …             112.0         67.0
2               NaN            NaN  …             102.0         48.0
3               NaN            NaN  …             121.0         56.0
4               NaN            NaN  …              62.0         23.0
…                …              …  …                …           …
67046        1494.0         2990.0  …             116.0         55.0
67047        2272.0         2361.0  …              31.0         24.0
67048        7902.0         4987.0  …              31.0         24.0
67049        1591.0         1367.0  …               0.0          0.0
67050        4237.0         5383.0  …              77.0         49.0

       AVERAGE STORIES  AVERAGE BUILDING AGE  AVERAGE HOUSESIZE  \
0                 2.00                 71.33               3.87
1                 2.00                 41.00               1.81
2                 3.00                 86.00               3.00
3                 2.00                 84.00               2.95
4                 2.00                 85.00               3.26
…                  …                    …                  …
67046             1.00                  0.00               3.14
67047             3.00                104.50               2.07
67048             2.33                100.67               2.07
67049             1.00                  0.00               0.00
67050             2.00                 79.40               2.57

       OCCUPIED UNITS  OCCUPIED UNITS PERCENTAGE  \
0                23.0                      0.9582
1                62.0                      0.9254
2                34.0                      0.7082
3                41.0                      0.7321
4                19.0                      0.8261
…                 …                         …
```

```
67046            37.0                    0.6727
67047            15.0                    0.6250
67048            15.0                    0.6250
67049             0.0                       NaN
67050            30.0                    0.6122

       RENTER-OCCUPIED HOUSING UNITS  RENTER-OCCUPIED HOUSING PERCENTAGE  \
0                                9.0                                0.3910
1                               50.0                                0.8059
2                               23.0                                0.6759
3                               32.0                                0.7800
4                               11.0                                0.5790
…                                …                                   …
67046                           26.0                                0.7030
67047                           13.0                                0.8670
67048                           13.0                                0.8670
67049                            0.0                                   NaN
67050                           28.0                                0.9329

       OCCUPIED HOUSING UNITS
0                        23.0
1                        62.0
2                        34.0
3                        41.0
4                        19.0
…                         …
67046                    37.0
67047                    15.0
67048                    15.0
67049                     0.0
67050                    30.0

[67051 rows x 73 columns]
```

El dataset contiene 72 columnas que contienen —además del consumo energético mensual— características de cada uno de los bloques censales de cada comunidad de Chicago.

[3]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 67051 entries, 0 to 67050
Data columns (total 73 columns):
 #   Column                          Non-Null Count  Dtype
---  ------                          --------------  -----
 0   COMMUNITY AREA NAME             67051 non-null  object
 1   CENSUS BLOCK                    66974 non-null  float64
 2   BUILDING TYPE                   66974 non-null  object
 3   BUILDING_SUBTYPE                66974 non-null  object
```

```
4   KWH JANUARY 2010                     66180 non-null  float64
5   KWH FEBRUARY 2010                    66180 non-null  float64
6   KWH MARCH 2010                       66180 non-null  float64
7   KWH APRIL 2010                       66180 non-null  float64
8   KWH MAY 2010                         66180 non-null  float64
9   KWH JUNE 2010                        66180 non-null  float64
10  KWH JULY 2010                        66180 non-null  float64
11  KWH AUGUST 2010                      66180 non-null  float64
12  KWH SEPTEMBER 2010                   66180 non-null  float64
13  KWH OCTOBER 2010                     66180 non-null  float64
14  KWH NOVEMBER 2010                    66180 non-null  float64
15  KWH DECEMBER 2010                    66180 non-null  float64
16  TOTAL KWH                            66180 non-null  float64
17  ELECTRICITY ACCOUNTS                 66180 non-null  float64
18  ZERO KWH ACCOUNTS                    67051 non-null  int64
19  THERM JANUARY 2010                   64821 non-null  float64
20  THERM FEBRUARY 2010                  62819 non-null  float64
21  THERM MARCH 2010                     65569 non-null  float64
22  THERM APRIL 2010                     65476 non-null  float64
23  THERM MAY 2010                       65194 non-null  float64
24  THERM JUNE 2010                      65284 non-null  float64
25  THERM JULY 2010                      65231 non-null  float64
26  THERM AUGUST 2010                    65143 non-null  float64
27  THERM SEPTEMBER 2010                 64769 non-null  float64
28  THERM OCTOBER 2010                   65329 non-null  float64
29  THERM NOVEMBER 2010                  65492 non-null  float64
30  THERM DECEMBER 2010                  65507 non-null  float64
31  TOTAL THERMS                         65755 non-null  float64
32  GAS ACCOUNTS                         65755 non-null  float64
33  KWH TOTAL SQFT                       65901 non-null  float64
34  THERMS TOTAL SQFT                    65378 non-null  float64
35  KWH MEAN 2010                        66180 non-null  float64
36  KWH STANDARD DEVIATION 2010          57095 non-null  float64
37  KWH MINIMUM 2010                     66180 non-null  float64
38  KWH 1ST QUARTILE 2010                66180 non-null  float64
39  KWH 2ND QUARTILE 2010                66180 non-null  float64
40  KWH 3RD QUARTILE 2010                66180 non-null  float64
41  KWH MAXIMUM 2010                     66180 non-null  float64
42  KWH SQFT MEAN 2010                   65901 non-null  float64
43  KWH SQFT STANDARD DEVIATION 2010     51666 non-null  float64
44  KWH SQFT MINIMUM 2010                65901 non-null  float64
45  KWH SQFT 1ST QUARTILE 2010           65901 non-null  float64
46  KWH SQFT 2ND QUARTILE 2010           65901 non-null  float64
47  KWH SQFT 3RD QUARTILE 2010           65901 non-null  float64
48  KWH SQFT MAXIMUM 2010                65901 non-null  float64
49  THERM MEAN 2010                      65755 non-null  float64
50  THERM STANDARD DEVIATION 2010        56821 non-null  float64
51  THERM MINIMUM 2010                   65755 non-null  float64
```

```
 52  THERM 1ST QUARTILE 2010              65755 non-null  float64
 53  THERM 2ND QUARTILE 2010              65755 non-null  float64
 54  THERM 3RD QUARTILE 2010              65755 non-null  float64
 55  THERM MAXIMUM 2010                   65755 non-null  float64
 56  THERMS SQFT MEAN 2010                65378 non-null  float64
 57  THERMS SQFT STANDARD DEVIATION 2010  51367 non-null  float64
 58  THERMS SQFT MINIMUM 2010             65378 non-null  float64
 59  THERMS SQFT 1ST QUARTILE 2010        65378 non-null  float64
 60  THERMS SQFT 2ND QUARTILE 2010        65378 non-null  float64
 61  THERMS SQFT 3RD QUARTILE 2010        65378 non-null  float64
 62  THERMS SQFT MAXIMUM 2010             65378 non-null  float64
 63  TOTAL POPULATION                     67037 non-null  float64
 64  TOTAL UNITS                          67037 non-null  float64
 65  AVERAGE STORIES                      67051 non-null  float64
 66  AVERAGE BUILDING AGE                 67051 non-null  float64
 67  AVERAGE HOUSESIZE                    67037 non-null  float64
 68  OCCUPIED UNITS                       67037 non-null  float64
 69  OCCUPIED UNITS PERCENTAGE            64606 non-null  float64
 70  RENTER-OCCUPIED HOUSING UNITS        67037 non-null  float64
 71  RENTER-OCCUPIED HOUSING PERCENTAGE   64433 non-null  float64
 72  OCCUPIED HOUSING UNITS               67037 non-null  float64
dtypes: float64(69), int64(1), object(3)
memory usage: 37.3+ MB
```

Las columnas más importantes son: - **COMMUNITY AREA NAME**: nombre de un área de Chicago - **CENSUS BLOCK**: número censal. Los registros en blanco son zonas a las que no se agregaron información por privacidad - **BUILDING TYPE**: tipo de edificios en ese área (Residential, Commercial, Industrial). Los registros en blanco son zonas con número censal en blanco. - **BUILDING_SUBTYPE**: subtipo de edificio en ese área (Single Family, Multi <7, Multi 7+, Commercial, Industrial, Municipal). Los registros en blanco son zonas con número censal en blanco. - **KWH JANUARY 2010 ... KWH DECEMBER 2010**: KWH consumidos en cada mes en ese área. - **TOTAL KWH**: Total de KWH consumidos - **ELECTRICITY ACCOUNTS**: número de contadores en la zona. Cada contador no corresponde a un edificio. - **ZERO KWH ACCOUNTS**: contadores que no han consumido energía. - **THERM JANUARY 2010 ... THERM DECEMBER 2010**: therms consumidos en cada mes en ese área. - **TOTAL THERMS**: consumo total de gas. - **GAS ACCOUNTS**: contadores de gas. - **TOTAL POPULATION**: tamaño de la población - **TOTAL UNITS**: Número de viviendas/comercios/fábricas según el tipo de zona. - **AVERAGE BUILDING AGE**: edad media de los edificios - **AVERAGE HOUSESIZE**: dimensión media del hogar. Se obtiene dividiendo el número de personas en los hogares por el número de hogares. - **OCCUPIED UNITS**: edificios ocupados - **OCCUPIED UNITS PERCENTAGE**: porcentaje de edificios ocupados - **RENTER-OCCUPIED HOUSING UNITS**: unidades rentadas - **RENTER-OCCUPIED HOUSING PERCENTAGE**: porcentaje de unidades rentadas

Las demás columnas están descritas en: https://data.cityofchicago.org/Environment-Sustainable-Development/Energy-Usage-2010/8yq3-m6wp

1.2.1 Estudio de datos

La clase ***dataloader*** facilita la selección de datos no nulos del *dataset*. A través de las propiedades *energy_cols* y *gas_cols* podemos seleccionar todas las columnas que reflejan el consumo mensual de electricidad y gas, respectivamente.

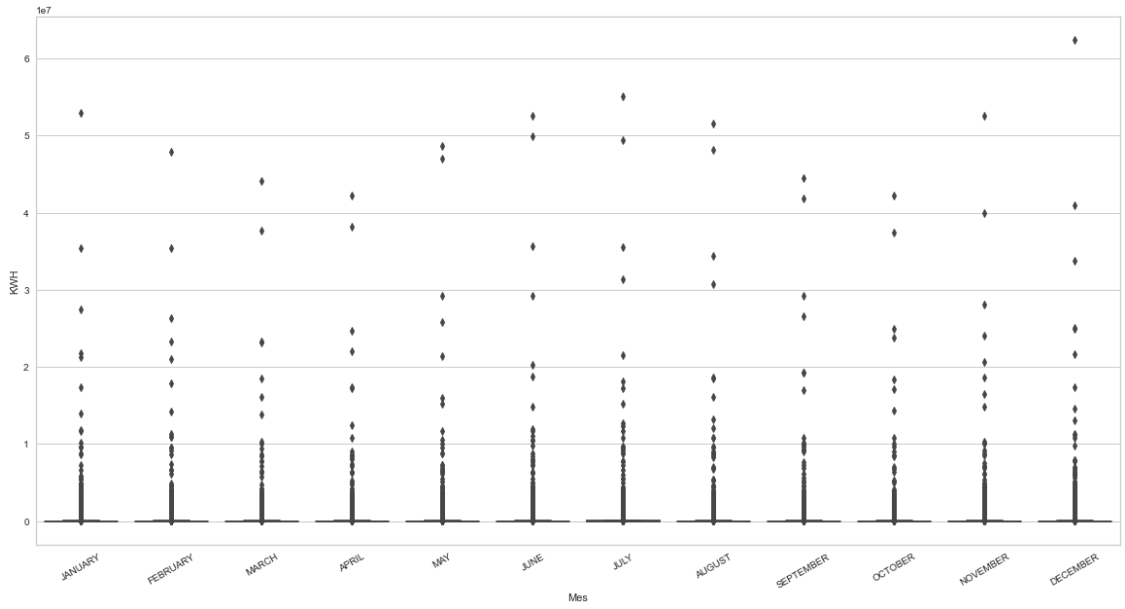
[4]:
```
dl = DataLoader(df)
```

[5]:
```
all_energy_df = dl[dl.energy_cols]
all_gas_df = dl[dl.gas_cols]
```

Outliers

Energía El siguiente *boxplot* muestra la existencia de *outliers*, que corresponden a comunidades con un alto consumo de electricidad.

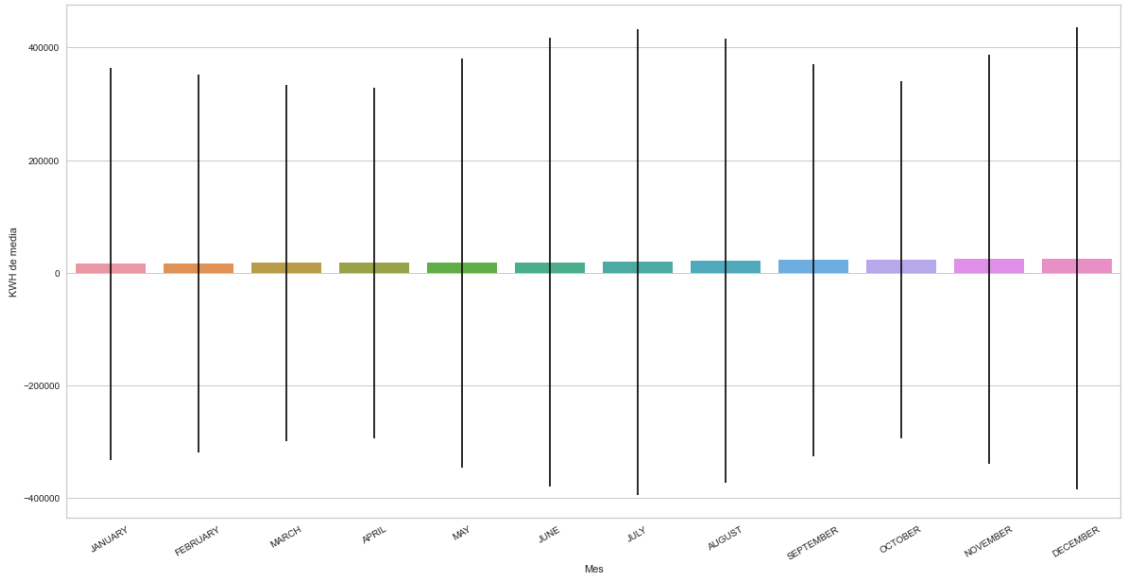
[6]:
```
figure = plt.figure(figsize = (20,10))

ax = sns.boxplot(data=all_energy_df)
_ = ax.set_xticklabels([f"{datetime.date(2008, i, 1).strftime('%B').upper()}"␣
 ↪for i in range(1, 13)], rotation=30)
_ = ax.set(xlabel="Mes", ylabel="KWH")

path = images_path / Path("exploracion_datos/outliers/energia")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("boxplot.eps"))
```



Los valores más altos de consumo hacen notar que en los meses de vacaciones (navidades y verano)

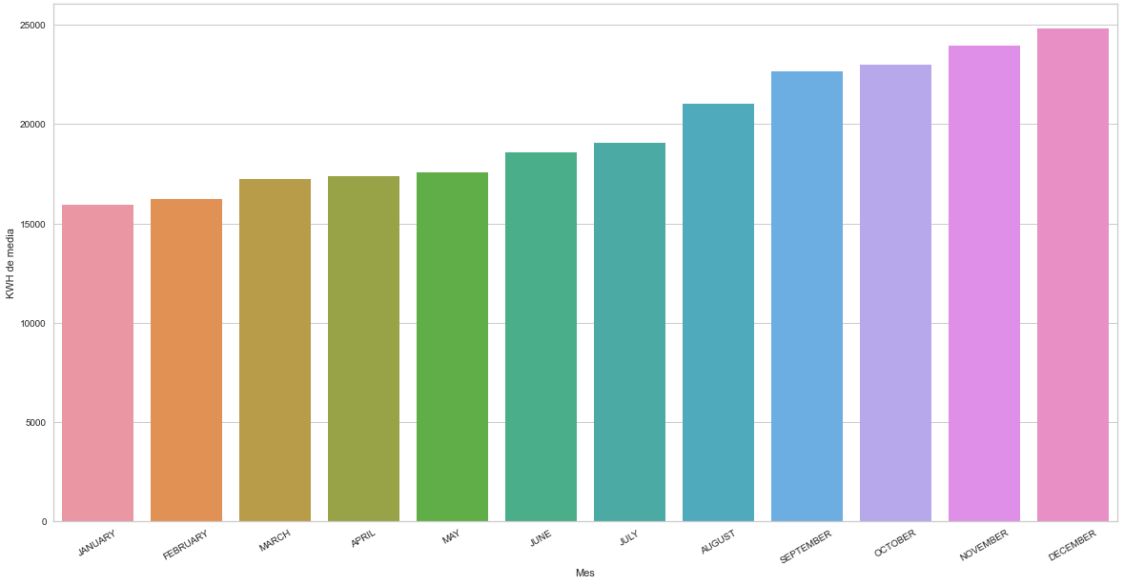
aumenta el consumo de energía. Aún así, según el valor medio se observa un crecimiento del consumo de energía. Sería interesante tener los datos de siguientes años para poder comprobar si esta tendencia continúa en los siguientes años.

[7]:
```
figure = plt.figure(figsize = (20,10))

energy_all_mean = all_energy_df.mean(axis=0)
energy_all_sd = all_energy_df.std(axis=0)
ax = sns.barplot(data=energy_all_mean, x = energy_all_mean.index, y =␣
 ↪energy_all_mean.values, yerr=energy_all_sd)
_ = ax.set_xticklabels([f"{datetime.date(2008, i, 1).strftime('%B').upper()}"␣
 ↪for i in range(1, 13)], rotation=30)
_ = ax.set(xlabel="Mes", ylabel="KWH de media")

path = images_path / Path("exploracion_datos/outliers/energia")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("barplot.eps"))
```



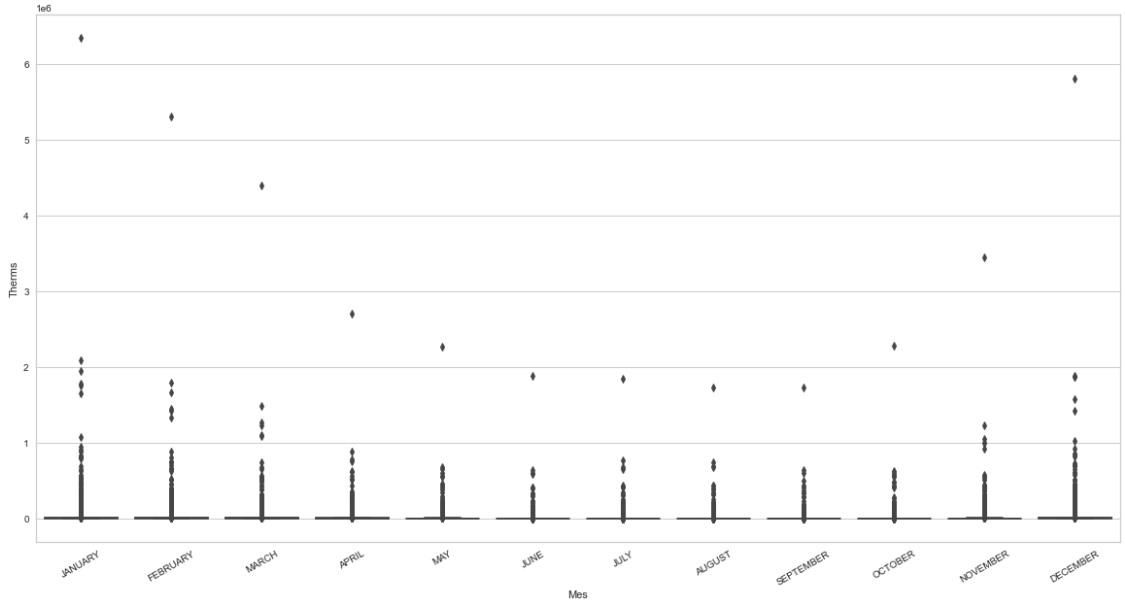
[8]:
```
figure = plt.figure(figsize = (20,10))

energy_all_mean = all_energy_df.mean(axis=0)
ax = sns.barplot(data=energy_all_mean, x = energy_all_mean.index, y =␣
 ↪energy_all_mean.values)
_ = ax.set_xticklabels([f"{datetime.date(2008, i, 1).strftime('%B').upper()}"␣
 ↪for i in range(1, 13)], rotation=30)
_ = ax.set(xlabel="Mes", ylabel="KWH de media")

path = images_path / Path("exploracion_datos/outliers/energia")
```

```
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("barras.eps"))
```



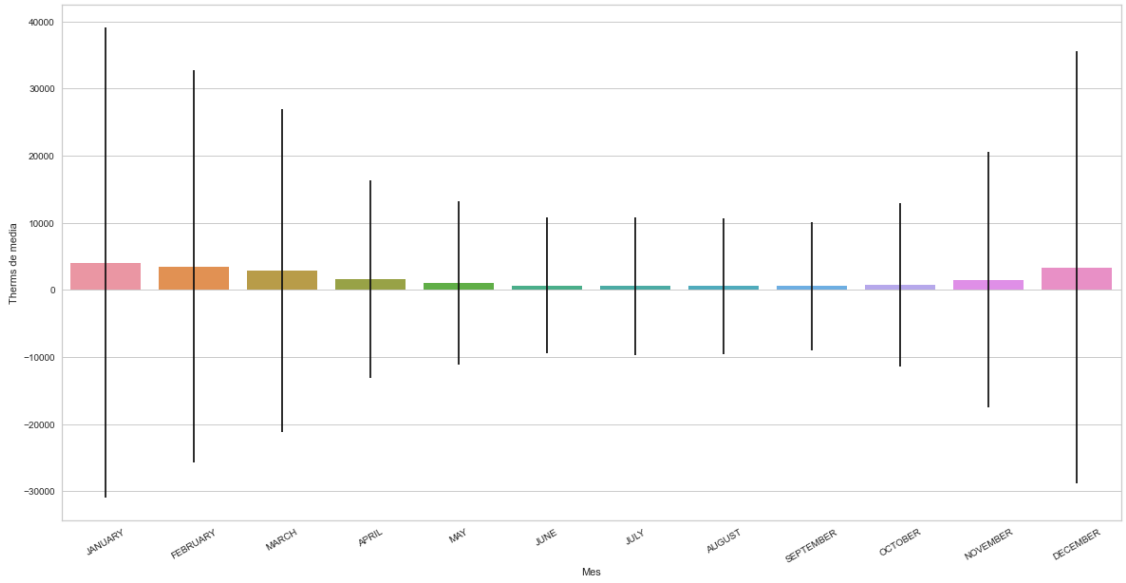
Gas Al igual que con la electricidad, el siguiente *boxplot* muestra la existencia de *outliers*, que corresponden a comunidades con un alto consumo de gas.

[9]:
```
figure = plt.figure(figsize = (20,10))

ax = sns.boxplot(data=all_gas_df)
_ = ax.set_xticklabels([f"{datetime.date(2008, i, 1).strftime('%B').upper()}"␣
 ↪for i in range(1, 13)], rotation=30)
_ = ax.set(xlabel="Mes", ylabel="Therms")

path = images_path / Path("exploracion_datos/outliers/gas")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("boxplot.svg"))
```

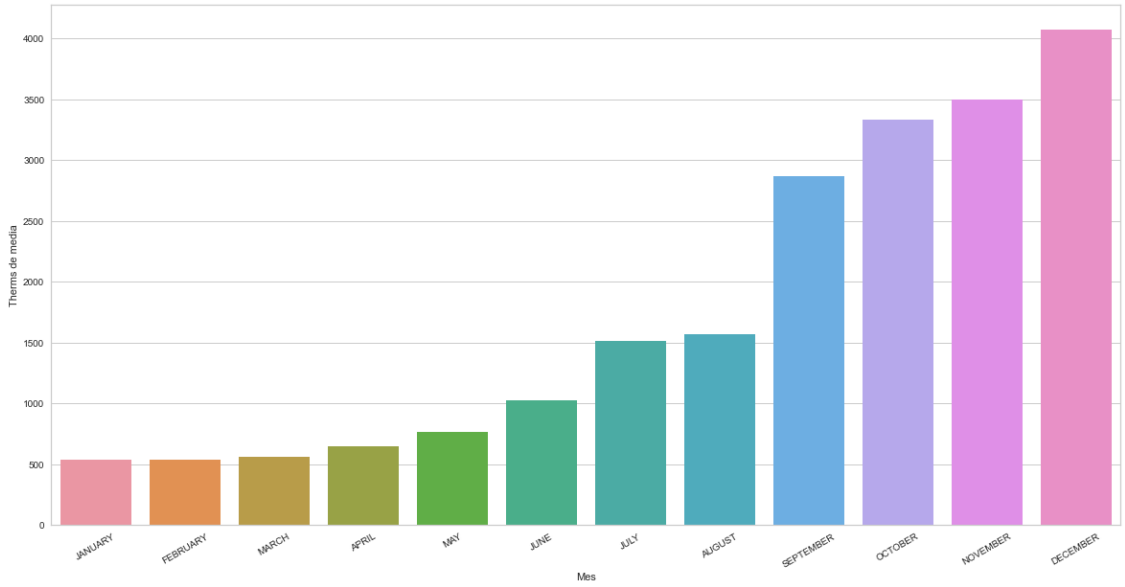


Los valores más altos de varianza reflejan la tendencia de uso que se podría esperar, que se utilice más el gas en épocas invernales. Aún así, el valor medio refleja justo lo opuesto, el consumo de gas tiende a aumentar.

[10]:
```
figure = plt.figure(figsize = (20,10))

gas_all_mean = all_gas_df.mean(axis=0)
gas_all_sd = all_gas_df.std(axis=0)
ax = sns.barplot(data=energy_all_mean, x = gas_all_mean.index, y = gas_all_mean.
 ↪values, yerr=gas_all_sd)
_ = ax.set_xticklabels([f"{datetime.date(2008, i, 1).strftime('%B').upper()}"␣
 ↪for i in range(1, 13)], rotation=30)
_ = ax.set(xlabel="Mes", ylabel="Therms de media")

path = images_path / Path("exploracion_datos/outliers/gas")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("barplot.svg"))
```



[11]:
```
figure = plt.figure(figsize = (20,10))

gas_all_mean = all_gas_df.mean(axis=0)
ax = sns.barplot(data=gas_all_mean, x = gas_all_mean.index, y = gas_all_mean.
 ↪values)
_ = ax.set_xticklabels([f"{datetime.date(2008, i, 1).strftime('%B').upper()}"␣
 ↪for i in range(1, 13)], rotation=30)
_ = ax.set(xlabel="Mes", ylabel="Therms de media")

path = images_path / Path("exploracion_datos/outliers/gas")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("barras.svg"))
```



Comunidades que existen En el dataset se encuentran las 77 comunidades de Chicago.

[12]:
```
dl['COMMUNITY AREA NAME'].drop_duplicates()
```

[12]:
```
0          Archer Heights
1                 Ashburn
2          Auburn Gresham
3                  Austin
12               Avondale
              …
591             West Elsdon
602          South Deering
607    Washington Heights
736       Mount Greenwood
740       Near South Side
Name: COMMUNITY AREA NAME, Length: 77, dtype: object
```

1.2.2 Volver al inicio

1.3 Ensayos

1.3.1 Clustering de energía por tipo de zonas agrupando por ciudades

Se va a calcular el valor medio de consumo eléctrico por cada una de las comunidades y buscar las posibles agrupaciones de consumo para cada tipo de zona en la ciudad (comercial, industrial y residencial).

A continuación se calcula la media de consumo eléctrico para cada zona.

[13]:
```
energy_mean_by_community_type = dl[['COMMUNITY AREA NAME', 'BUILDING TYPE'] +␣
↪dl.energy_cols].groupby(['COMMUNITY AREA NAME', 'BUILDING TYPE'],␣
↪as_index=False).mean()
energy_mean_by_community_type
```

[13]:

	COMMUNITY AREA NAME	BUILDING TYPE	KWH JANUARY 2010	KWH FEBRUARY 2010	KWH MARCH 2010	KWH APRIL 2010	KWH MAY 2010	KWH JUNE 2010	KWH JULY 2010	KWH AUGUST 2010	KWH SEPTEMBER 2010	KWH OCTOBER 2010	KWH NOVEMBER 2010	KWH DECEMBER 2010
0	Albany Park	Commercial	6485.973190	6598.123324	6350.994638	6565.369973	8521.268097	10648.798928	10824.525469	9204.222520	7193.860590	7063.611260		
1	Albany Park	Industrial	4387.000000	3671.000000	4849.000000	4923.000000	6580.000000	8196.000000	7074.000000	7556.000000	4655.000000	6193.000000		
2	Albany Park	Residential	4493.482445	4868.560468	4831.300390	4704.583875	6202.847854	9507.514954	10494.762029	7880.837451	5533.161248	5713.767230		
3	Archer Heights	Commercial	76154.428571	77902.155844	74525.714286	72982.935065	76074.220779	85996.129870	84604.389610	85607.376623	77916.636364	73602.805195		
4	Archer Heights	Industrial	1100.000000	634.000000	508.000000	456.000000	552.000000	829.000000	4405.000000	1304.000000	895.000000	657.000000		
..	…	…	…	…	…	…	…	…	…	…	…	…		
168	West Town	Commercial	7890.108716	8219.340206	7867.381443	7907.283037	9768.865979	12332.753515	13705.432052	13291.417057	10410.490159	9151.462980		
169	West Town	Industrial	7642.000000	6762.000000	5174.000000	6681.000000	8466.000000	16203.000000	16007.000000	13908.000000	9200.000000	7256.000000		
170	West Town	Residential	4035.878754	4627.875921	4392.944476	4392.706516	5338.343909	7254.643626	9278.979603	9354.343909	7011.481020	5714.180737		
171	Woodlawn	Commercial	66321.169231	62376.574359	46025.943590	44388.046154	41310.712821	48550.261538	44273.635897	52702.687179	59037.825641	50860.584615		
172	Woodlawn	Residential	4064.865482	4168.939086	4120.944162	4000.304569	4449.756345	5899.697970	7016.401015	6889.312183	5007.302030	5081.281726		

```
0          8875.603217        9536.718499
1         12597.000000        8237.000000
2          8412.635891        9043.319896
3         76310.571429       84052.545455
4           784.000000        1586.000000
..                  …                  …
168       10051.944705       11954.955014
169        8621.000000       10312.000000
170        6840.808499        8876.675921
171       53246.569231       76645.671795
172        6718.832487        7993.091371

[173 rows x 14 columns]
```

Comercial Se seleccionan los valores medio de consumo para las zonas comerciales.

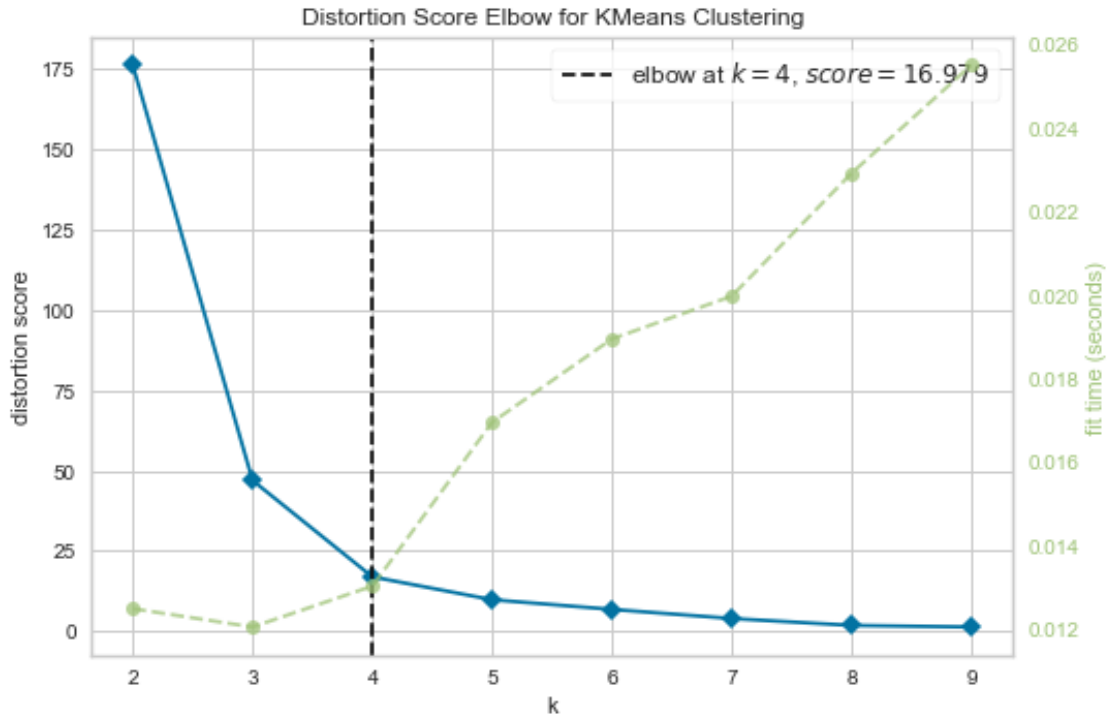
[14]:
```
energy_commercial_by_community =␣
↪energy_mean_by_community_type[energy_mean_by_community_type['BUILDING␣
↪TYPE']=="Commercial"]
energy_commercial_by_community
```

[14]:
```
    COMMUNITY AREA NAME BUILDING TYPE  KWH JANUARY 2010  KWH FEBRUARY 2010  \
0             Albany Park    Commercial       6485.973190        6598.123324
3           Archer Heights    Commercial      76154.428571       77902.155844
6            Armour Square    Commercial      37698.041667       34363.145833
9                  Ashburn    Commercial      24129.737374       22406.404040
11          Auburn Gresham    Commercial       5184.100386        5056.308880
..                     …           …                …                  …
162             West Lawn    Commercial      65512.178947       71017.073684
164           West Pullman    Commercial       4018.162791        4657.023256
166             West Ridge    Commercial       9307.416335        9402.619522
168              West Town    Commercial       7890.108716        8219.340206
171               Woodlawn    Commercial      66321.169231       62376.574359

     KWH MARCH 2010  KWH APRIL 2010  KWH MAY 2010  KWH JUNE 2010  \
0       6350.994638     6565.369973   8521.268097   10648.798928
3      74525.714286    72982.935065  76074.220779   85996.129870
6      30411.239583    31034.666667  31278.093750   33461.270833
9      20967.484848    21031.595960  26618.222222   28948.676768
11      5084.494208     5122.316602   6511.660232    7688.424710
..              …               …             …              …
162    67999.989474    69908.431579  84869.778947   94092.010526
164     4679.779070     4860.197674   5639.651163    6886.476744
166     8822.699203     9619.591633  12595.511952   15000.701195
168     7867.381443     7907.283037   9768.865979   12332.753515
171    46025.943590    44388.046154  41310.712821   48550.261538
```

```
     KWH JULY 2010  KWH AUGUST 2010  KWH SEPTEMBER 2010  KWH OCTOBER 2010  \
0      10824.525469      9204.222520         7193.860590       7063.611260
3      84604.389610     85607.376623        77916.636364      73602.805195
6      42105.614583     39720.145833        32412.708333      23017.114583
9      29027.707071     26762.808081        22696.373737      21852.020202
11      7925.613900      6999.822394         5477.868726       5558.455598
..              …                …                   …                 …
162    91777.000000     92741.736842        78164.905263      66165.336842
164     7874.616279      7650.453488         6234.651163       5450.593023
166    14534.021912     12644.521912         9959.976096      10351.906375
168    13705.432052     13291.417057        10410.490159       9151.462980
171    44273.635897     52702.687179        59037.825641      50860.584615

     KWH NOVEMBER 2010  KWH DECEMBER 2010
0           8875.603217        9536.718499
3          76310.571429       84052.545455
6          31733.833333       48211.739583
9          25317.878788       28463.444444
11          6506.455598        7659.637066
..                  …                  …
162        69139.621053       78315.905263
164         6107.093023        7010.883721
166        12838.211155       13003.069721
168        10051.944705       11954.955014
171        53246.569231       76645.671795

[77 rows x 14 columns]
```

Aplicamos *KMeans* para varios valores de k y seleccionamos el punto codo en la gráfica de *Elbow*.

[15]:
```
kmax = 10
k_values = (2, kmax)

scaler = preprocessing.StandardScaler()
data = energy_commercial_by_community[dl.energy_cols]
norm_data = scaler.fit(data).transform(data)

km = KMeansCluster(norm_data)
km.cluster(k_values)

path = images_path / Path("ensayos/clustering_energia_tipo_zona_agrupa_ciudades/
 ↪comercial")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("elbow.svg"))
```




```
<Figure size 576x396 with 0 Axes>
```

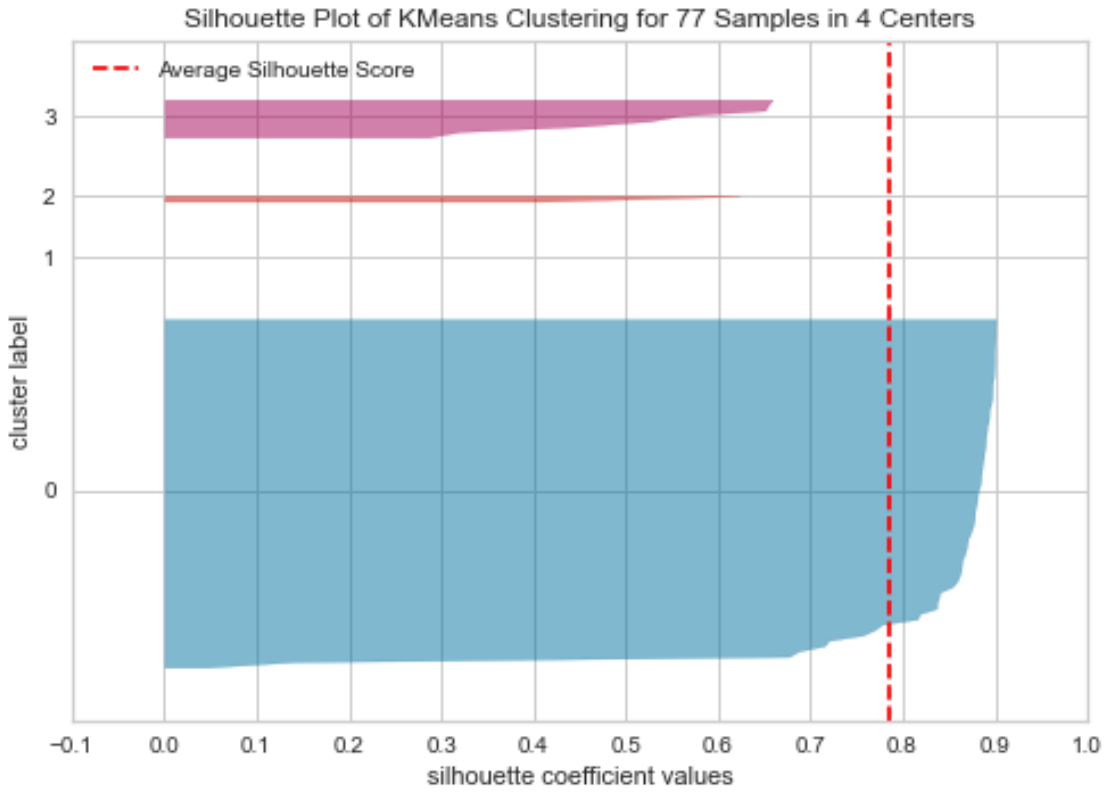
Seleccionamos el mejor resultado de clustering. El punto de codo está sobre k=4.

[16]:
```
km_best = km.get_kmeans_of(4)
km_labels = km_best.labels_
km_centroids = scaler.inverse_transform(km_best.cluster_centers_)

km_labels, np.where(km_labels==0)[0]
```

[16]:
```
(array([0, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0,
        0, 0, 0, 2, 2, 3, 0, 0, 0, 0, 0, 3, 0, 0, 3, 3, 0, 0, 0, 3, 0, 0,
        0, 0, 0, 0, 0, 0, 3, 0, 0, 0, 0]),
 array([ 0,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16, 17,
        18, 19, 20, 21, 22, 23, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35,
        36, 37, 38, 39, 40, 42, 43, 44, 45, 46, 50, 51, 52, 53, 54, 56, 57,
        60, 61, 62, 64, 65, 66, 67, 68, 69, 70, 71, 73, 74, 75, 76],
       dtype=int64))
```

[17]:
```
_ = silhouette_visualizer(km_best, norm_data, colors='yellowbrick')

path = images_path / Path("ensayos/clustering_energia_tipo_zona_agrupa_ciudades/
↪comercial")
```

```
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("silhouette.svg"))
```

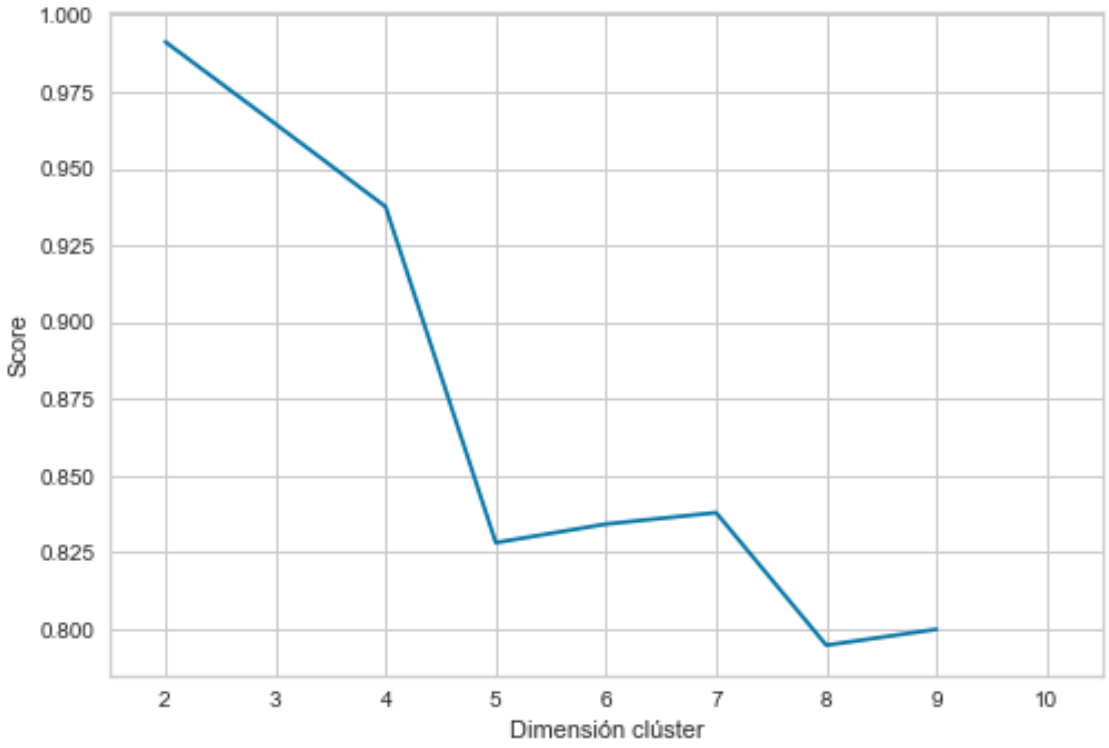



```
<Figure size 576x396 with 0 Axes>
```

A continuación aplicamos *CMeans*.

[18]:
```
kmax = 10
k_values = [e for e in range(2, kmax)]

scaler = preprocessing.StandardScaler()
data = energy_commercial_by_community[dl.energy_cols]
norm_data = scaler.fit(data).transform(data)

cm = CMeansCluster(norm_data)
cm.cluster(k_values)

scores = cm.get_scores()
ax = sns.lineplot(data=scores, x="cluster_dim", y="score")
_ = ax.set(xlabel="Dimensión clúster", ylabel="Score")
_ = ax.set(xlim=(1.5, kmax + .5))
```

```
path = images_path / Path("ensayos/clustering_energia_tipo_zona_agrupa_ciudades/
↪comercial")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("elbow_cmeans.svg"))
```



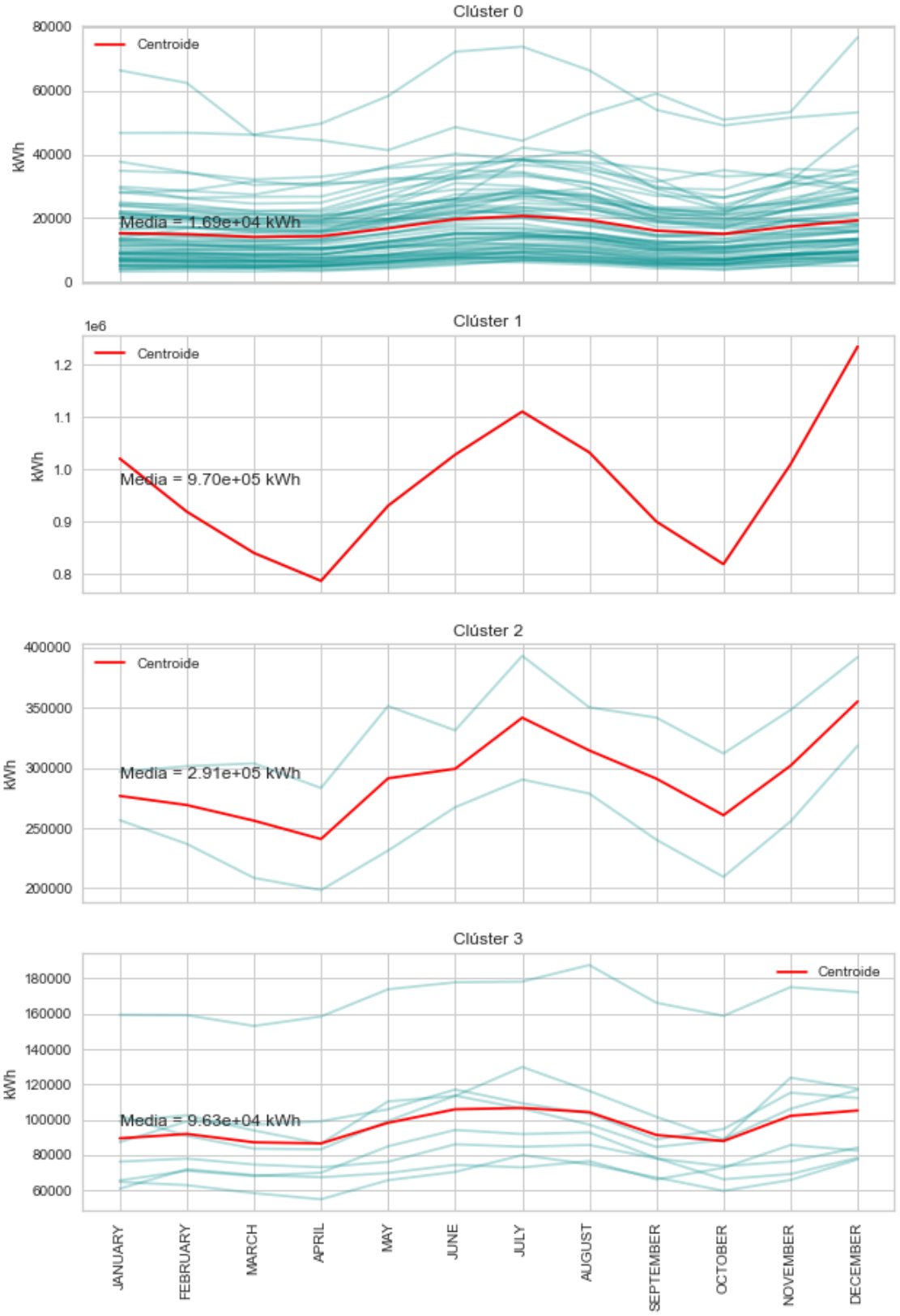
[19]:
```
cm_best = cm.get_cmeans_of(2)
cntr, u, u0, d, jm, p, fpc = cm_best

labels = u.argmax(axis=0)

labels, np.where(labels==1)[0]
```

[19]:
```
(array([1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
        1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1,
        1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
        1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1], dtype=int64),
 array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16,
        17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33,
        34, 35, 36, 37, 38, 39, 40, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51,
        52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68,
        69, 70, 71, 72, 73, 74, 75, 76], dtype=int64))
```

Con Cmeans hemos obtenido una división de una muestra en un único clúster.

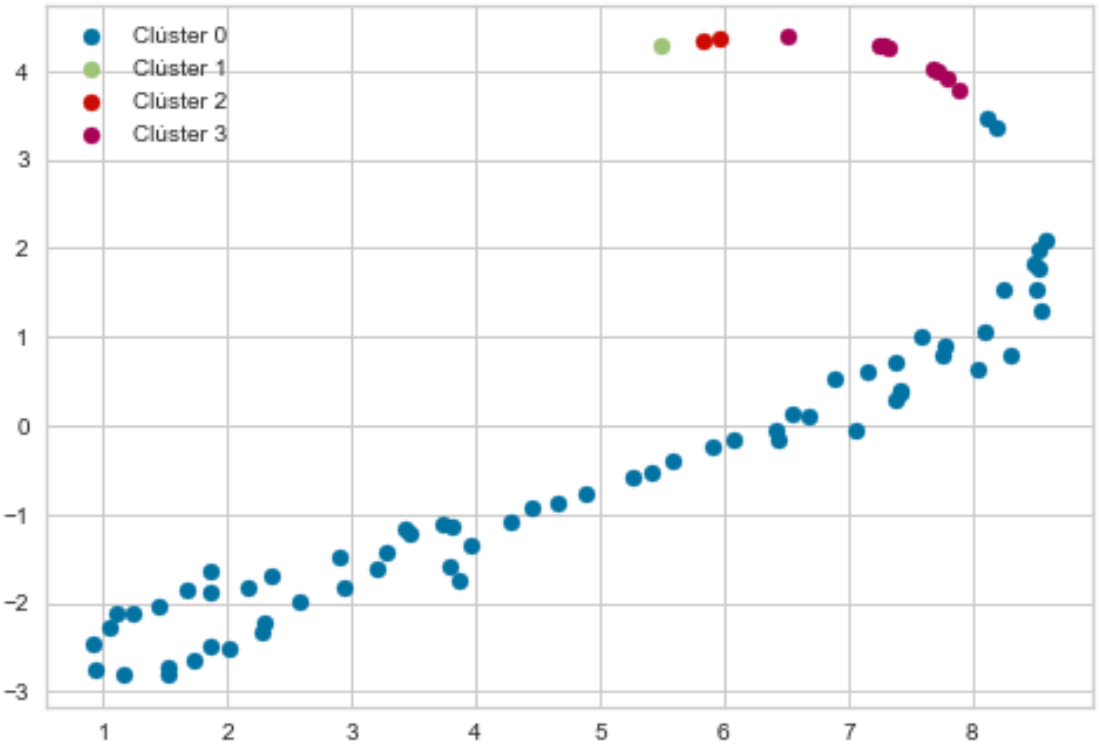
En la siguiente gráfica se muestran los 4 clústers obtenidos con KMeans. *Loop* se encuentra en un único clúster y su consumo medio es el más alto. Este resultado era de esperable pues es, junto a *Near North Side* y *Near South Side*, las comunidades centrales de Chicago.

[20]:
```
labels = pd.DataFrame(km_labels, columns=["label"])

k = km_centroids.shape[0]

f, axes = plt.subplots(k, 1, figsize=(10, 15), sharex=True)

for ki in range(k):
    cluster = energy_commercial_by_community.iloc[labels.query(f'label ==␣
 ↪{ki}').index][dl.energy_cols]
    palette={i:"darkcyan" for i in range(cluster.shape[0])}

    ax0 = sns.lineplot(data=cluster.values.T, ax=axes[ki], palette=palette,␣
 ↪alpha=0.3)
    _ = [line.set_linestyle("-") for line in ax0.lines]

    ax0c = sns.lineplot(data=km_centroids[ki, :].T, ax=axes[ki], color="red")
    _ = [line.set_linestyle("-") for line in ax0c.lines]

    ax0c.legend(ax0c.lines[-1:], ["Centroide"]);

    ax0.set_title(f"Clúster {ki}")
    Media = km_centroids[ki, :].mean()
    ax0.text(0, Media, f"{Media = :0.2e} kWh")
    ax0.set(ylabel="kWh")

    ax0.set_xticks([e for e in range(0, 12)])
    ax0.set_xticklabels([f"{datetime.date(2008, i, 1).strftime('%B').upper()}"␣
 ↪for i in range(1, 13)], rotation=90)

path = images_path / Path("ensayos/clustering_energia_tipo_zona_agrupa_ciudades/
 ↪comercial")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("cluster_grafica.svg"))
```


Clúster 0
Centroide
kWh
Media = 1.69e+04 kWh
Clúster 1
1e6
Centroide
kWh
Media = 9.70e+05 kWh
Clúster 2
Centroide
kWh
Media = 2.91e+05 kWh
Clúster 3
Centroide
kWh
Media = 9.63e+04 kWh
JANUARY
FEBRUARY
MARCH
APRIL
MAY
JUNE
JULY
AUGUST
SEPTEMBER
OCTOBER
NOVEMBER
DECEMBER

La siguiente gráfica nos muestra la división de los clúster encontrados.

[21]:
```
v = tsne(energy_commercial_by_community[dl.energy_cols], 2)
for label in range(k):
    plt.scatter(
            x=v[(labels['label']==label).values, 0],
            y=v[(labels['label']==label).values, 1]
    )
plt.legend([f"Clúster {e}" for e in range(k)])

path = images_path / Path("ensayos/clustering_energia_tipo_zona_agrupa_ciudades/
↪comercial")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("cluster_scatter.svg"))
```



¿Que comunidades pertenecen a cada cluster?

Junto a Loop, Near North Side y Near South Side forman parte del centro de la ciudad de Chicago, por lo que era de esperar que estas ciudades se encontraran entre las de más consumo de energía, en los clúster 1 y 3.

[ ]:
```
comunities_by_cluster = {}

for nlab in range(k):
```

```
    com_index = np.where(labels==nlab)[0]
    com_list = []
    for c in energy_commercial_by_community.values[com_index][:,0]:
        com_list.append(c)
    comunities_by_cluster[nlab] = com_list

for k in sorted(comunities_by_cluster, key=lambda k:␣
↪len(comunities_by_cluster[k])):
    print(f"Al clúster {k} pertenecen las comunidades:")
    for e in comunities_by_cluster[k]:
        print("\t", e)
```

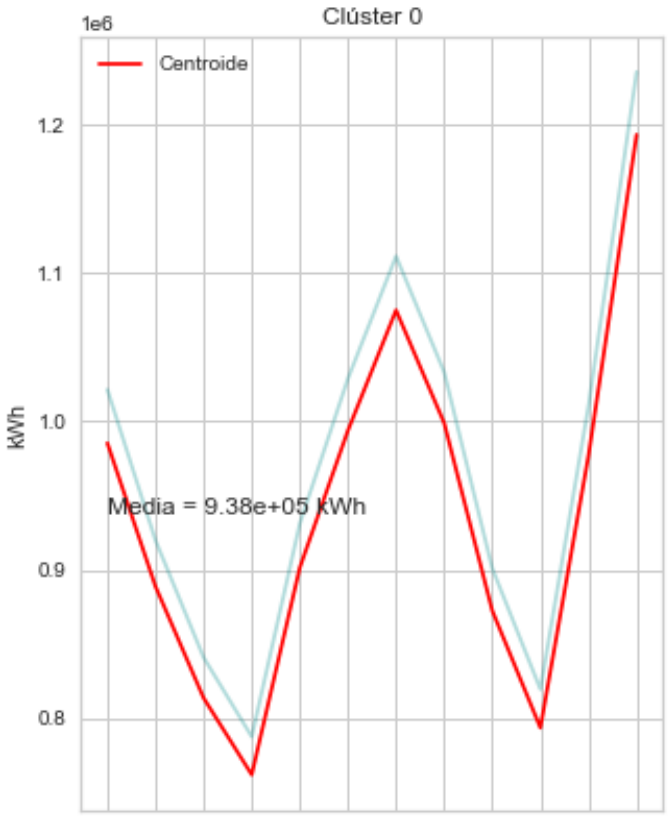
Veamos los resultados de cmeans. Se obtiene dos clústeres, donde en uno hay una única muestra.

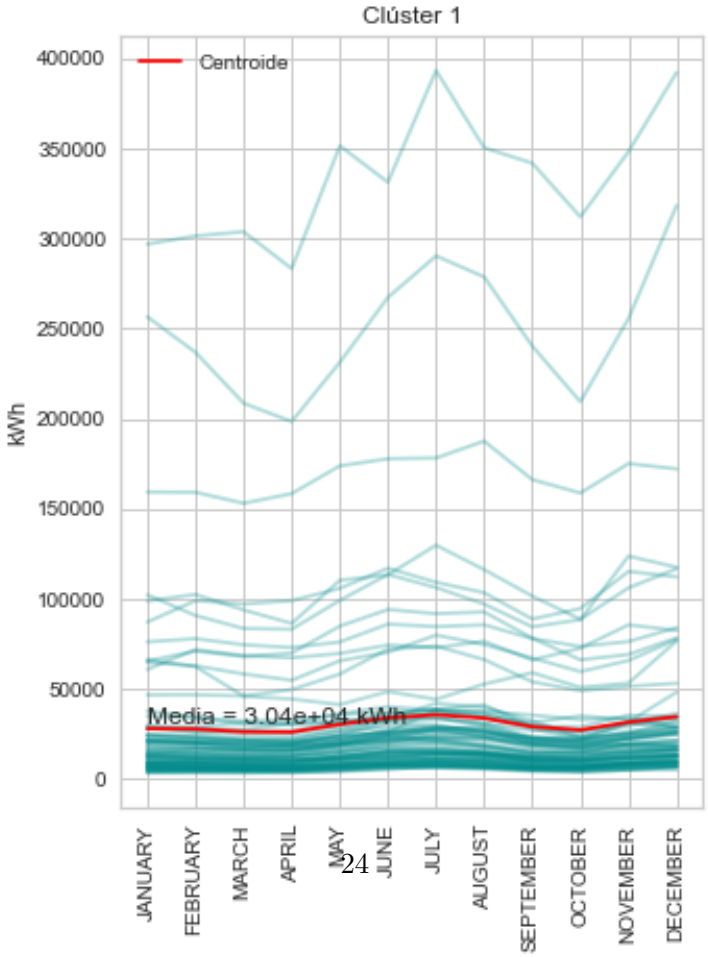
[24]:
```
cm_best = cm.get_cmeans_of(2)
cntr, u, u0, d, jm, p, fpc = cm_best

cntr = scaler.inverse_transform(cntr)
cm_labels = u.argmax(axis=0)

cm_labels
```

[24]:
```
array([1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1], dtype=int64)
```

[25]:
```
labels = pd.DataFrame(cm_labels, columns=["label"])

k = cntr.shape[0]

f, axes = plt.subplots(k, 1, figsize=(5, 15), sharex=True)

for ki in range(k):
    cluster = energy_commercial_by_community.iloc[labels.query(f'label ==␣
↪{ki}').index][dl.energy_cols]
    palette={i:"darkcyan" for i in range(cluster.shape[0])}

    ax0 = sns.lineplot(data=cluster.values.T, ax=axes[ki], palette=palette,␣
↪alpha=0.3)
    _ = [line.set_linestyle("-") for line in ax0.lines]

    ax0c = sns.lineplot(data=cntr[ki, :].T, ax=axes[ki], color="red")
    _ = [line.set_linestyle("-") for line in ax0c.lines]

    ax0c.legend(ax0c.lines[-1:], ["Centroide"]);
```

```
    ax0.set_title(f"Clúster {ki}")
    Media = cntr[ki, :].mean()
    ax0.text(0, Media, f"{Media = :0.2e} kWh")
    ax0.set(ylabel="kWh")

    ax0.set_xticks([e for e in range(0, 12)])
    ax0.set_xticklabels([f"{datetime.date(2008, i, 1).strftime('%B').upper()}"␣
 ↪for i in range(1, 13)], rotation=90)

path = images_path / Path("ensayos/clustering_energia_tipo_zona_agrupa_ciudades/
 ↪comercial")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("cluster_grafica_cmeans.svg"))
```


Clúster 0
1e6
Centroide
1.2
1.1
1.0
0.9
0.8
kWh
Media = 9.38e+05 kWh


Clúster 1
Centroide
400000
350000
300000
250000
200000
150000
100000
50000
0
kWh
Media = 3.04e+04 kWh
JANUARY
FEBRUARY
MARCH
APRIL
MAY
JUNE
JULY
AUGUST
SEPTEMBER
OCTOBER
NOVEMBER
DECEMBER

Residencial A continuación se selecciona los valores medio de consumo por comunidad para las zonas tipo residencial.

[26]:
```
energy_residential_by_community = energy_mean_by_community_type[energy_mean_by_community_type['BUILDING TYPE']=="Residential"]
energy_residential_by_community
```

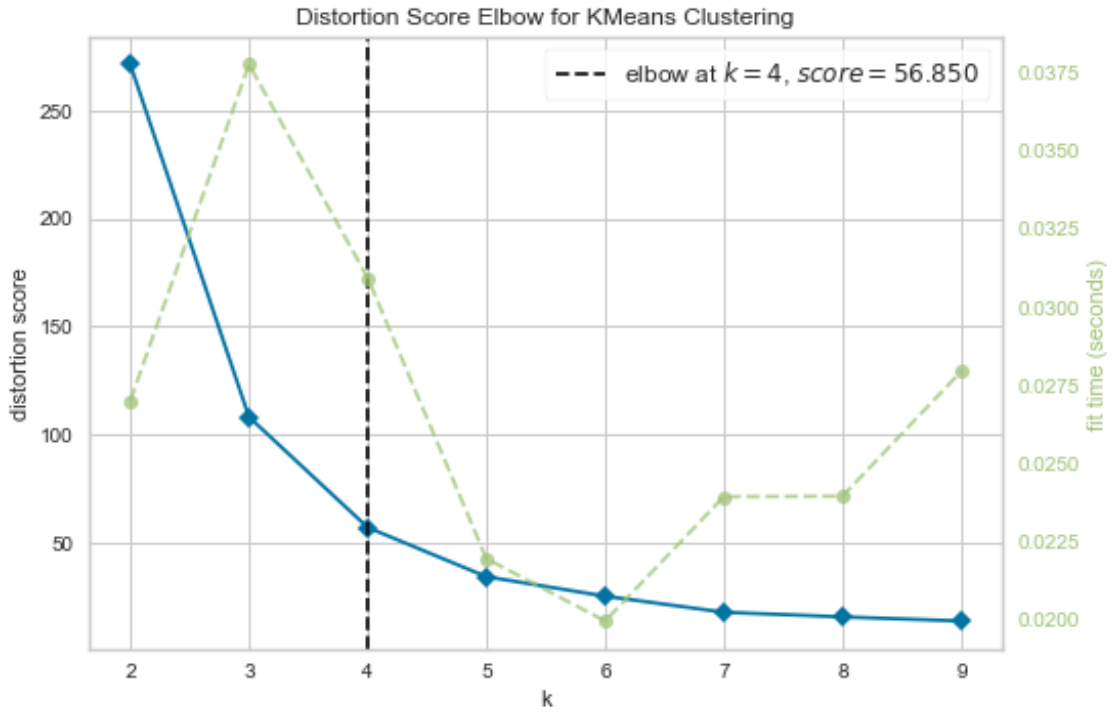
[26]:

	COMMUNITY AREA NAME	BUILDING TYPE	KWH JANUARY 2010	KWH FEBRUARY 2010	KWH MARCH 2010	KWH APRIL 2010	KWH MAY 2010	KWH JUNE 2010	KWH JULY 2010	KWH AUGUST 2010	KWH SEPTEMBER 2010	KWH OCTOBER 2010
2	Albany Park	Residential	4493.482445	4868.560468	4831.300390	4704.583875	6202.847854	9507.514954	10494.762029	7880.837451	5533.161248	5713.767230
5	Archer Heights	Residential	4779.779605	3830.042763	3381.667763	4259.944079	5050.868421	7143.690789	10079.098684	8852.562500	5186.148026	4123.243421
8	Armour Square	Residential	4880.892655	5422.807910	5269.322034	5195.807910	5739.259887	6977.977401	10876.067797	10688.847458	8135.774011	6055.570621
10	Ashburn	Residential	7685.557441	7535.497389	7290.016971	7476.365535	10381.006527	16359.704961	18197.938642	13426.244125	9015.569191	8341.000000
12	Auburn Gresham	Residential	5250.104076	5428.169991	5243.229835	5094.189939	6166.013877	8637.880312	9349.874241	7706.798786	5858.764094	6152.209887
..	…	…	…	…	…	…	…	…	…	…	…	…
163	West Lawn	Residential	5884.403716	5566.481419	5527.908784	5224.923986	7577.138514	11927.521959	13929.736486	11061.168919	6631.606419	5928.351351
165	West Pullman	Residential	2512.937376	4048.336978	3890.841948	4105.110338	4580.604374	5652.351889	7412.451292	7063.981113	5111.528827	4619.525845
167	West Ridge	Residential	5471.784787	5573.705937	5368.393321	5863.319109	7990.311688	11648.390538	11575.739332	8624.476809	5947.495362	6420.602041
170	West Town	Residential	4035.878754	4627.875921	4392.944476	4392.706516	5338.343909	7254.643626				
172	Woodlawn	Residential	4064.865482	4168.939086	4120.944162	4000.304569	4449.756345	5899.697970				

```
170    9278.979603      9354.343909         7011.481020       5714.180737
172    7016.401015      6889.312183         5007.302030       5081.281726

     KWH NOVEMBER 2010  KWH DECEMBER 2010
2          8412.635891        9043.319896
5          5542.542763        7024.115132
8          7107.508475        9459.282486
10        12181.706266       13469.849869
12         8788.036427       10000.882914
..                 …                  …
163        7740.572635        9650.545608
165        5552.450298        7356.383698
167        9243.713358        9150.877551
170        6840.808499        8876.675921
172        6718.832487        7993.091371

[77 rows x 14 columns]
```

Aplicamos *KMeans*

[27]:
```
kmax = 10
k_values = (2, kmax)

scaler = preprocessing.StandardScaler()
data = energy_residential_by_community[dl.energy_cols]
norm_data = scaler.fit(data).transform(data)

km = KMeansCluster(norm_data)
km.cluster(k_values)

path = images_path / Path("ensayos/clustering_energia_tipo_zona_agrupa_ciudades/
↪residencial")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("elbow.svg"))
```



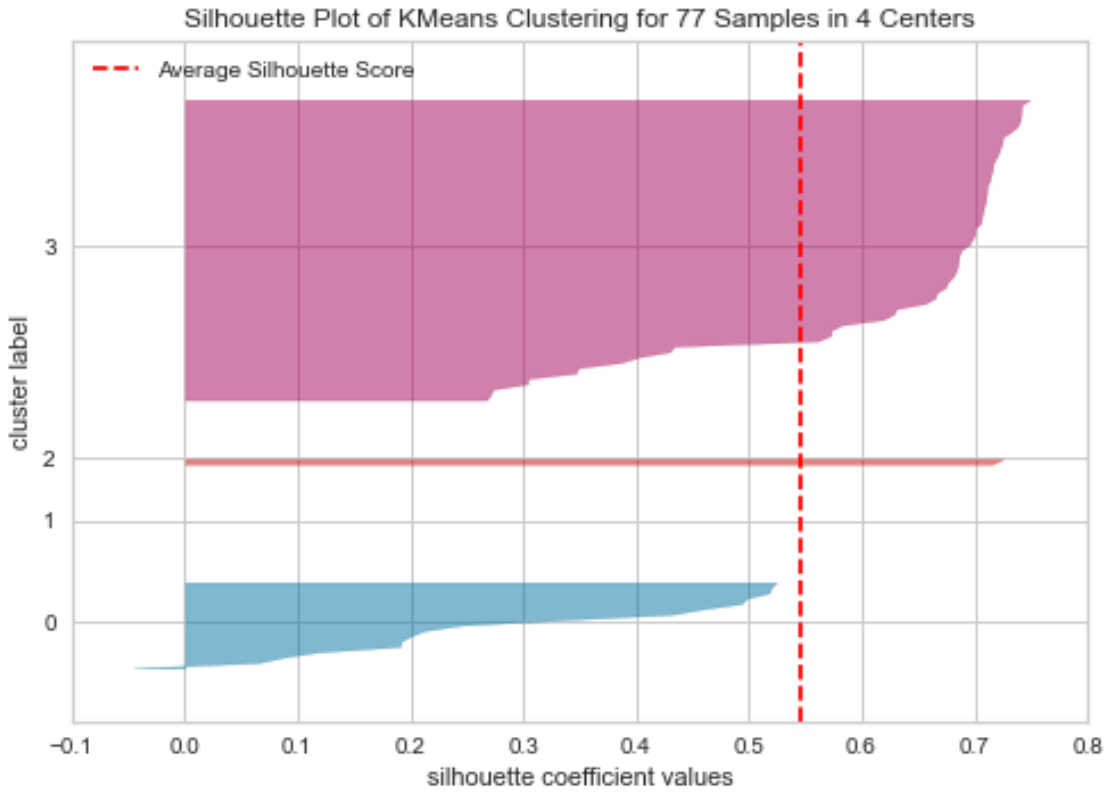

```
<Figure size 576x396 with 0 Axes>
```

En este caso, el punto de codo se encuentra en k=4

[28]:
```
km_best = km.get_kmeans_of(4)
km_labels = km_best.labels_
km_centroids = scaler.inverse_transform(km_best.cluster_centers_)

km_labels
```

[28]:
```
array([3, 3, 3, 0, 3, 3, 3, 3, 3, 0, 3, 3, 3, 3, 3, 3, 0, 0, 0, 3, 3, 3,
       0, 3, 0, 3, 3, 0, 3, 3, 3, 3, 3, 3, 3, 3, 0, 3, 0, 3, 3, 1, 3, 3,
       3, 0, 0, 2, 2, 0, 3, 3, 3, 3, 0, 0, 3, 3, 3, 0, 3, 3, 3, 3, 3, 3,
       0, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3])
```

[29]:
```
_ = silhouette_visualizer(km_best, norm_data, colors='yellowbrick')

path = images_path / Path("ensayos/clustering_energia_tipo_zona_agrupa_ciudades/
 ↪residencial")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("silhouette.svg"))
```

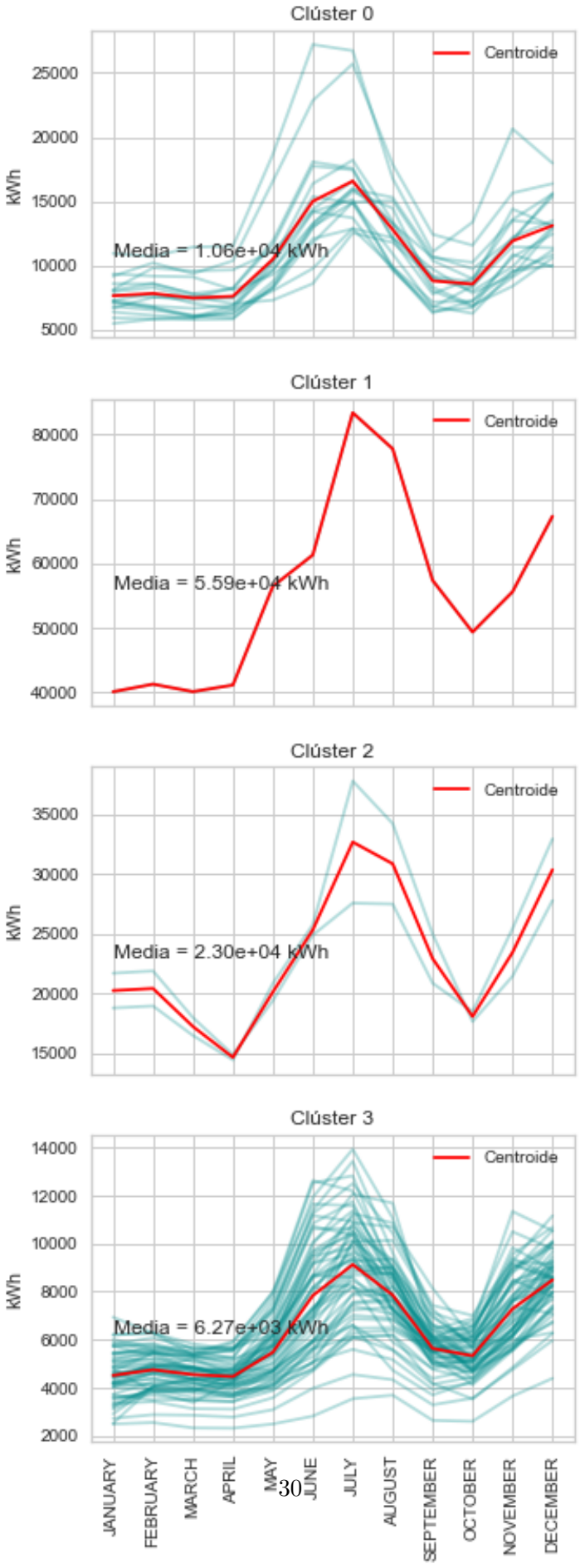



```
<Figure size 576x396 with 0 Axes>
```

En la siguientes gráfica podemos observar las tendencias de cada uno de los clústeres.

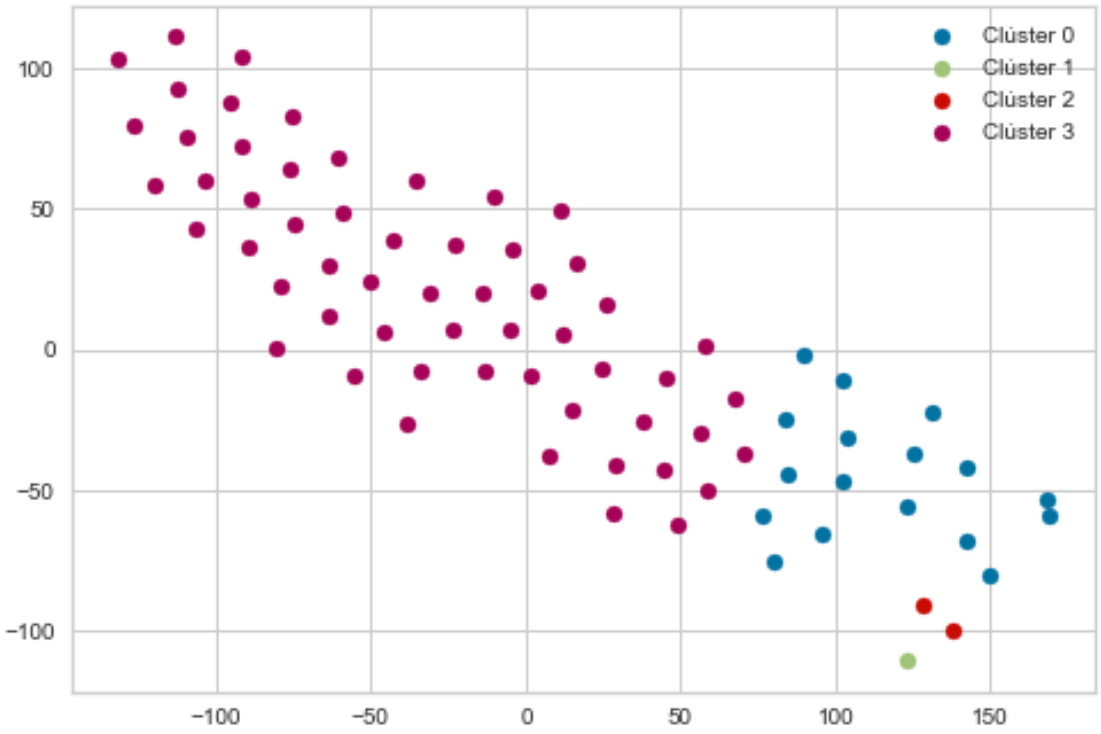
[30]:
```
labels = pd.DataFrame(km_labels, columns=["label"])

k = km_centroids.shape[0]

f, axes = plt.subplots(k, 1, figsize=(5, 15), sharex=True)

for ki in range(k):
    cluster = energy_residential_by_community.iloc[labels.query(f'label ==␣
↪{ki}').index][dl.energy_cols]
    palette={i:"darkcyan" for i in range(cluster.shape[0])}

    ax0 = sns.lineplot(data=cluster.values.T, ax=axes[ki], palette=palette,␣
↪alpha=0.3)
    _ = [line.set_linestyle("-") for line in ax0.lines]

    ax0c = sns.lineplot(data=km_centroids[ki, :].T, ax=axes[ki], color="red")
    _ = [line.set_linestyle("-") for line in ax0c.lines]
```

```
    ax0c.legend(ax0c.lines[-1:], ["Centroide"]);

    ax0.set_title(f"Clúster {ki}")
    Media = km_centroids[ki, :].mean()
    ax0.text(0, Media, f"{Media = :0.2e} kWh")
    ax0.set(ylabel="kWh")

    ax0.set_xticks([e for e in range(0, 12)])
    ax0.set_xticklabels([f"{datetime.date(2008, i, 1).strftime('%B').upper()}"␣
 ↪for i in range(1, 13)], rotation=90)

path = images_path / Path("ensayos/clustering_energia_tipo_zona_agrupa_ciudades/
 ↪residencial")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("cluster_grafica.svg"))
```


Clúster 0
Centroide
kWh
Media = 1.06e+04 kWh
Clúster 1
Centroide
kWh
Media = 5.59e+04 kWh
Clúster 2
Centroide
kWh
Media = 2.30e+04 kWh
Clúster 3
Centroide
kWh
Media = 6.27e+03 kWh
JANUARY
FEBRUARY
MARCH
APRIL
MAY
JUNE
JULY
AUGUST
SEPTEMBER
OCTOBER
NOVEMBER
DECEMBER

En el siguiente gráfico se muestra las muestras agrupadas en cada uno de los clústers.

[33]:
```
v = tsne(energy_residential_by_community[dl.energy_cols], 2)
for label in range(k):
    plt.scatter(
            x=v[(labels['label']==label).values, 0],
            y=v[(labels['label']==label).values, 1]
    )
plt.legend([f"Clúster {i}" for i in range(k)])

path = images_path / Path("ensayos/clustering_energia_tipo_zona_agrupa_ciudades/
 ↪residencial")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("cluster_scatter.svg"))
```



¿Qué comunidad está en cada cluster?

[ ]:
```
comunities_by_cluster = {}

for nlab in range(k):
    com_index = np.where(labels==nlab)[0]
```

```
    com_list = []
    for c in energy_residential_by_community.values[com_index][:,0]:
        com_list.append(c)
    comunities_by_cluster[nlab] = com_list

for k in sorted(comunities_by_cluster, key=lambda k:␣
 ↪len(comunities_by_cluster[k])):
    print(f"Al clúster {k} pertenecen las comunidades:")
    for e in comunities_by_cluster[k]:
        print("\t", e)
```

Industrial Seleccionamos el consumo medio por comunidad para las zonas industriales.

[35]:
```
energy_industrial_by_community =␣
 ↪energy_mean_by_community_type[energy_mean_by_community_type['BUILDING␣
 ↪TYPE']=="Industrial"]
energy_industrial_by_community
```

[35]:

	COMMUNITY AREA NAME	BUILDING TYPE	KWH JANUARY 2010	KWH FEBRUARY 2010	\
1	Albany Park	Industrial	4.387000e+03	3.671000e+03	
4	Archer Heights	Industrial	1.100000e+03	6.340000e+02	
7	Armour Square	Industrial	2.537600e+04	2.299100e+04	
14	Austin	Industrial	4.581100e+04	4.222500e+04	
19	Avondale	Industrial	1.364200e+04	1.302200e+04	
28	Brighton Park	Industrial	0.000000e+00	0.000000e+00	
49	Edgewater	Industrial	3.373000e+03	2.297500e+03	
56	Forest Glen	Industrial	4.909000e+03	4.079000e+03	
61	Gage Park	Industrial	4.140000e+02	8.370000e+02	
72	Hermosa	Industrial	7.989000e+03	1.101300e+04	
79	Irving Park	Industrial	1.262500e+03	1.201000e+03	
84	Kenwood	Industrial	5.748000e+03	5.309000e+03	
87	Lakeview	Industrial	4.662500e+03	4.486500e+03	
94	Logan Square	Industrial	1.196000e+03	1.311333e+03	
111	Near South Side	Industrial	4.288200e+04	1.323340e+05	
114	Near West Side	Industrial	7.081809e+06	7.030604e+06	
121	North Lawndale	Industrial	0.000000e+00	7.670000e+03	
146	South Lawndale	Industrial	2.742770e+05	2.763710e+05	
169	West Town	Industrial	7.642000e+03	6.762000e+03	

	KWH MARCH 2010	KWH APRIL 2010	KWH MAY 2010	KWH JUNE 2010	\
1	4849.0	4.923000e+03	6.580000e+03	8.196000e+03	
4	508.0	4.560000e+02	5.520000e+02	8.290000e+02	
7	18723.0	2.217700e+04	3.171000e+04	3.259000e+04	
14	34700.0	4.234500e+04	3.812000e+04	4.082000e+04	
19	11056.0	1.025400e+04	1.116000e+04	1.404700e+04	
28	0.0	0.000000e+00	0.000000e+00	5.440000e+02	
49	2023.5	3.184500e+03	4.155000e+03	5.877000e+03	

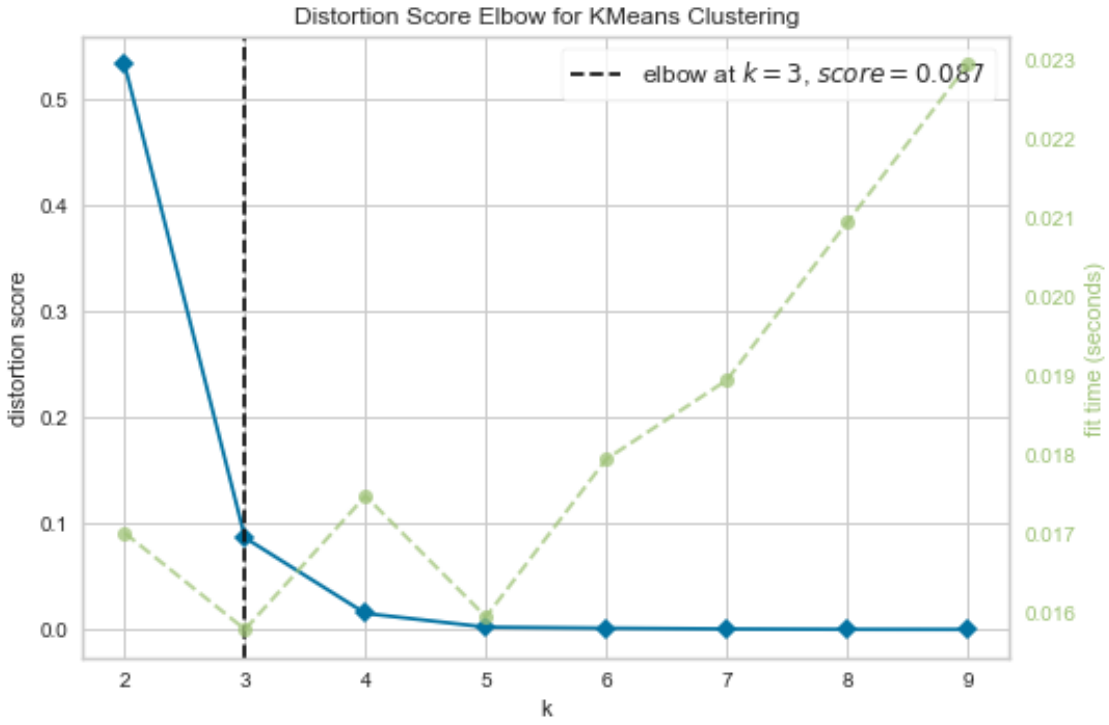
56	6355.0	5.398000e+03	9.727000e+03	1.626400e+04
61	1083.0	1.182000e+03	8.420000e+02	1.140000e+03
72	10109.0	9.970000e+03	9.236000e+03	1.098200e+04
79	1081.0	9.265000e+02	1.410000e+03	2.742500e+03
84	5134.0	5.234000e+03	6.408000e+03	8.681000e+03
87	4072.5	6.243000e+03	7.026000e+03	8.059500e+03
94	1283.0	1.115667e+03	1.338333e+03	1.713000e+03
111	126654.0	1.296920e+05	8.660800e+04	9.994500e+04
114	5381515.0	5.780886e+06	5.327624e+06	6.754392e+06
121	3828.0	1.701000e+03	3.416000e+03	4.208000e+03
146	246830.0	2.764630e+05	2.286560e+05	3.336580e+05
169	5174.0	6.681000e+03	8.466000e+03	1.620300e+04

	KWH JULY 2010	KWH AUGUST 2010	KWH SEPTEMBER 2010	KWH OCTOBER 2010	\
1	7.074000e+03	7.556000e+03	4.655000e+03	6.193000e+03	
4	4.405000e+03	1.304000e+03	8.950000e+02	6.570000e+02	
7	6.834600e+04	5.245100e+04	3.563400e+04	2.720200e+04	
14	4.624200e+04	5.580700e+04	4.414400e+04	4.315900e+04	
19	1.264000e+04	1.122500e+04	1.078100e+04	1.049000e+04	
28	9.760000e+02	7.620000e+02	1.137000e+03	4.780000e+02	
49	5.980000e+03	4.898000e+03	3.381500e+03	2.712000e+03	
56	1.414400e+04	8.681000e+03	5.932000e+03	4.960000e+03	
61	1.468000e+03	3.129000e+03	1.752000e+03	1.122000e+03	
72	1.255600e+04	1.058800e+04	9.284000e+03	8.591000e+03	
79	2.670000e+03	2.321500e+03	1.177000e+03	1.290500e+03	
84	1.065400e+04	1.018900e+04	7.336000e+03	5.350000e+03	
87	7.757000e+03	8.133000e+03	6.719500e+03	5.346000e+03	
94	2.599667e+03	2.688000e+03	1.659667e+03	1.460667e+03	
111	1.491320e+05	1.419530e+05	1.091400e+05	8.356000e+04	
114	5.782397e+06	6.184596e+06	6.439283e+06	5.710067e+06	
121	4.714000e+03	5.175000e+03	4.363000e+03	3.994000e+03	
146	3.289420e+05	3.336430e+05	2.802420e+05	3.353490e+05	
169	1.600700e+04	1.390800e+04	9.200000e+03	7.256000e+03	

	KWH NOVEMBER 2010	KWH DECEMBER 2010
1	1.259700e+04	8.237000e+03
4	7.840000e+02	1.586000e+03
7	2.700300e+04	3.384400e+04
14	4.946300e+04	5.294800e+04
19	1.267200e+04	1.496800e+04
28	6.360000e+02	7.020000e+02
49	4.574000e+03	3.878000e+03
56	7.843000e+03	6.714000e+03
61	1.156000e+03	1.213000e+03
72	1.145800e+04	3.539000e+04
79	1.844000e+03	2.265500e+03
84	5.629000e+03	5.372000e+03

```
87          5.262500e+03        6.658500e+03
94          1.656333e+03        2.131000e+03
111         1.187180e+05        1.740390e+05
114         6.235068e+06        8.365907e+06
121         6.656000e+03        1.142300e+04
146         2.643730e+05        2.832740e+05
169         8.621000e+03        1.031200e+04
```

[36]:
```
kmax = 10
k_values = (2, kmax)

scaler = preprocessing.StandardScaler()
data = energy_industrial_by_community[dl.energy_cols]
norm_data = scaler.fit(data).transform(data)

km = KMeansCluster(norm_data)
km.cluster(k_values)

path = images_path / Path("ensayos/clustering_energia_tipo_zona_agrupa_ciudades/
↪industrial")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("elbow.svg"))
```



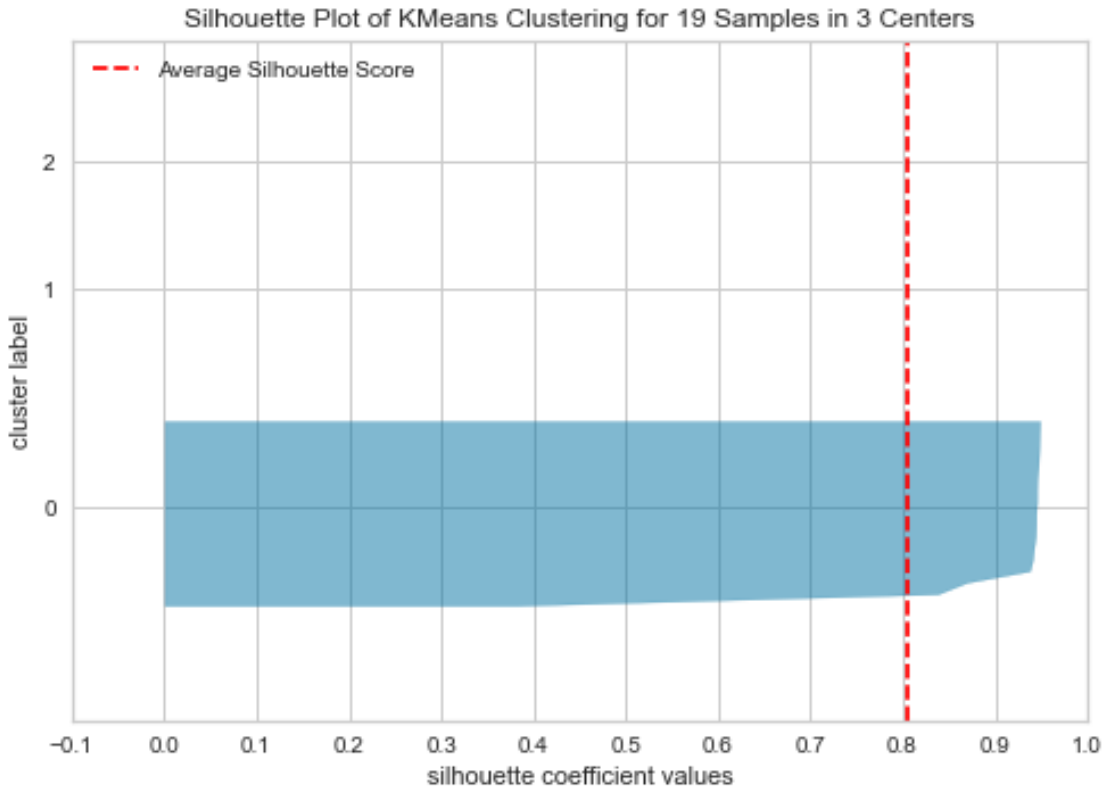

```
<Figure size 576x396 with 0 Axes>
```

Con k=3 tendríamos una división óptima.

[37]:
```
km_best = km.get_kmeans_of(3)
km_labels = km_best.labels_
km_centroids = scaler.inverse_transform(km_best.cluster_centers_)

km_labels
```

[37]:
```
array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 2, 0])
```

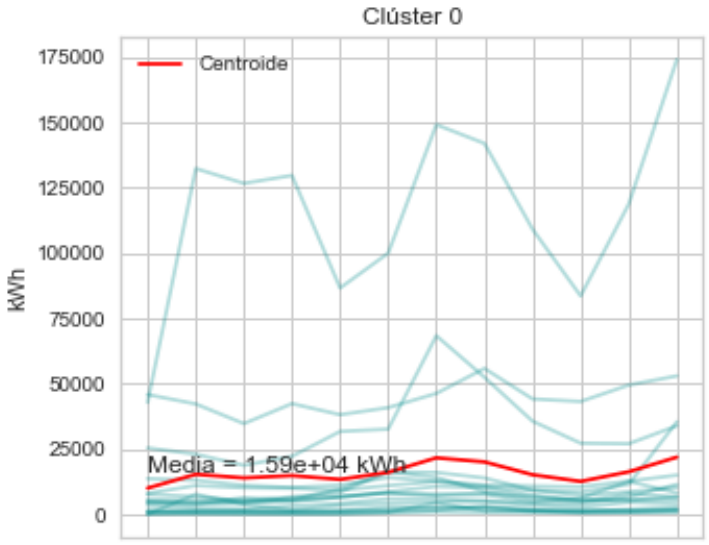
[38]:
```
_ = silhouette_visualizer(km_best, norm_data, colors='yellowbrick')

path = images_path / Path("ensayos/clustering_energia_tipo_zona_agrupa_ciudades/
 ↪industrial")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("silhouette.svg"))
```

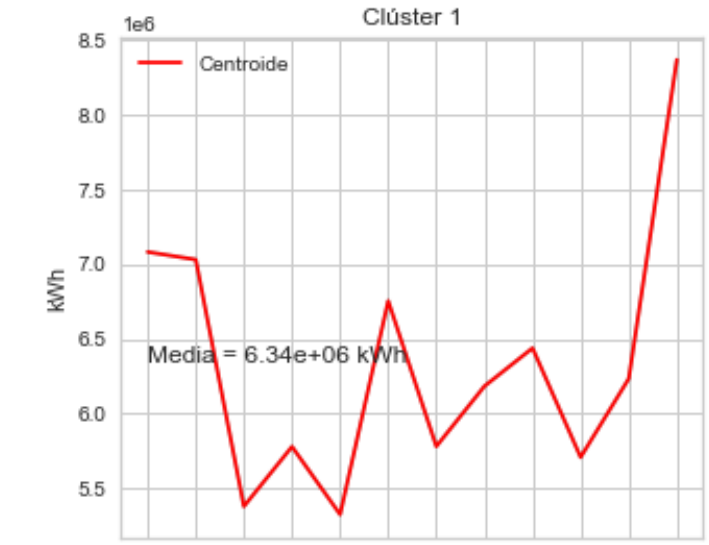


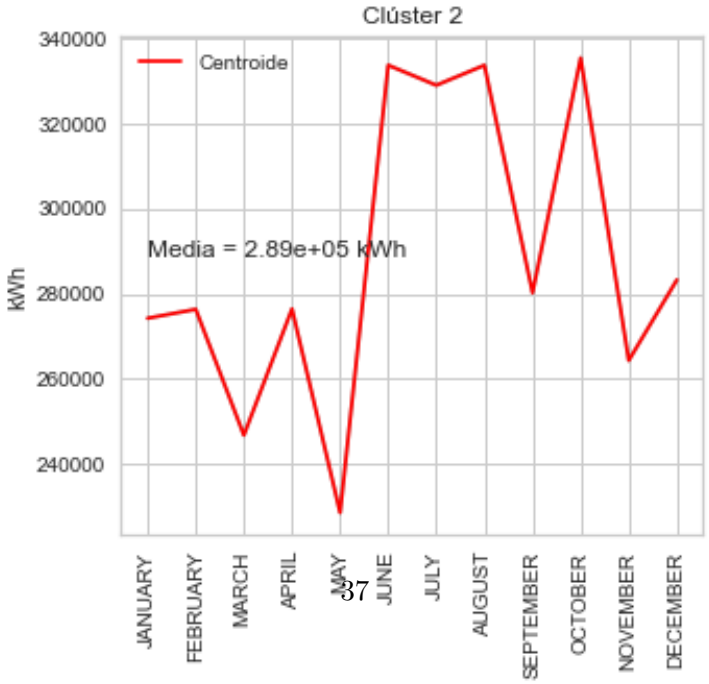

```
<Figure size 576x396 with 0 Axes>
```

A continuación se muestran las gráficas de tendencias para cada clúster.

[39]:
```
labels = pd.DataFrame(km_labels, columns=["label"])

k = km_centroids.shape[0]

f, axes = plt.subplots(k, 1, figsize=(5, 15), sharex=True)

for ki in range(k):
    cluster = energy_industrial_by_community.iloc[labels.query(f'label ==␣
 ↪{ki}').index][dl.energy_cols]
    palette={i:"darkcyan" for i in range(cluster.shape[0])}

    ax0 = sns.lineplot(data=cluster.values.T, ax=axes[ki], palette=palette,␣
 ↪alpha=0.3)
    _ = [line.set_linestyle("-") for line in ax0.lines]

    ax0c = sns.lineplot(data=km_centroids[ki, :].T, ax=axes[ki], color="red")
    _ = [line.set_linestyle("-") for line in ax0c.lines]

    ax0c.legend(ax0c.lines[-1:], ["Centroide"]);

    ax0.set_title(f"Clúster {ki}")
    Media = km_centroids[ki, :].mean()
    ax0.text(0, Media, f"{Media = :0.2e} kWh")
    ax0.set(ylabel="kWh")

    ax0.set_xticks([e for e in range(0, 12)])
    ax0.set_xticklabels([f"{datetime.date(2008, i, 1).strftime('%B').upper()}"␣
 ↪for i in range(1, 13)], rotation=90)

path = images_path / Path("ensayos/clustering_energia_tipo_zona_agrupa_ciudades/
 ↪industrial")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("cluster_grafica.svg"))
```

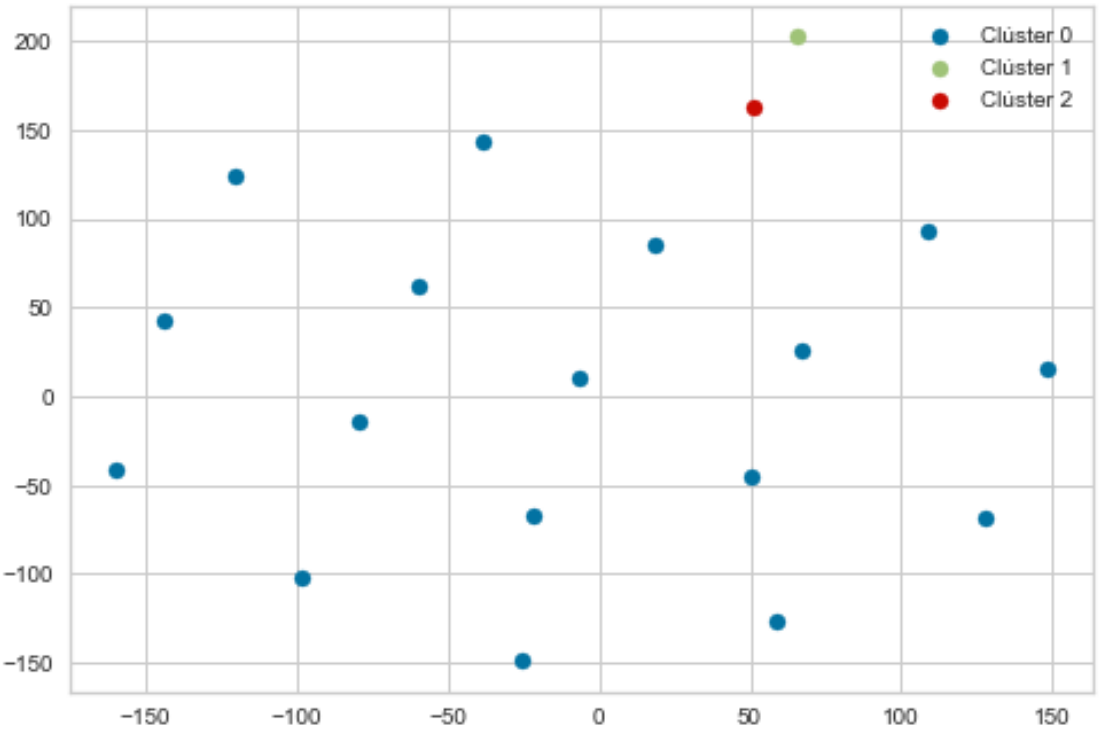

Clúster 0
Centroide
175000
150000
125000
100000
75000
50000
25000
0
kWh
Media = 1.59e+04 kWh


Clúster 1
1e6
Centroide
8.5
8.0
7.5
7.0
6.5
6.0
5.5
kWh
Media = 6.34e+06 kWh


Clúster 2
Centroide
340000
320000
300000
280000
260000
240000
kWh
Media = 2.89e+05 kWh
JANUARY
FEBRUARY
MARCH
APRIL
MAY
JUNE
JULY
AUGUST
SEPTEMBER
OCTOBER
NOVEMBER
DECEMBER

Se grafican las agrupaciones en clases para cada muestra.

[40]:
```
v = tsne(energy_industrial_by_community[dl.energy_cols], 2)
for label in range(k):
    plt.scatter(
            x=v[(labels['label']==label).values, 0],
            y=v[(labels['label']==label).values, 1]
    )
plt.legend([f"Clúster {i}" for i in range(k)])

path = images_path / Path("ensayos/clustering_energia_tipo_zona_agrupa_ciudades/
 ↪industrial")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("cluster_scatter.svg"))
```



¿Qué comunidad hay en cada clúster?

[ ]:
```
comunities_by_cluster = {}

for nlab in range(k):
    com_index = np.where(labels==nlab)[0]
```

```
    com_list = []
    for c in energy_industrial_by_community.values[com_index][:,0]:
        com_list.append(c)
    comunities_by_cluster[nlab] = com_list

for k in sorted(comunities_by_cluster, key=lambda k:␣
 ↪len(comunities_by_cluster[k])):
    print(f"Al clúster {k} pertenecen las comunidades:")
    for e in comunities_by_cluster[k]:
        print("\t", e)
```

1.3.2 Volver al inicio

1.3.3 Clustering de gas por tipo de zonas agrupando por ciudades

Se va a calcular el valor medio de consumo de gas por cada una de las comunidades y buscar las posibles agrupaciones de consumo para cada tipo de zona en la ciudad (comercial, industrial y residencial).

A continuación se calcula la media de consumo de gas para cada zona.

[42]:
```
gas_mean_by_community_type = dl[['COMMUNITY AREA NAME', 'BUILDING TYPE'] + dl.
 ↪gas_cols].groupby(['COMMUNITY AREA NAME', 'BUILDING TYPE'], as_index=False).
 ↪mean()
gas_mean_by_community_type
```

[42]:

	COMMUNITY AREA NAME	BUILDING TYPE	THERM JANUARY 2010	\
0	Albany Park	Commercial	2452.952909	
1	Albany Park	Industrial	2146.000000	
2	Albany Park	Residential	3241.642670	
3	Archer Heights	Commercial	8188.828571	
4	Archer Heights	Industrial	955.000000	
..	…	…	…	
169	West Town	Commercial	1986.941355	
170	West Town	Industrial	1579.000000	
171	West Town	Residential	2513.479660	
172	Woodlawn	Commercial	2834.130682	
173	Woodlawn	Residential	2977.754569	

	THERM FEBRUARY 2010	THERM MARCH 2010	THERM APRIL 2010	THERM MAY 2010	\
0	2075.800554	1740.199446	959.753463	651.722992	
1	1963.000000	1967.000000	1038.000000	681.000000	
2	2769.778796	2315.806283	1258.925393	809.119110	
3	6049.671429	4799.785714	2470.214286	1610.285714	
4	815.000000	713.000000	586.000000	442.000000	
..	…	…	…	…	

```
169          1696.244692       1459.205258        727.702730      455.524772
170          1517.000000        944.000000        374.000000      299.000000
171          2132.196114       1986.321190        969.794171      573.477231
172          2401.289773       2113.295455       1213.556818      792.159091
173          2526.342037       2466.485640       1323.885117      802.026110

     THERM JUNE 2010  THERM JULY 2010  THERM AUGUST 2010  \
0         428.936288       383.991690         338.675900
1         475.000000       336.000000         344.000000
2         453.017016       340.260471         299.039267
3        1260.871429      1232.642857        1075.342857
4         367.000000       363.000000         328.000000
..               …                …                  …
169       327.140546       277.324570         253.211325
170       227.000000       224.000000         211.000000
171       359.237401       251.375228         227.674560
172       471.852273       310.397727         258.914773
173       462.357702       273.563969         218.070496

     THERM SEPTEMBER 2010  THERM OCTOBER 2010  THERM NOVEMBER 2010  \
0               371.581717          427.340720           735.626039
1               486.000000          549.000000           621.000000
2               310.349476          378.269634           804.899215
3              1031.357143         1411.228571          2701.742857
4               330.000000          330.000000           402.000000
..                     …                   …                    …
169             268.819009          318.184024           540.920121
170             219.000000          254.000000           491.000000
171             234.496053          285.306618           536.063752
172             335.318182          500.318182           994.585227
173             250.334204          345.509138           767.381201

     THERM DECEMBER 2010
0            1650.426593
1            1227.000000
2            2178.858639
3            6576.000000
4             651.000000
..                   …
169          1413.849343
170          1720.000000
171          1626.820886
172          1930.295455
173          1826.772846

[174 rows x 14 columns]
```

Comercial Se seleccionan las muestras de consumo medio para las zonas comerciales de cada comunidad.

[43]:
```
gas_commercial_by_community = gas_mean_by_community_type[gas_mean_by_community_type['BUILDING TYPE']=="Commercial"]
gas_commercial_by_community
```

[43]:

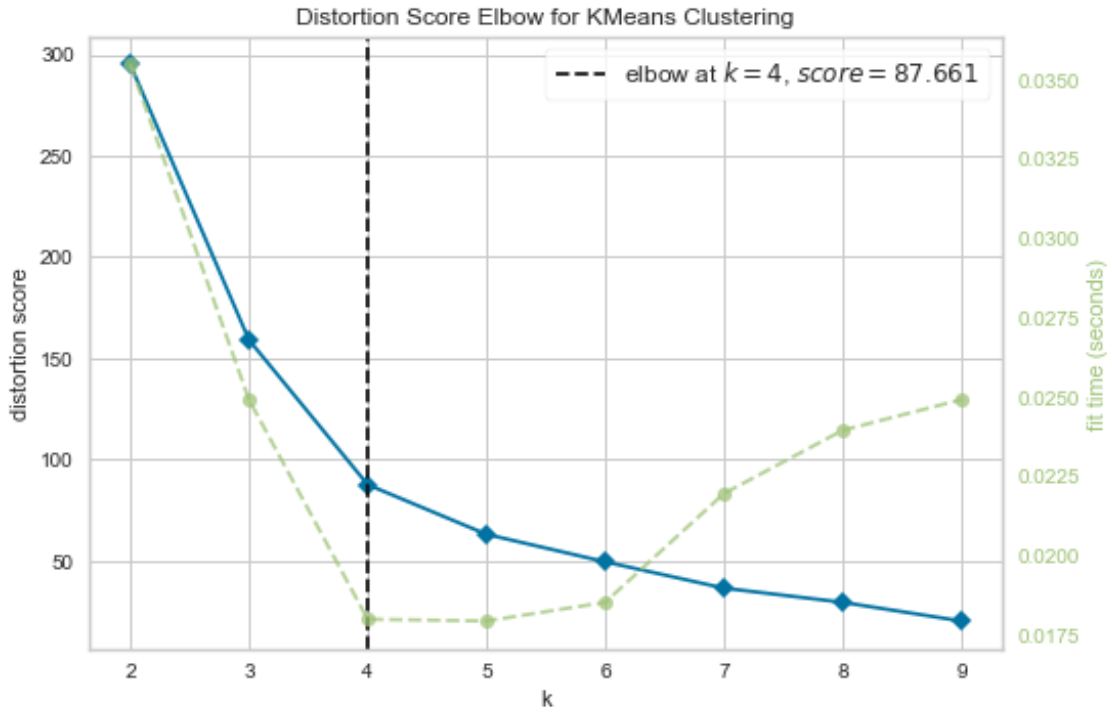
	COMMUNITY AREA NAME	BUILDING TYPE	THERM JANUARY 2010	\
0	Albany Park	Commercial	2452.952909	
3	Archer Heights	Commercial	8188.828571	
6	Armour Square	Commercial	6478.955556	
8	Ashburn	Commercial	3481.688889	
11	Auburn Gresham	Commercial	2171.017021	
..	…	…	…	
163	West Lawn	Commercial	7348.329545	
165	West Pullman	Commercial	1978.388889	
167	West Ridge	Commercial	3263.905782	
169	West Town	Commercial	1986.941355	
172	Woodlawn	Commercial	2834.130682	

	THERM FEBRUARY 2010	THERM MARCH 2010	THERM APRIL 2010	THERM MAY 2010	\
0	2075.800554	1740.199446	959.753463	651.722992	
3	6049.671429	4799.785714	2470.214286	1610.285714	
6	4191.155556	6708.666667	3393.488889	2674.088889	
8	2982.266667	2690.400000	1358.300000	643.333333	
11	2023.059574	1663.072340	904.761702	575.544681	
..	…	…	…	…	
163	6110.534091	4972.920455	3105.943182	2553.590909	
165	1748.277778	1328.944444	689.555556	396.388889	
167	2743.177730	2170.552463	1297.751606	960.359743	
169	1696.244692	1459.205258	727.702730	455.524772	
172	2401.289773	2113.295455	1213.556818	792.159091	

	THERM JUNE 2010	THERM JULY 2010	THERM AUGUST 2010	\
0	428.936288	383.991690	338.675900	
3	1260.871429	1232.642857	1075.342857	
6	2236.422222	1926.355556	2014.577778	
8	489.133333	410.922222	394.155556	
11	387.736170	298.910638	287.625532	
..	…	…	…	
163	2220.943182	2006.045455	2479.113636	
165	225.263889	187.805556	188.222222	
167	686.691649	679.357602	594.314775	
169	327.140546	277.324570	253.211325	
172	471.852273	310.397727	258.914773	

```
     THERM SEPTEMBER 2010  THERM OCTOBER 2010  THERM NOVEMBER 2010  \
0              371.581717          427.340720           735.626039
3             1031.357143         1411.228571          2701.742857
6             1994.600000         2614.133333          4044.177778
8              380.022222          480.088889           793.022222
11             315.757447          427.374468           815.834043
..                    …                   …                   …
163           2314.534091         2225.090909          2769.556818
165            210.152778          358.972222           671.430556
167            552.002141          662.603854          1267.509636
169            268.819009          318.184024           540.920121
172            335.318182          500.318182           994.585227

     THERM DECEMBER 2010
0             1650.426593
3             6576.000000
6             6843.533333
8             1809.377778
11            1793.906383
..                    …
163           5755.295455
165           1529.125000
167           2628.933619
169           1413.849343
172           1930.295455

[77 rows x 14 columns]
```

[44]:
```
kmax = 10
k_values = (2, kmax)

scaler = preprocessing.StandardScaler()
data = gas_commercial_by_community[dl.gas_cols]
norm_data = scaler.fit_transform(data)

km = KMeansCluster(norm_data)
km.cluster(k_values)

path = images_path / Path("ensayos/clustering_gas_tipo_zona_agrupa_ciudades/
 ↪comercial")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("elbow.svg"))
```



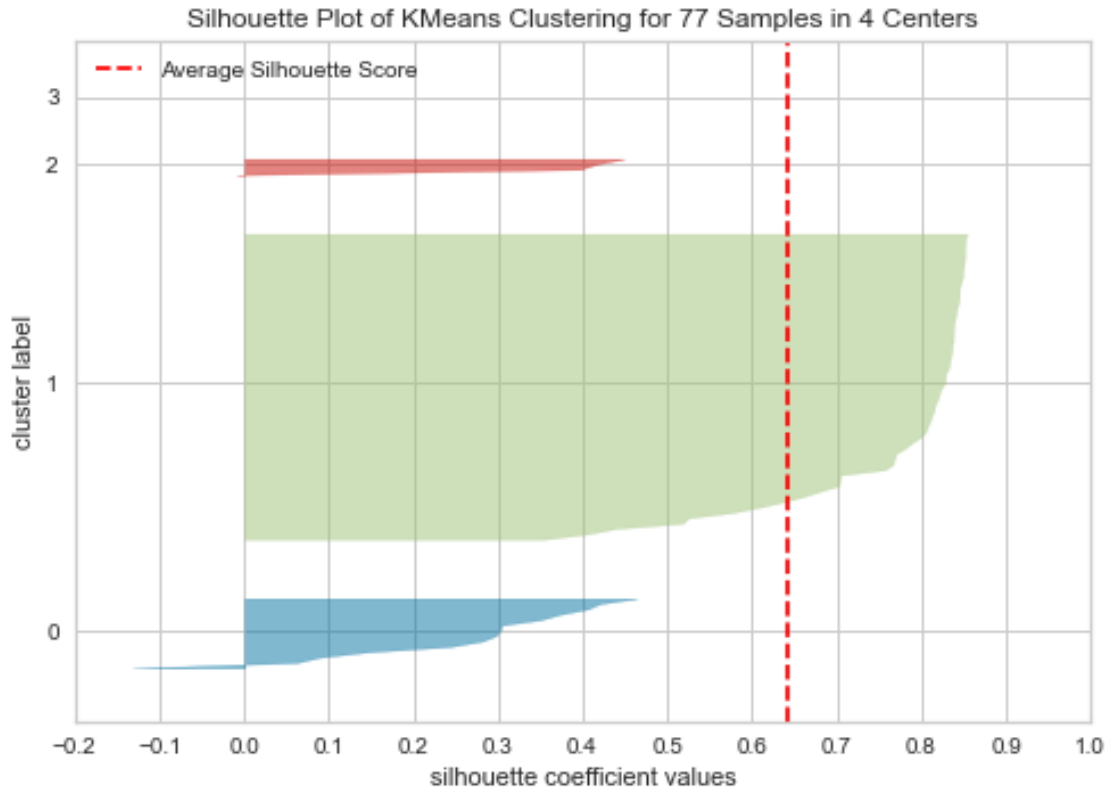

```
<Figure size 576x396 with 0 Axes>
```

Con K=4 seleccionamos la mejor división.

[45]:
```
km_best = km.get_kmeans_of(4)
km_labels = km_best.labels_
km_centroids = scaler.inverse_transform(km_best.cluster_centers_)

km_labels
```

[45]:
```
array([1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 0, 1, 1, 1, 0,
       1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 3, 1, 1,
       1, 1, 1, 2, 0, 0, 0, 1, 1, 1, 1, 0, 1, 0, 2, 2, 1, 1, 1, 0, 1, 1,
       1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1])
```

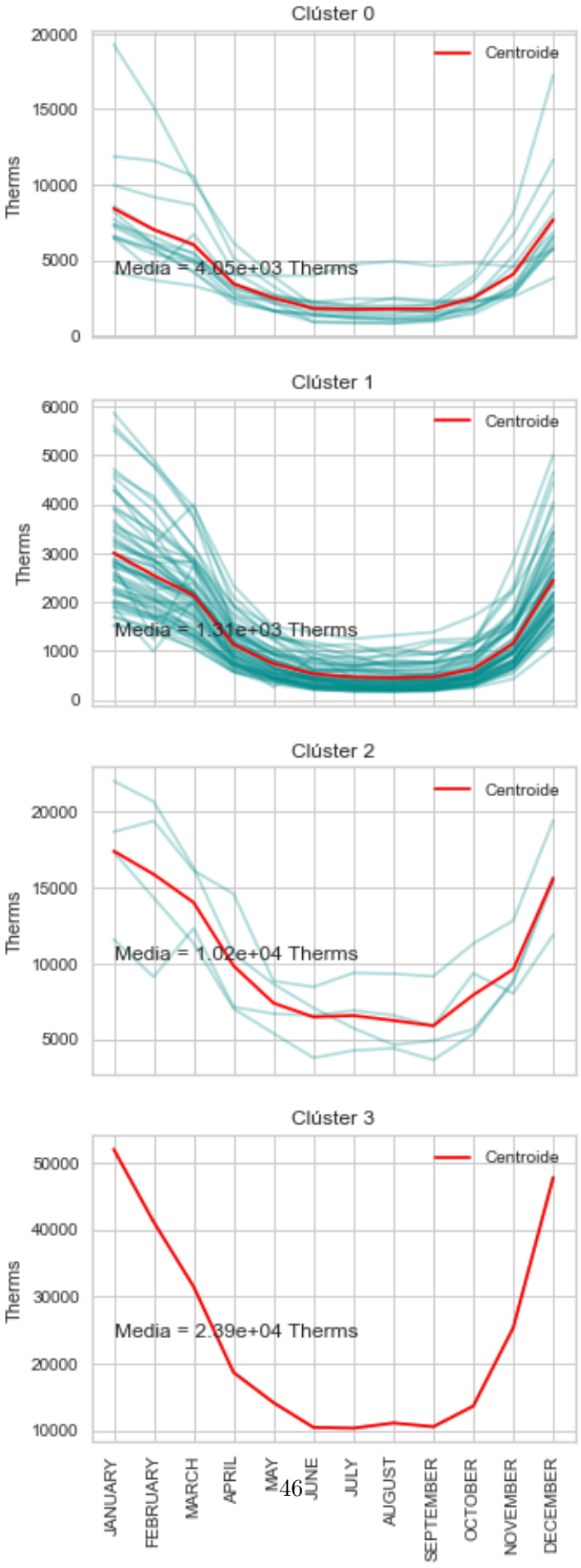
[46]:
```
_ = silhouette_visualizer(km_best, norm_data, colors='yellowbrick')

path = images_path / Path("ensayos/clustering_gas_tipo_zona_agrupa_ciudades/
 ↪comercial")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("silhouette.svg"))
```



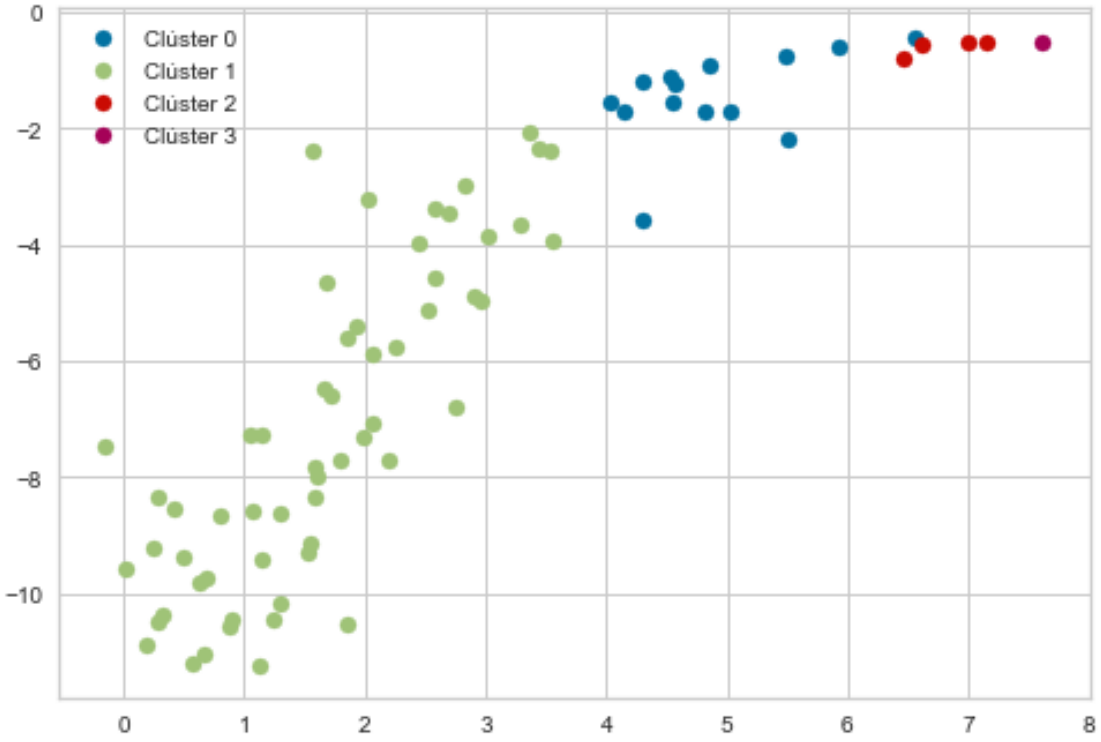

```
<Figure size 576x396 with 0 Axes>
```

[47]:
```
labels = pd.DataFrame(km_labels, columns=["label"])

k = km_centroids.shape[0]

f, axes = plt.subplots(k, 1, figsize=(5, 15), sharex=True)

for ki in range(k):
    cluster = gas_commercial_by_community.iloc[labels.query(f'label == {ki}').
↪index][dl.gas_cols]
    palette={i:"darkcyan" for i in range(cluster.shape[0])}

    ax0 = sns.lineplot(data=cluster.values.T, ax=axes[ki], palette=palette,␣
↪alpha=0.3)
    _ = [line.set_linestyle("-") for line in ax0.lines]

    ax0c = sns.lineplot(data=km_centroids[ki, :].T, ax=axes[ki], color="red")
    _ = [line.set_linestyle("-") for line in ax0c.lines]

    ax0c.legend(ax0c.lines[-1:], ["Centroide"]);
```

```
    ax0.set_title(f"Clúster {ki}")
    Media = km_centroids[ki, :].mean()
    ax0.text(0, Media, f"{Media = :0.2e} Therms")
    ax0.set(ylabel="Therms")

    ax0.set_xticks([e for e in range(0, 12)])
    ax0.set_xticklabels([f"{datetime.date(2008, i, 1).strftime('%B').upper()}"␣
 ↪for i in range(1, 13)], rotation=90)

path = images_path / Path("ensayos/clustering_gas_tipo_zona_agrupa_ciudades/
 ↪comercial")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("cluster_grafica.svg"))
```


Clúster 0
Centroide
Therms
Media = 4.05e+03 Therms
Clúster 1
Centroide
Therms
Media = 1.31e+03 Therms
Clúster 2
Centroide
Therms
Media = 1.02e+04 Therms
Clúster 3
Centroide
Therms
Media = 2.39e+04 Therms
JANUARY
FEBRUARY
MARCH
APRIL
MAY
JUNE
JULY
AUGUST
SEPTEMBER
OCTOBER
NOVEMBER
DECEMBER

[48]:
```
v = tsne(gas_commercial_by_community[dl.gas_cols], 2)
for label in range(k):
    plt.scatter(
            x=v[(labels['label']==label).values, 0],
            y=v[(labels['label']==label).values, 1]
    )
plt.legend([f"Clúster {i}" for i in range(k)])

path = images_path / Path("ensayos/clustering_gas_tipo_zona_agrupa_ciudades/
 ↪comercial")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("cluster_scatter.svg"))
```



¿Qué comunidades hay en cada clúster?

[ ]:
```
comunities_by_cluster = {}

for nlab in range(k):
    com_index = np.where(labels==nlab)[0]
    com_list = []
```

```
    for c in gas_commercial_by_community.values[com_index][:,0]:
        com_list.append(c)
    comunities_by_cluster[nlab] = com_list

for k in sorted(comunities_by_cluster, key=lambda k:␣
↪len(comunities_by_cluster[k])):
    print(f"Al clúster {k} pertenecen las comunidades:")
    for e in comunities_by_cluster[k]:
        print("\t", e)
```

Residencial Se seleccionan las muestras de consumo medio para las zonas residenciales de cada comunidad.

[50]:
```
gas_residential_by_community =␣
↪gas_mean_by_community_type[gas_mean_by_community_type['BUILDING␣
↪TYPE']=="Residential"]
gas_residential_by_community
```

[50]:

	COMMUNITY AREA NAME	BUILDING TYPE	THERM JANUARY 2010	\
2	Albany Park	Residential	3241.642670	
5	Archer Heights	Residential	2545.916399	
7	Armour Square	Residential	3862.246377	
10	Ashburn	Residential	3909.274536	
12	Auburn Gresham	Residential	3406.123684	
..	…	…	…	
164	West Lawn	Residential	3252.747871	
166	West Pullman	Residential	2077.643002	
168	West Ridge	Residential	3534.174115	
171	West Town	Residential	2513.479660	
173	Woodlawn	Residential	2977.754569	

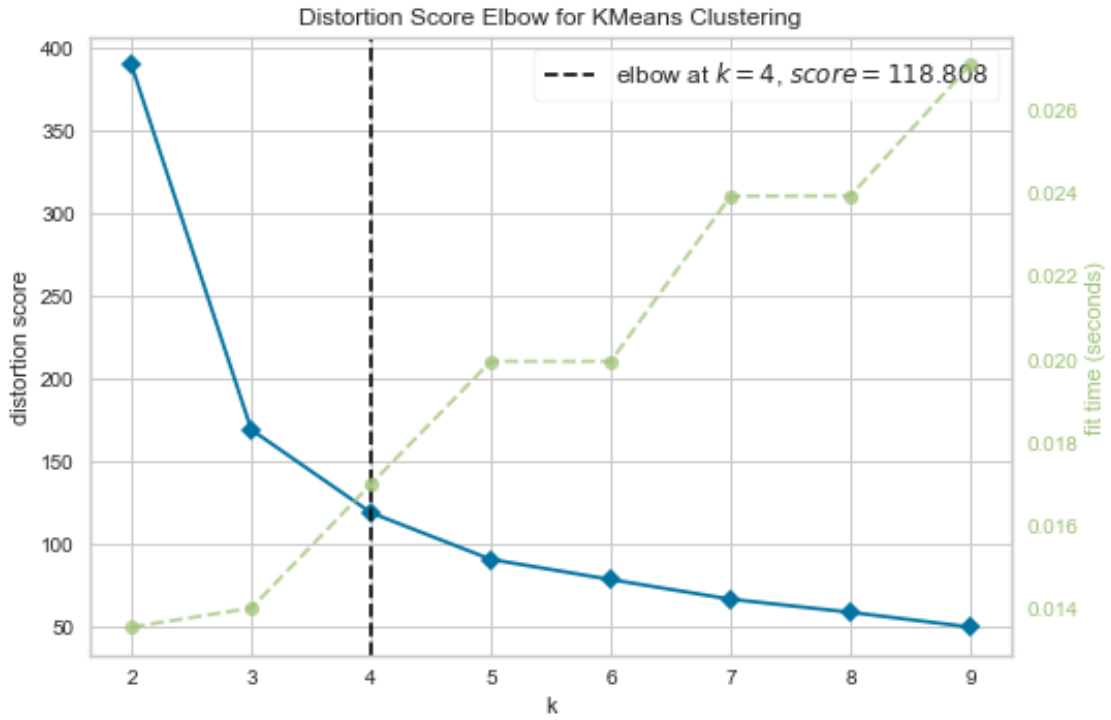
	THERM FEBRUARY 2010	THERM MARCH 2010	THERM APRIL 2010	THERM MAY 2010	\
2	2769.778796	2315.806283	1258.925393	809.119110	
5	2156.491961	1613.273312	876.234727	541.424437	
7	1321.884058	4437.101449	1282.884058	804.057971	
10	3348.224138	3204.883289	1777.124668	923.917772	
12	3148.660526	2421.146491	1318.972807	796.865789	
..	…	…	…	…	
164	2892.359455	2004.451448	1125.492334	670.580920	
166	1966.321501	1413.970588	762.053753	506.345842	
168	2885.684358	2397.912477	1426.387337	880.140596	
171	2132.196114	1986.321190	969.794171	573.477231	
173	2526.342037	2466.485640	1323.885117	802.026110	

	THERM JUNE 2010	THERM JULY 2010	THERM AUGUST 2010	\
2	453.017016	340.260471	299.039267	
5	303.745981	284.848875	242.289389	

```
7         638.521739       524.304348         535.840580
10        646.332891       368.294430         386.696286
12        401.747368       266.898246         257.805263
..               …                …                  …
164       376.269165       348.132879         319.649063
166       222.771805       179.826572         178.399594
168       459.661080       356.593110         307.603352
171       359.237401       251.375228         227.674560
173       462.357702       273.563969         218.070496

     THERM SEPTEMBER 2010  THERM OCTOBER 2010  THERM NOVEMBER 2010  \
2               310.349476          378.269634           804.899215
5               257.781350          322.742765           743.231511
7               510.826087          707.594203          1762.594203
10              352.381963          445.805040           833.238727
12              264.690351          471.408772          1128.202632
..                     …                   …                  …
164             307.131175          428.001704           964.115843
166             168.265720          278.640974           696.110548
168             327.020484          424.310056           933.571695
171             234.496053          285.306618           536.063752
173             250.334204          345.509138           767.381201

     THERM DECEMBER 2010
2             2178.858639
5             1967.601286
7             3795.260870
10            2128.458886
12            2890.464912
..                   …
164           2642.299830
166           1775.339757
168           2384.201117
171           1626.820886
173           1826.772846

[77 rows x 14 columns]
```

[51]:
```
kmax = 10
k_values = (2, kmax)

scaler = preprocessing.StandardScaler()
data = gas_residential_by_community[dl.gas_cols]
norm_data = scaler.fit_transform(data)

km = KMeansCluster(norm_data)
km.cluster(k_values)
```

```
path = images_path / Path("ensayos/clustering_gas_tipo_zona_agrupa_ciudades/
 ↪residencial")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("elbow.svg"))
```




```
<Figure size 576x396 with 0 Axes>
```

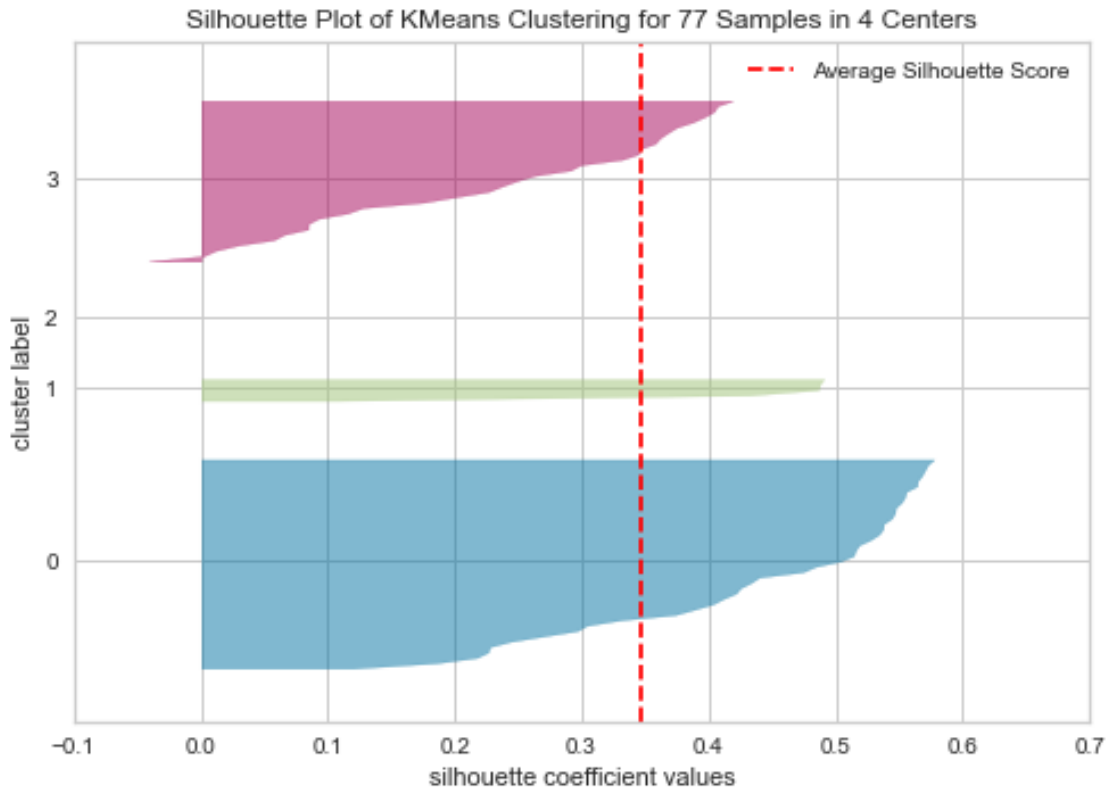
Con k=4 tenemos la división óptima.

[52]:
```
km_best = km.get_kmeans_of(4)
km_labels = km_best.labels_
km_centroids = scaler.inverse_transform(km_best.cluster_centers_)

km_labels
```

[52]:
```
array([3, 0, 3, 3, 3, 3, 3, 0, 0, 3, 0, 0, 0, 3, 3, 0, 0, 1, 3, 3, 0, 3,
       3, 0, 3, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 3, 3, 3, 0, 2, 0, 0,
       0, 3, 3, 3, 1, 0, 0, 0, 0, 3, 3, 1, 0, 3, 0, 3, 3, 0, 0, 0, 0, 3,
       3, 3, 3, 0, 0, 0, 3, 0, 3, 0, 0])
```

[53]:
```
_ = silhouette_visualizer(km_best, norm_data, colors='yellowbrick')
```

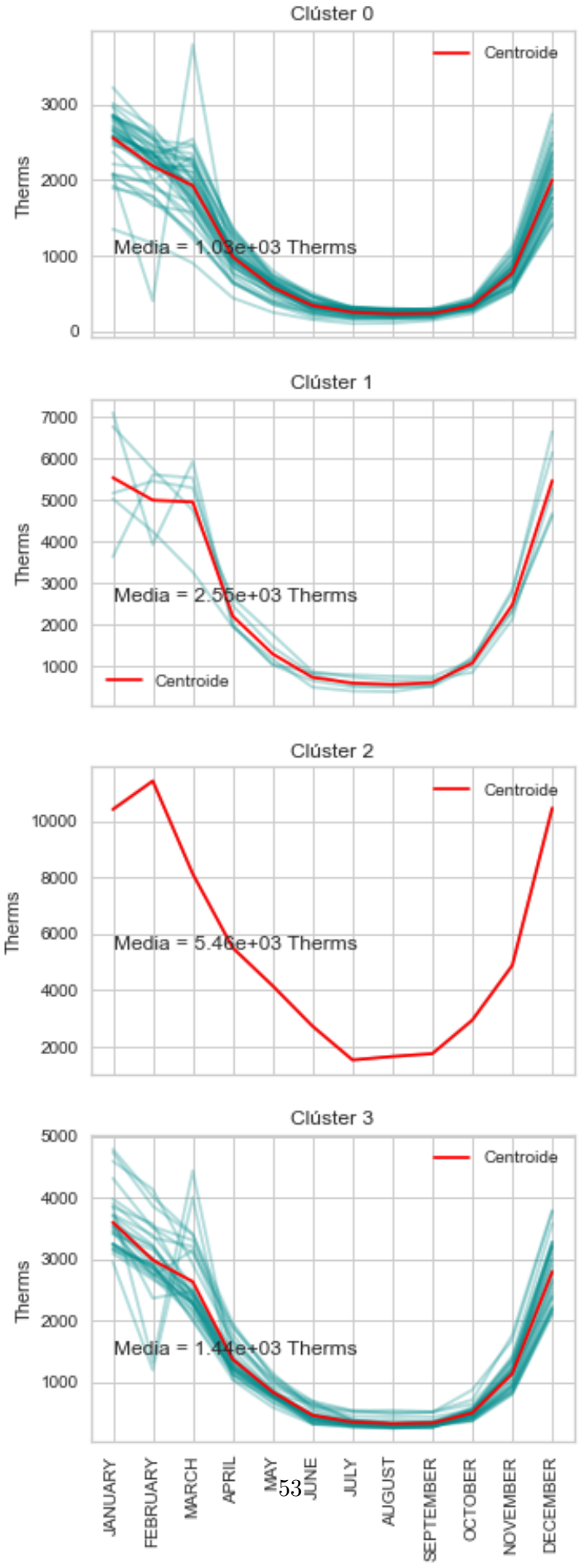
```
path = images_path / Path("ensayos/clustering_gas_tipo_zona_agrupa_ciudades/
 ↪residencial")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("silhouette.svg"))
```



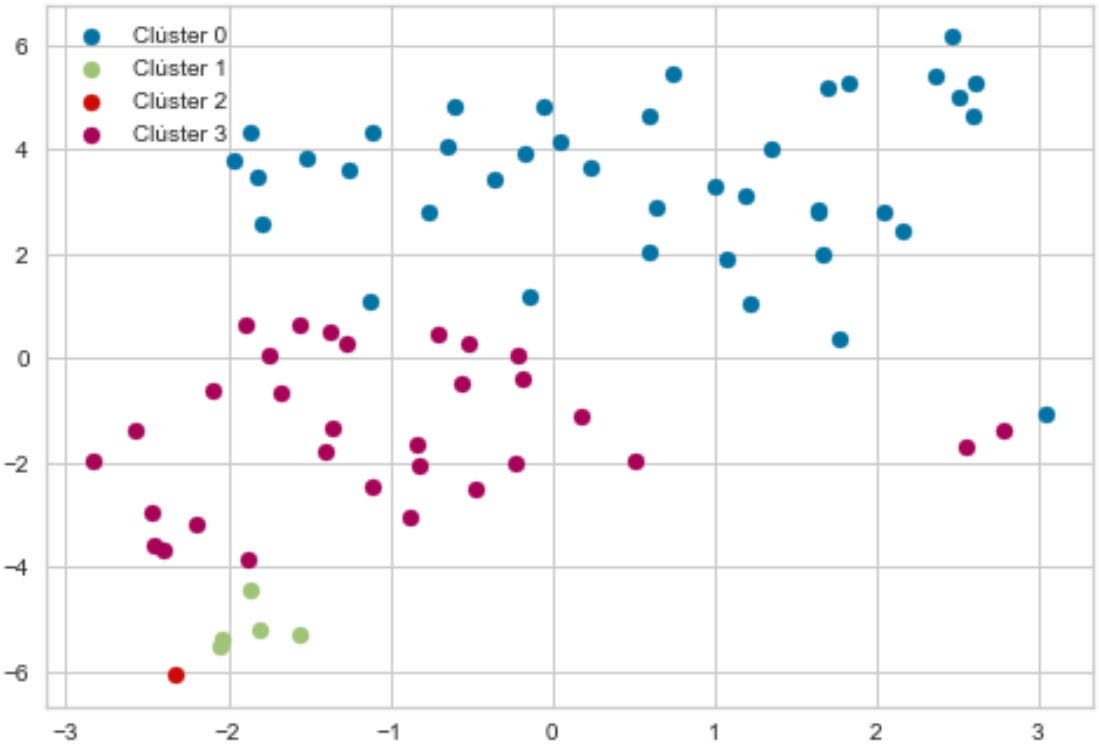

```
<Figure size 576x396 with 0 Axes>
```

[54]:
```
labels = pd.DataFrame(km_labels, columns=["label"])

k = km_centroids.shape[0]

f, axes = plt.subplots(k, 1, figsize=(5, 15), sharex=True)

for ki in range(k):
    cluster = gas_residential_by_community.iloc[labels.query(f'label == {ki}').
 ↪index][dl.gas_cols]
    palette={i:"darkcyan" for i in range(cluster.shape[0])}

    ax0 = sns.lineplot(data=cluster.values.T, ax=axes[ki], palette=palette,␣
 ↪alpha=0.3)
    _ = [line.set_linestyle("-") for line in ax0.lines]
```

```
    ax0c = sns.lineplot(data=km_centroids[ki, :].T, ax=axes[ki], color="red")
    _ = [line.set_linestyle("-") for line in ax0c.lines]

    ax0c.legend(ax0c.lines[-1:], ["Centroide"]);

    ax0.set_title(f"Clúster {ki}")
    Media = km_centroids[ki, :].mean()
    ax0.text(0, Media, f"{Media = :0.2e} Therms")
    ax0.set(ylabel="Therms")

    ax0.set_xticks([e for e in range(0, 12)])
    ax0.set_xticklabels([f"{datetime.date(2008, i, 1).strftime('%B').upper()}"␣
 ↪for i in range(1, 13)], rotation=90)

path = images_path / Path("ensayos/clustering_gas_tipo_zona_agrupa_ciudades/
 ↪residencial")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("cluster_grafica.svg"))
```


Clúster 0
Centroide
Therms
3000
2000
1000
0
Media = 1.03e+03 Therms
Clúster 1
7000
6000
5000
4000
3000
2000
1000
Therms
Media = 2.55e+03 Therms
Centroide
Clúster 2
Centroide
10000
8000
6000
4000
2000
Therms
Media = 5.46e+03 Therms
Clúster 3
Centroide
5000
4000
3000
2000
1000
Therms
Media = 1.44e+03 Therms
JANUARY
FEBRUARY
MARCH
APRIL
MAY
JUNE
JULY
AUGUST
SEPTEMBER
OCTOBER
NOVEMBER
DECEMBER

[55]:
```
v = tsne(gas_residential_by_community[dl.gas_cols], 2)
for label in range(k):
    plt.scatter(
            x=v[(labels['label']==label).values, 0],
            y=v[(labels['label']==label).values, 1]
    )
plt.legend([f"Clúster {i}" for i in range(k)])

path = images_path / Path("ensayos/clustering_gas_tipo_zona_agrupa_ciudades/
 ↪residencial")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("cluster_scatter.svg"))
```



[ ]:
```
comunities_by_cluster = {}

for nlab in range(k):
    com_index = np.where(labels==nlab)[0]
    com_list = []
    for c in gas_residential_by_community.values[com_index][:,0]:
        com_list.append(c)
```

```
    comunities_by_cluster[nlab] = com_list

for k in sorted(comunities_by_cluster, key=lambda k:␣
 ↪len(comunities_by_cluster[k])):
    print(f"Al clúster {k} pertenecen las comunidades:")
    for e in comunities_by_cluster[k]:
        print("\t", e)
```

Industrial Se seleccionan las muestras de consumo medio para las zonas industriales de cada comunidad.

[57]:
```
gas_industrial_by_community =␣
 ↪gas_mean_by_community_type[gas_mean_by_community_type['BUILDING␣
 ↪TYPE']=="Industrial"]
gas_industrial_by_community
```

[57]:

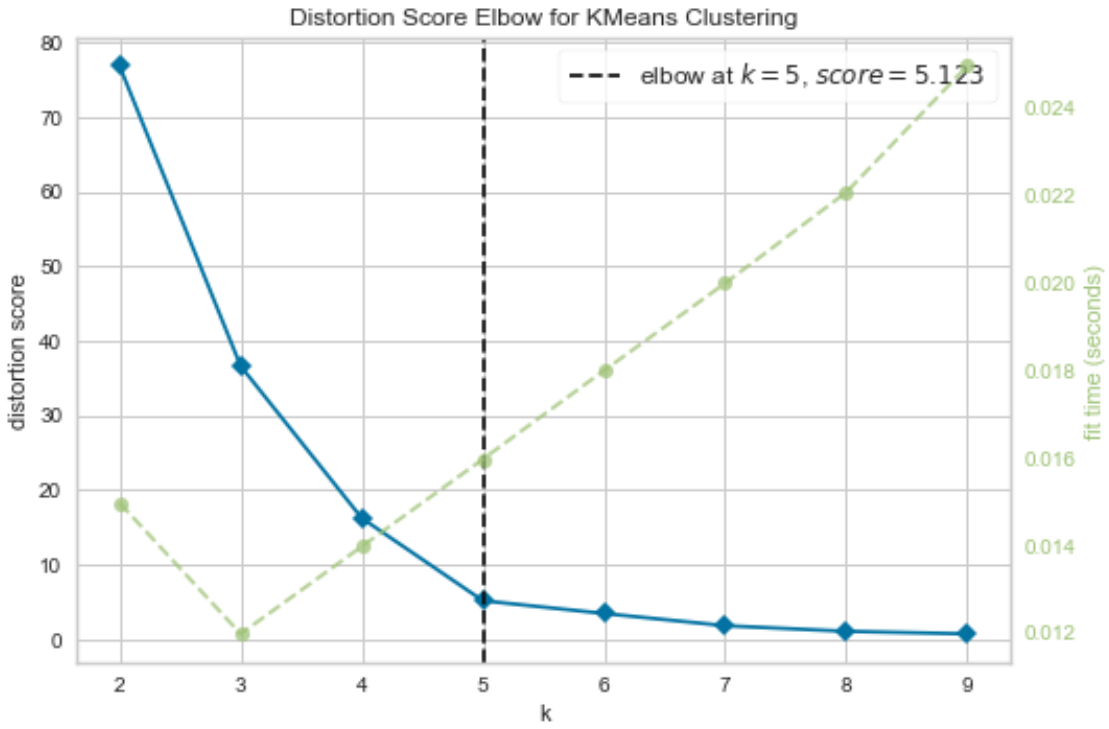
	COMMUNITY AREA NAME	BUILDING TYPE	THERM JANUARY 2010	\
1	Albany Park	Industrial	2146.0	
4	Archer Heights	Industrial	955.0	
9	Ashburn	Industrial	930.0	
14	Austin	Industrial	18043.5	
27	Brighton Park	Industrial	331.0	
46	East Side	Industrial	2587.0	
49	Edgewater	Industrial	797.0	
56	Forest Glen	Industrial	1126.0	
61	Gage Park	Industrial	24.0	
72	Hermosa	Industrial	2157.0	
79	Irving Park	Industrial	659.0	
82	Jefferson Park	Industrial	3328.0	
85	Kenwood	Industrial	1179.0	
88	Lakeview	Industrial	13290.0	
111	Near South Side	Industrial	13111.0	
114	Near West Side	Industrial	9963.0	
121	North Lawndale	Industrial	2505.0	
126	Norwood Park	Industrial	1120.0	
147	South Lawndale	Industrial	8450.0	
170	West Town	Industrial	1579.0	

	THERM FEBRUARY 2010	THERM MARCH 2010	THERM APRIL 2010	THERM MAY 2010	\
1	1963.000000	1967.000000	1038.000000	681.000000	
4	815.000000	713.000000	586.000000	442.000000	
9	940.000000	978.000000	368.000000	246.000000	
14	17736.000000	15444.500000	6863.000000	5171.000000	
27	66.000000	205.000000	93.000000	65.000000	
46	2152.000000	1263.000000	404.000000	242.000000	
49	749.000000	511.500000	232.000000	148.500000	
56	921.000000	592.000000	362.000000	228.000000	

```
61            30.000000        25.000000        15.000000       24.000000
72          1434.000000      1022.000000       499.000000      152.000000
79           579.500000       367.500000       143.500000       62.000000
82          3034.000000      2073.000000      1140.500000      317.500000
85          2306.000000      1878.000000       957.000000      768.000000
88         11153.500000     10964.500000      8081.500000     7949.500000
111        10778.000000      8246.000000      3034.000000    12429.000000
114         8585.857143      5366.142857      2600.714286     1314.571429
121         3313.000000      2372.000000       787.000000      150.000000
126          888.000000       797.000000       406.000000      191.000000
147         7489.000000      7957.000000      4860.000000     1718.000000
170         1517.000000       944.000000       374.000000      299.000000

     THERM JUNE 2010  THERM JULY 2010  THERM AUGUST 2010  \
1         475.000000            336.0         344.000000
4         367.000000            363.0         328.000000
9         104.000000             11.0          18.000000
14       2614.500000           3016.0        2598.000000
27         43.000000             28.0          33.000000
46          7.000000              6.0           6.000000
49         93.500000             78.0          84.000000
56        160.000000            243.0         192.000000
61         15.000000             19.0          12.000000
72         55.000000             46.0          28.000000
79         30.000000             19.0          16.000000
82         75.000000             39.0          38.500000
85        776.000000            660.0         729.000000
88       7089.500000           8063.0        7671.000000
111       249.000000            288.0         252.000000
114      1281.857143            218.0         269.714286
121        38.000000             86.0          38.000000
126        28.000000              2.0           1.000000
147        17.000000             17.0         637.000000
170       227.000000            224.0         211.000000

     THERM SEPTEMBER 2010  THERM OCTOBER 2010  THERM NOVEMBER 2010  \
1              486.000000          549.000000           621.000000
4              330.000000          330.000000           402.000000
9               11.000000           18.000000           126.000000
14            3047.000000         4616.000000          7567.500000
27              19.000000           38.000000            86.000000
46               6.000000           40.000000           260.000000
49             102.000000          117.500000           272.000000
56             179.000000          230.000000           396.000000
61              13.000000           14.000000            12.000000
72              25.000000          102.000000           248.000000
79              15.500000           29.500000            86.500000
```

```
82              37.000000          124.500000          775.500000
85             811.000000          767.000000          813.000000
88            7952.500000         8158.500000         8921.000000
111            276.000000         3281.000000         4919.000000
114            289.428571         1161.428571         3826.428571
121              8.000000          156.000000          100.000000
126              5.000000           46.000000          269.000000
147             18.000000          195.000000         2767.000000
170            219.000000          254.000000          491.000000

     THERM DECEMBER 2010
1            1227.000000
4             651.000000
9             380.000000
14          14517.000000
27            191.000000
46           1759.000000
49            844.500000
56           1046.000000
61             18.000000
72           1544.000000
79            492.500000
82           2094.000000
85           2251.000000
88          13633.000000
111         10904.000000
114          7932.285714
121          1786.000000
126           690.000000
147          4067.000000
170          1720.000000
```

[58]:
```
kmax = 10
k_values = (2, kmax)

scaler = preprocessing.StandardScaler()
data = gas_industrial_by_community[dl.gas_cols]
norm_data = scaler.fit_transform(data)

km = KMeansCluster(norm_data)
km.cluster(k_values)

path = images_path / Path("ensayos/clustering_gas_tipo_zona_agrupa_ciudades/
 ↪industrial")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("elbow.svg"))
```



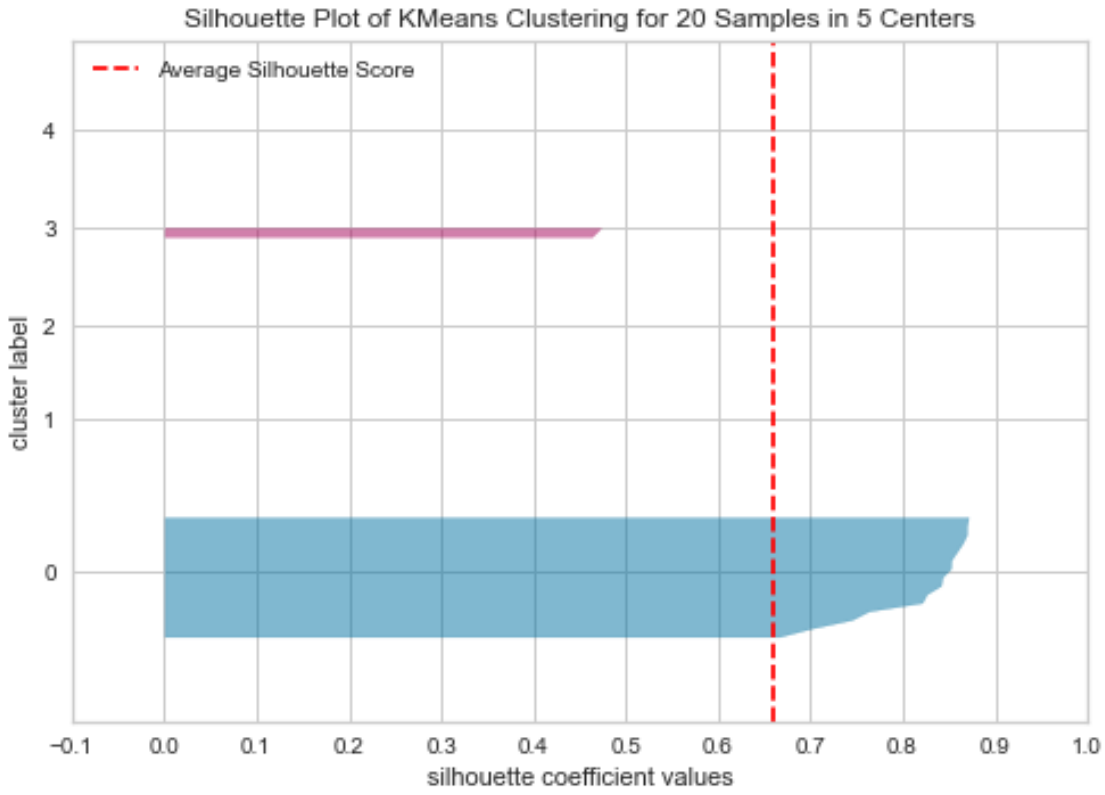

```
<Figure size 576x396 with 0 Axes>
```

En k=5 tenemos el punto codo, por lo que seleccionamos ese número de clústeres.

[59]:
```
km_best = km.get_kmeans_of(5)
km_labels = km_best.labels_
km_centroids = scaler.inverse_transform(km_best.cluster_centers_)

km_labels
```

[59]:
```
array([0, 0, 0, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 4, 3, 0, 0, 3, 0])
```

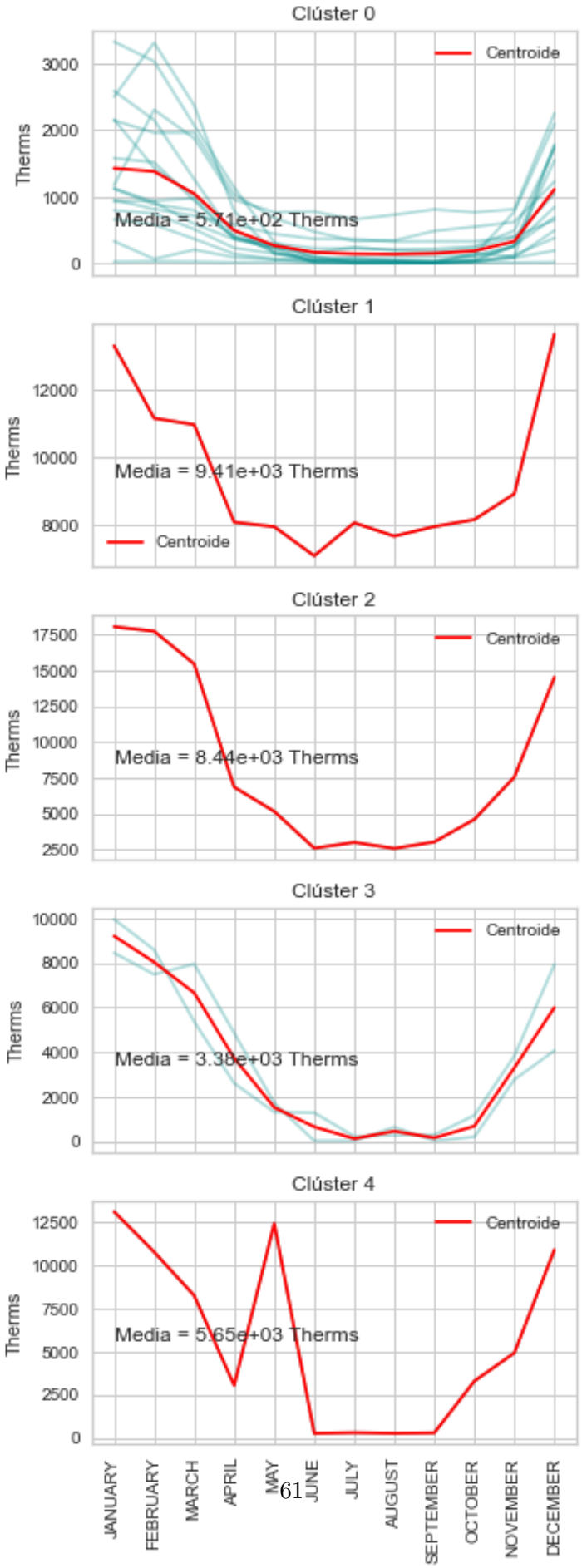
[60]:
```
_ = silhouette_visualizer(km_best, norm_data, colors='yellowbrick')

path = images_path / Path("ensayos/clustering_gas_tipo_zona_agrupa_ciudades/
↪industrial")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("silhouette.svg"))
```



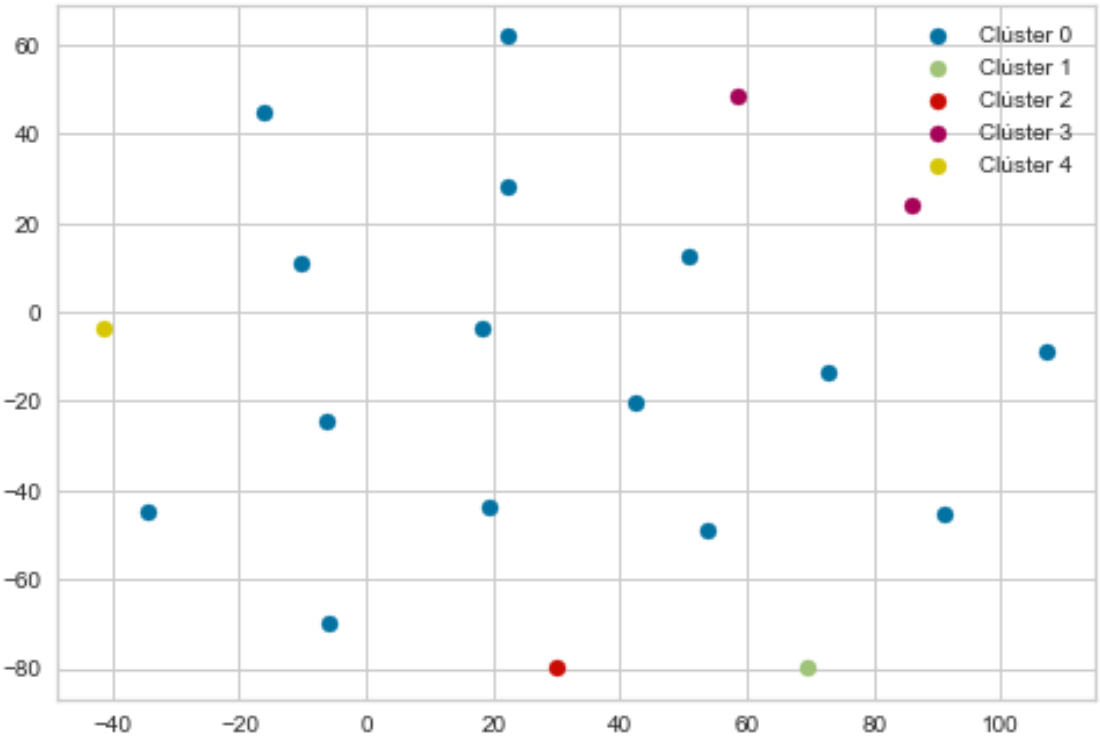
<Figure size 576x396 with 0 Axes>

[61]:
```
labels = pd.DataFrame(km_labels, columns=["label"])

k = km_centroids.shape[0]

f, axes = plt.subplots(k, 1, figsize=(5, 15), sharex=True)

for ki in range(k):
    cluster = gas_industrial_by_community.iloc[labels.query(f'label == {ki}').
↪index][dl.gas_cols]
    palette={i:"darkcyan" for i in range(cluster.shape[0])}

    ax0 = sns.lineplot(data=cluster.values.T, ax=axes[ki], palette=palette,␣
↪alpha=0.3)
    _ = [line.set_linestyle("-") for line in ax0.lines]

    ax0c = sns.lineplot(data=km_centroids[ki, :].T, ax=axes[ki], color="red")
    _ = [line.set_linestyle("-") for line in ax0c.lines]

    ax0c.legend(ax0c.lines[-1:], ["Centroide"]);
```

```
    ax0.set_title(f"Clúster {ki}")
    Media = km_centroids[ki, :].mean()
    ax0.text(0, Media, f"{Media = :0.2e} Therms")
    ax0.set(ylabel="Therms")

    ax0.set_xticks([e for e in range(0, 12)])
    ax0.set_xticklabels([f"{datetime.date(2008, i, 1).strftime('%B').upper()}"␣
 ↪for i in range(1, 13)], rotation=90)

path = images_path / Path("ensayos/clustering_gas_tipo_zona_agrupa_ciudades/
 ↪industrial")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("cluster_grafica.svg"))
```


Clúster 0
Centroide
Therms
3000
2000
1000
0
Media = 5.71e+02 Therms
Clúster 1
Therms
12000
10000
8000
Media = 9.41e+03 Therms
Centroide
Clúster 2
Centroide
Therms
17500
15000
12500
10000
7500
5000
2500
Media = 8.44e+03 Therms
Clúster 3
Centroide
Therms
10000
8000
6000
4000
2000
0
Media = 3.38e+03 Therms
Clúster 4
Centroide
Therms
12500
10000
7500
5000
2500
0
Media = 5.65e+03 Therms
JANUARY
FEBRUARY
MARCH
APRIL
MAY
JUNE
JULY
AUGUST
SEPTEMBER
OCTOBER
NOVEMBER
DECEMBER

[62]:
```
v = tsne(gas_industrial_by_community[dl.gas_cols], 2)
for label in range(k):
    plt.scatter(
            x=v[(labels['label']==label).values, 0],
            y=v[(labels['label']==label).values, 1]
    )
plt.legend([f"Clúster {i}" for i in range(k)])

path = images_path / Path("ensayos/clustering_gas_tipo_zona_agrupa_ciudades/
 ↪industrial")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("cluster_scatter.svg"))
```



[63]:
```
v = tsne(gas_industrial_by_community[dl.gas_cols], 3)

fig = go.Figure()
for label in range(k):
    fig.add_trace(
        go.Scatter3d(
            x=v[(labels['label']==label).values, 0],
```

```
            y=v[(labels['label']==label).values, 1],
            z=v[(labels['label']==label).values, 2]
        )
    )
fig.show()

path = images_path / Path("ensayos/clustering_gas_tipo_zona_agrupa_ciudades/
↪industrial")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("cluster_scatter3d.svg"))
```

```
<Figure size 576x396 with 0 Axes>
```

[ ]:
```
comunities_by_cluster = {}

for nlab in range(k):
    com_index = np.where(labels==nlab)[0]
    com_list = []
    for c in energy_commercial_by_community.values[com_index][:,0]:
        com_list.append(c)
    comunities_by_cluster[nlab] = com_list

for k in sorted(comunities_by_cluster, key=lambda k:␣
↪len(comunities_by_cluster[k])):
    print(f"Al clúster {k} pertenecen las comunidades:")
    for e in comunities_by_cluster[k]:
        print("\t", e)
```

1.3.4 Volver al inicio

1.3.5 Clustering de energía para cada comunidad observando patrones para cada bloque

En este caso seleccionaremos la comunidad con más consumo para cada tipo encontrada anteriormente, y estudiaremos los patrones de consumo entre todos los bloques de dicha comunidad para cada tipo de comunidad (industrial, comercial y residencial).

[65]:
```
energy_df = dl[["COMMUNITY AREA NAME", "CENSUS BLOCK", "BUILDING TYPE",␣
↪"BUILDING_SUBTYPE"] + dl.energy_cols]
communities = dl['COMMUNITY AREA NAME'].drop_duplicates()
communities
```

[65]:
```
0          Archer Heights
1                 Ashburn
2          Auburn Gresham
3                  Austin
```

```
12               Avondale
              …
591           West Elsdon
602         South Deering
607    Washington Heights
736       Mount Greenwood
740       Near South Side
Name: COMMUNITY AREA NAME, Length: 77, dtype: object
```

Comercial La comunidad *Loop* es la que más electricidad consumió, por lo que seleccionamos dicha comunidad.

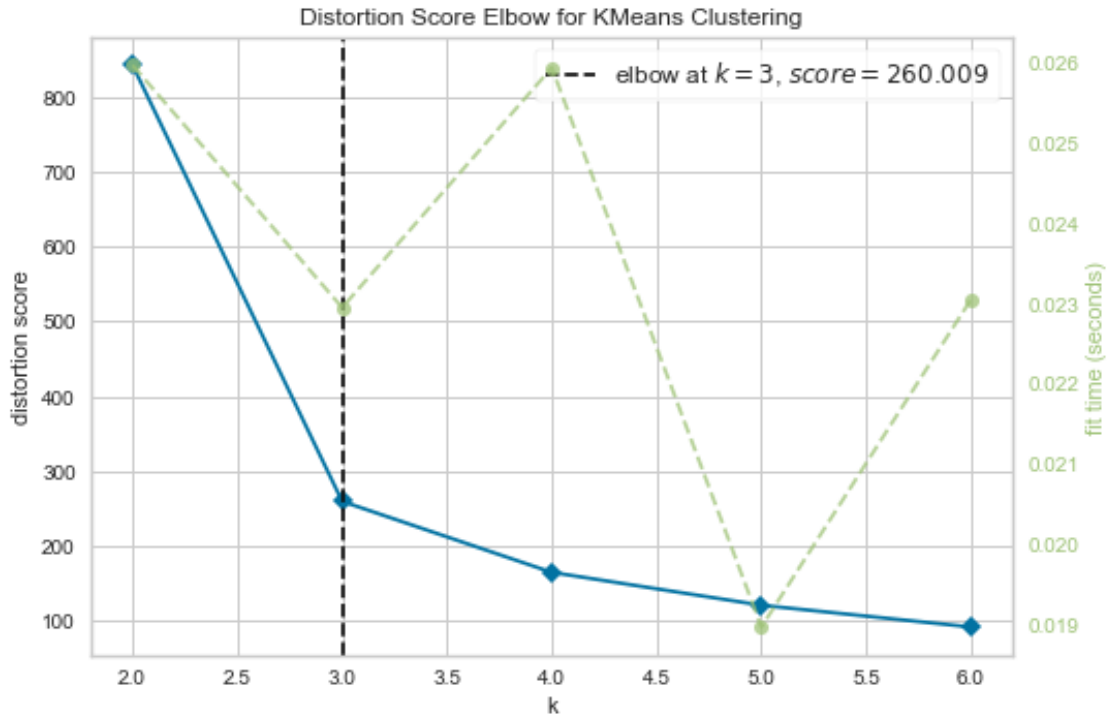
[66]:
```
commercial_loop = energy_df[(energy_df["COMMUNITY AREA NAME"] == "Loop") &␣
↪(energy_df["BUILDING TYPE"] == "Commercial")]
commercial_loop
```

[66]:
```
      COMMUNITY AREA NAME  CENSUS BLOCK BUILDING TYPE BUILDING_SUBTYPE  \
120                  Loop  1.703184e+14    Commercial         Multi 7+
121                  Loop  1.703184e+14    Commercial       Commercial
354                  Loop  1.703132e+14    Commercial       Commercial
945                  Loop  1.703184e+14    Commercial       Commercial
946                  Loop  1.703184e+14    Commercial        Municipal
…                     …           …            …               …
38591                Loop  1.703184e+14    Commercial       Commercial
38592                Loop  1.703184e+14    Commercial       Commercial
38593                Loop  1.703184e+14    Commercial       Commercial
38594                Loop  1.703184e+14    Commercial         Multi 7+
38595                Loop  1.703184e+14    Commercial       Commercial

       KWH JANUARY 2010  KWH FEBRUARY 2010  KWH MARCH 2010  KWH APRIL 2010  \
120             73300.0            73177.0         55407.0         40004.0
121           1937016.0          1612999.0       1378571.0       1299736.0
354           1931433.0          1702836.0       1708126.0       1922289.0
945           2370788.0          2119876.0       2082421.0       2087045.0
946           1775651.0          1433137.0       1393901.0       1171420.0
…                   …                  …               …               …
38591          306959.0           280224.0        134944.0         95925.0
38592           42600.0           485192.0        510282.0        522265.0
38593          247181.0           209955.0        213677.0        220961.0
38594           50822.0            46418.0         40971.0         35079.0
38595           51003.0            34568.0         29774.0         24527.0

       KWH MAY 2010  KWH JUNE 2010  KWH JULY 2010  KWH AUGUST 2010  \
120         41641.0        46163.0        58720.0          52091.0
121       1376510.0      1453082.0      1448453.0        1373392.0
354       2076298.0      1972560.0      2338763.0        2375021.0
945       2270125.0      3095521.0      3011561.0        2884957.0
```

```
946       1397811.0      1562200.0      1565775.0        1400773.0
…               …              …              …                …
38591      132642.0       150710.0       219940.0         187643.0
38592      607110.0       510410.0       593315.0         620349.0
38593      303981.0       308345.0       388278.0         370715.0
38594       47196.0        53944.0        77830.0          66578.0
38595       32431.0        37194.0        51768.0          42745.0

       KWH SEPTEMBER 2010  KWH OCTOBER 2010  KWH NOVEMBER 2010  \
120                53757.0           48746.0            75111.0
121              1313078.0         1180224.0          2100492.0
354              2122579.0         1554759.0          1863090.0
945              2371234.0         2128850.0          2341563.0
946              1353486.0         1327522.0          1697576.0
…                      …                 …                  …
38591             173423.0          137245.0           239216.0
38592             557672.0          512367.0           514068.0
38593             294093.0          242598.0           203601.0
38594              62182.0           41404.0            46031.0
38595              35956.0           26728.0            30434.0

       KWH DECEMBER 2010
120                119320.0
121               2229423.0
354               2003699.0
945               2729045.0
946               2130009.0
…                       …
38591              446574.0
38592              463212.0
38593              237975.0
38594               61226.0
38595               39684.0

[152 rows x 16 columns]
```

[67]:
```
kmax = 7
k_values = (2, kmax)

scaler = preprocessing.StandardScaler()

data = commercial_loop[dl.energy_cols]
norm_data = scaler.fit_transform(data)

km = KMeansCluster(norm_data)
km.cluster(k_values)
```

```
path = images_path / Path("ensayos/
 ↪clustering_energia_por_comunidad_observando_patrones/comercial")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("elbow.svg"))
```



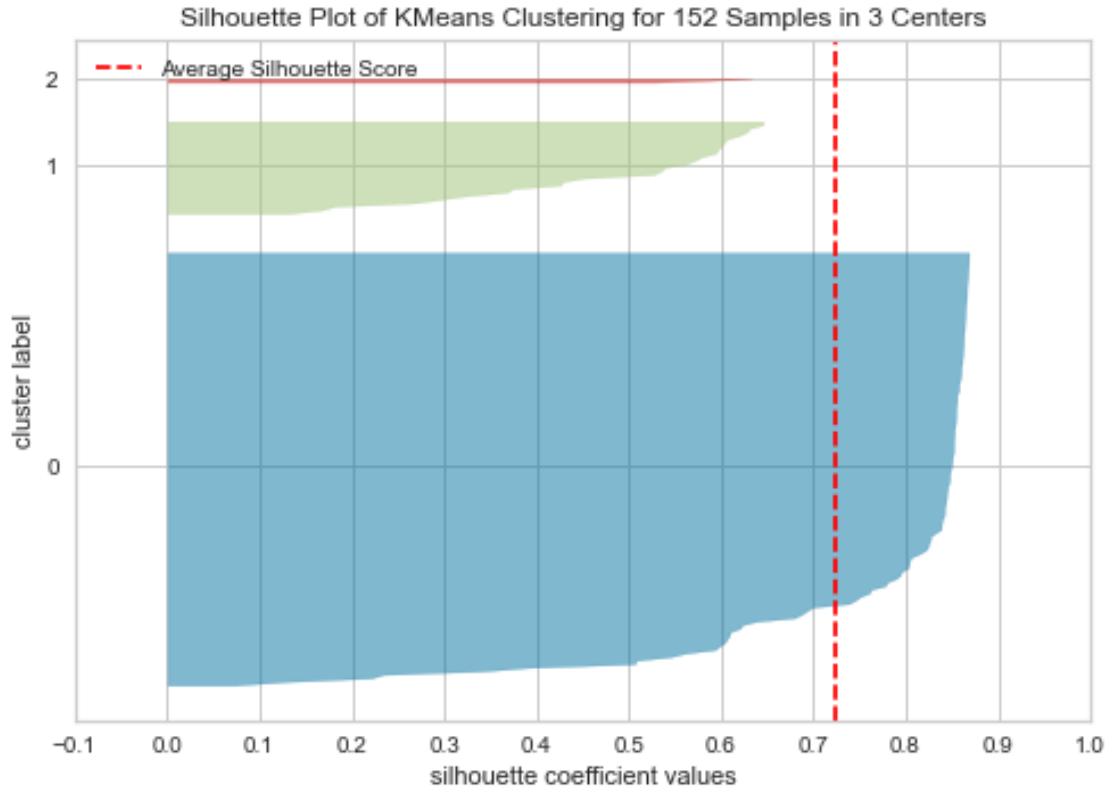

```
<Figure size 576x396 with 0 Axes>
```

Con k=3 tenemos una división óptima.

[68]:
```
km_best = km.get_kmeans_of(3)
km_labels = km_best.labels_
km_centroids = scaler.inverse_transform(km_best.cluster_centers_)

km_labels
```

[68]:
```
array([0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 2, 0, 1, 0, 0, 0, 0, 0, 0, 1,
       0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1,
       1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1,
       0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0,
       0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0, 1, 0,
       1, 0, 0, 0, 0, 1, 1, 1, 0, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0])
```

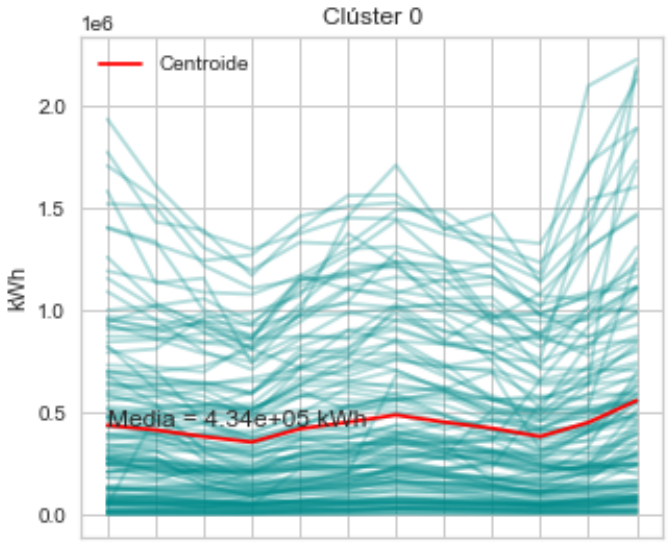
[69]:
```
_ = silhouette_visualizer(km_best, norm_data, colors='yellowbrick')

path = images_path / Path("ensayos/
 ↪clustering_energia_por_comunidad_observando_patrones/comercial")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("silhouette.svg"))
```

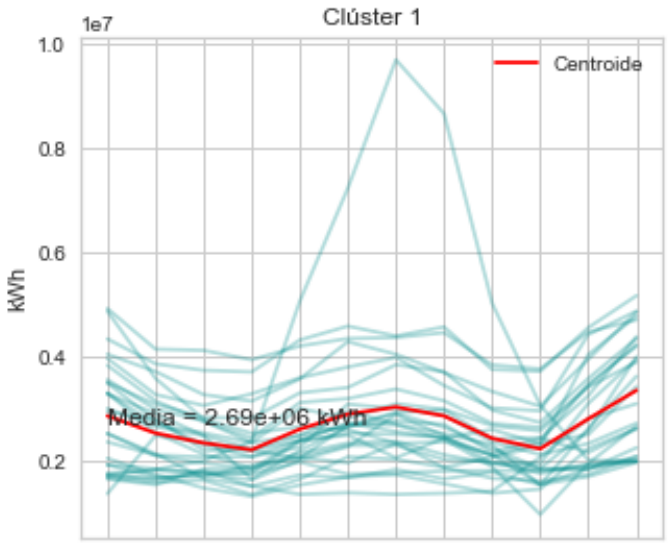


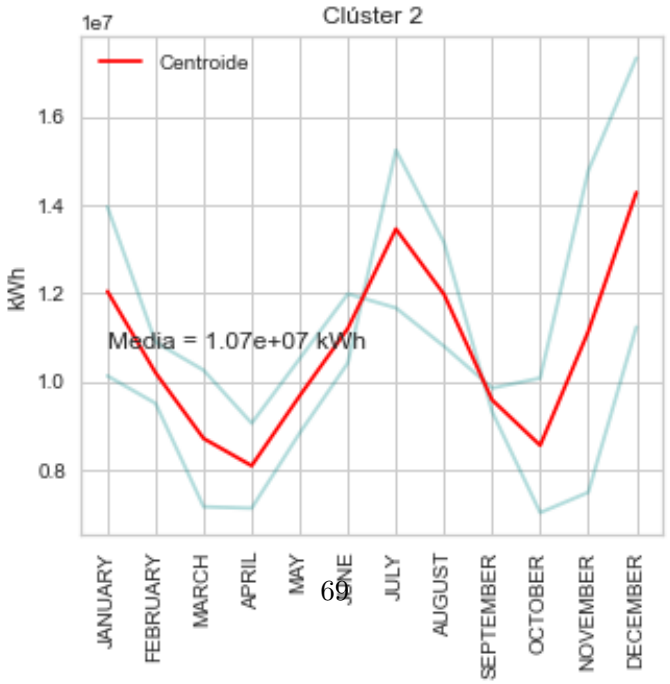

```
<Figure size 576x396 with 0 Axes>
```

[70]:
```
labels = pd.DataFrame(km_labels, columns=["label"])

k = km_centroids.shape[0]

f, axes = plt.subplots(k, 1, figsize=(5, 15), sharex=True)

for ki in range(k):
    cluster = commercial_loop.iloc[labels.query(f'label == {ki}').index][dl.
 ↪energy_cols]
    palette={i:"darkcyan" for i in range(cluster.shape[0])}

    ax0 = sns.lineplot(data=cluster.values.T, ax=axes[ki], palette=palette,␣
 ↪alpha=0.3)
```

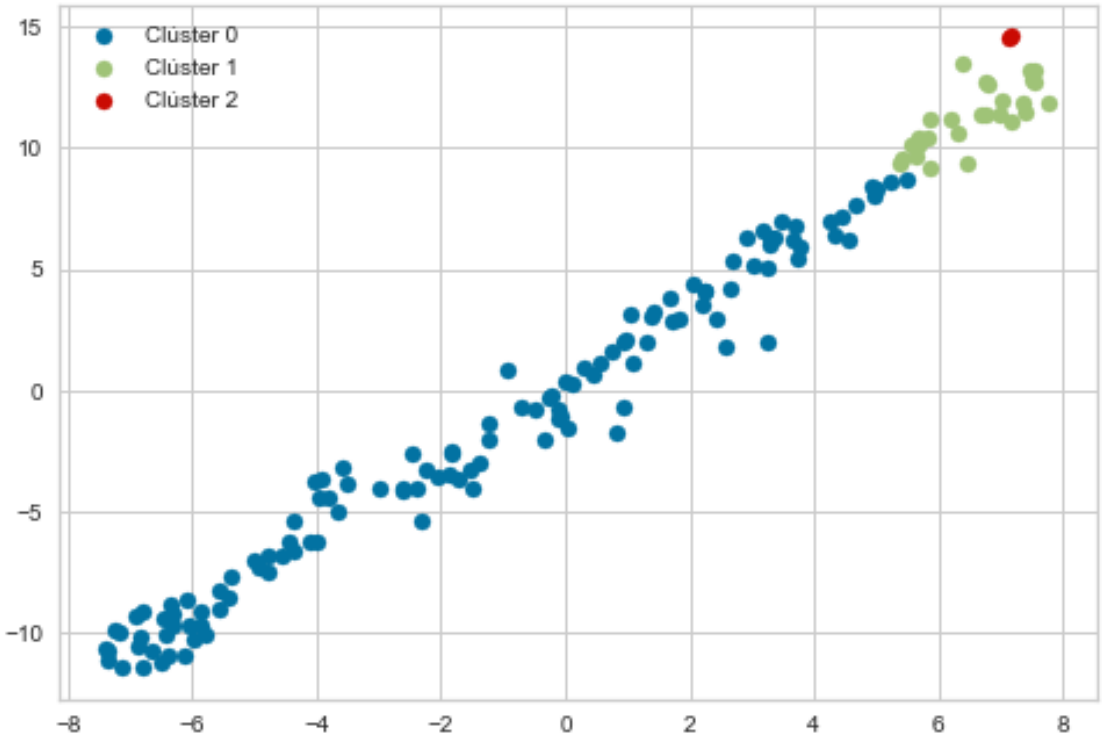
```
    _ = [line.set_linestyle("-") for line in ax0.lines]

    ax0c = sns.lineplot(data=km_centroids[ki, :].T, ax=axes[ki], color="red")
    _ = [line.set_linestyle("-") for line in ax0c.lines]

    ax0c.legend(ax0c.lines[-1:], ["Centroide"]);

    ax0.set_title(f"Clúster {ki}")
    Media = km_centroids[ki, :].mean()
    ax0.text(0, Media, f"{Media = :0.2e} kWh")
    ax0.set(ylabel="kWh")

    ax0.set_xticks([e for e in range(0, 12)])
    ax0.set_xticklabels([f"{datetime.date(2008, i, 1).strftime('%B').upper()}"␣
 ↪for i in range(1, 13)], rotation=90)

path = images_path / Path("ensayos/
 ↪clustering_energia_por_comunidad_observando_patrones/comercial")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("cluster_grafica.svg"))
```


Clúster 0
1e6
Centroide
2.0
1.5
1.0
0.5
0.0
kWh
Media = 4.34e+05 kWh


Clúster 1
1e7
Centroide
1.0
0.8
0.6
0.4
0.2
kWh
Media = 2.69e+06 kWh


Clúster 2
1e7
Centroide
1.6
1.4
1.2
1.0
0.8
kWh
Media = 1.07e+07 kWh
JANUARY
FEBRUARY
MARCH
APRIL
MAY
JUNE
JULY
AUGUST
SEPTEMBER
OCTOBER
NOVEMBER
DECEMBER

[71]:
```
v = tsne(commercial_loop[dl.energy_cols], 2)
for label in range(k):
    plt.scatter(
            x=v[(labels['label']==label).values, 0],
            y=v[(labels['label']==label).values, 1]
    )
plt.legend([f"Clúster {i}" for i in range(k)])

path = images_path / Path("ensayos/
 ↪clustering_energia_por_comunidad_observando_patrones/comercial")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("cluster_scatter.svg"))
```



[ ]:
```
comunities_by_cluster = {}

for nlab in range(k):
    com_index = np.where(labels==nlab)[0]
    com_list = []
    for c in commercial_loop.values[com_index][:,(0,1,3)]:
        com_list.append(c)
```

```
    comunities_by_cluster[nlab] = com_list

for k in sorted(comunities_by_cluster, key=lambda k:␣
 ↪len(comunities_by_cluster[k])):
    print(f"Al clúster {k} pertenecen las comunidades:")
    for e in comunities_by_cluster[k]:
        print("\t", e)
```

Residencial En las zonas residenciales, *Loop* también fue la que más consumió.

[73]:
```
residential_loop = energy_df[(energy_df["COMMUNITY AREA NAME"] == "Loop") &␣
 ↪(energy_df["BUILDING TYPE"] == "Residential")]
residential_loop
```

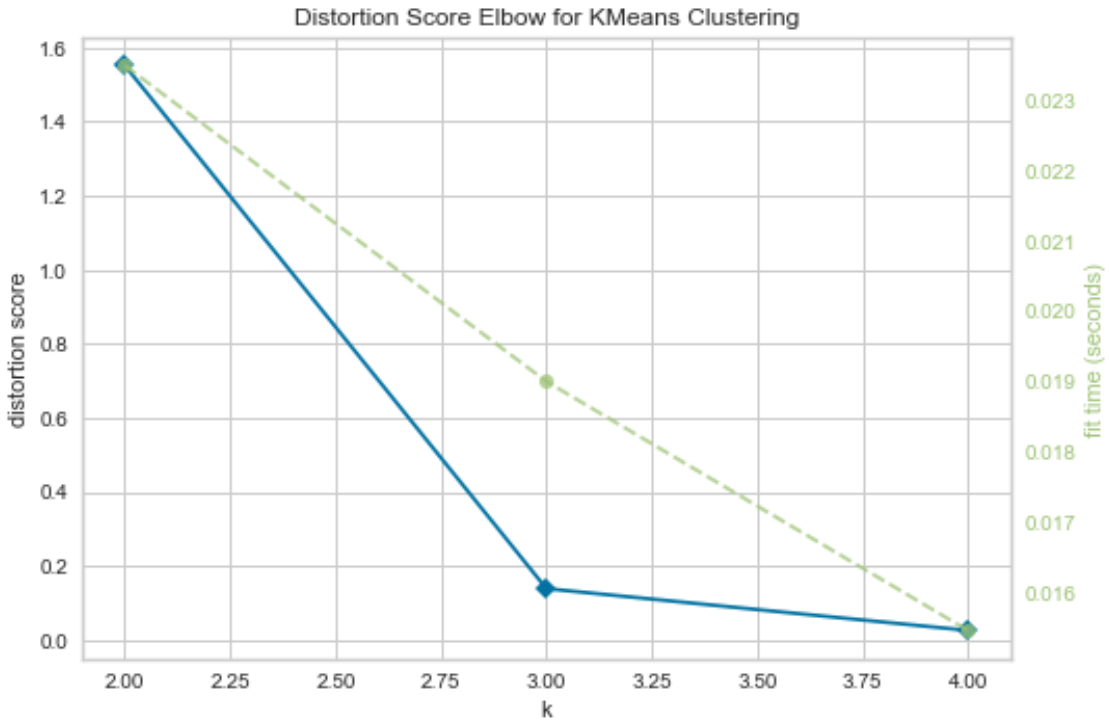
[73]:

	COMMUNITY AREA NAME	CENSUS BLOCK	BUILDING TYPE	BUILDING_SUBTYPE	KWH JANUARY 2010	KWH FEBRUARY 2010	KWH MARCH 2010	KWH APRIL 2010	KWH MAY 2010	KWH JUNE 2010	KWH JULY 2010	KWH AUGUST 2010	KWH SEPTEMBER 2010	KWH OCTOBER 2010	KWH NOVEMBER 2010	KWH DECEMBER 2010
21904	Loop	1.703132e+14	Residential	Multi < 7	4576.0	5264.0	4321.0	3977.0	5025.0	7709.0	8728.0	8350.0	9765.0	6754.0	6048.0	6631.0
38489	Loop	1.703132e+14	Residential	Multi 7+	12625.0	12072.0	11304.0	10223.0	12654.0	13221.0	19326.0	15676.0	11690.0	9369.0	13157.0	14402.0
38491	Loop	1.703132e+14	Residential	Multi 7+	31735.0	31549.0	30020.0	27812.0	38957.0	43405.0	57833.0	54271.0	39998.0	35803.0	34729.0	44797.0
38501	Loop	1.703132e+14	Residential	Multi 7+	150389.0	156138.0	153203.0	162340.0	223682.0	240178.0	329181.0	309317.0	224617.0	194145.0	221436.0	269572.0
38513	Loop	1.703184e+14	Residential	Single Family	690.0	887.0	1248.0	881.0	1724.0	1910.0	2020.0	1439.0	620.0	371.0	2386.0	

```
38513             1112.0
```

[74]:
```
kmax = 5
k_values = (2, kmax)

scaler = preprocessing.StandardScaler()
data = residential_loop[dl.energy_cols]
norm_data = scaler.fit_transform(data)

km = KMeansCluster(norm_data)
km.cluster(k_values)

path = images_path / Path("ensayos/
 ↪clustering_energia_por_comunidad_observando_patrones/residencial")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("elbow.svg"))
```

```
C:\Users\Alberto\miniconda3\envs\cdi-musiani-env\lib\site-
packages\yellowbrick\utils\kneed.py:140: YellowbrickWarning:

No 'knee' or 'elbow point' detected This could be due to bad clustering, no
actual clusters being formed etc.

C:\Users\Alberto\miniconda3\envs\cdi-musiani-env\lib\site-
packages\yellowbrick\cluster\elbow.py:343: YellowbrickWarning:

No 'knee' or 'elbow' point detected, pass `locate_elbow=False` to remove the
warning
```



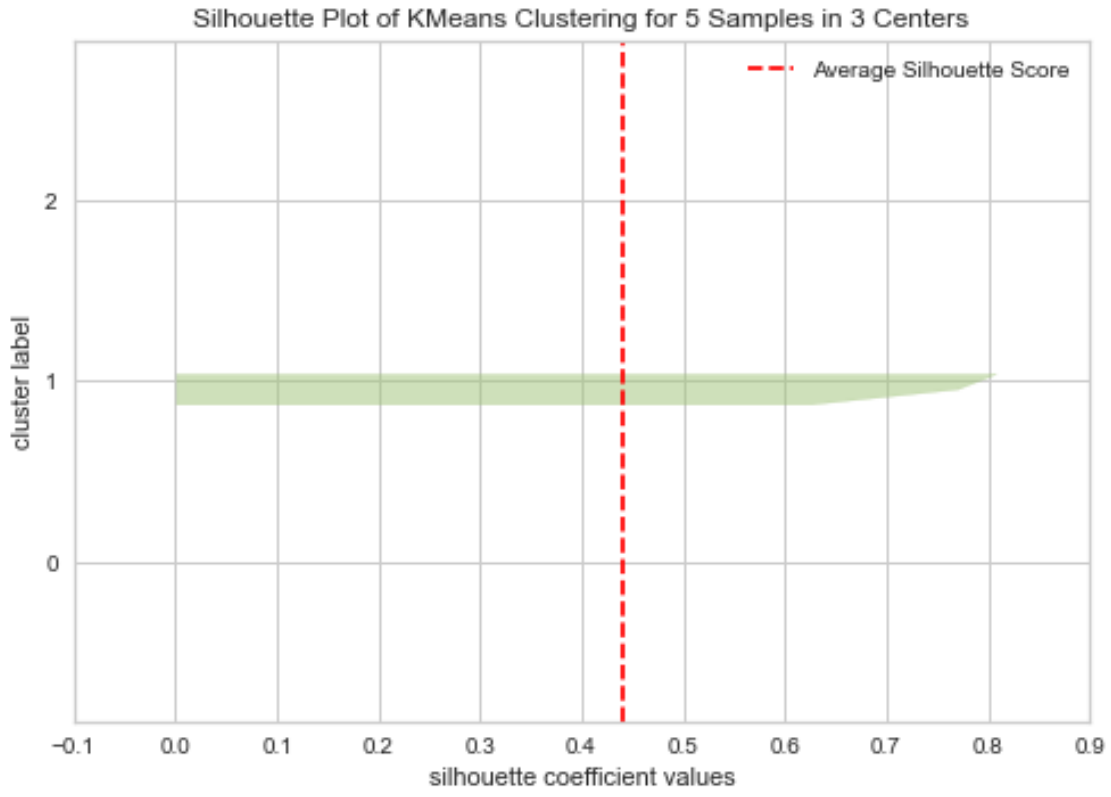

```
<Figure size 576x396 with 0 Axes>
```

Seleccionaremos K=3.

[75]:
```
km_best = km.get_kmeans_of(3)
km_labels = km_best.labels_
km_centroids = scaler.inverse_transform(km_best.cluster_centers_)

km_labels
```

[75]:
```
array([1, 1, 2, 0, 1])
```

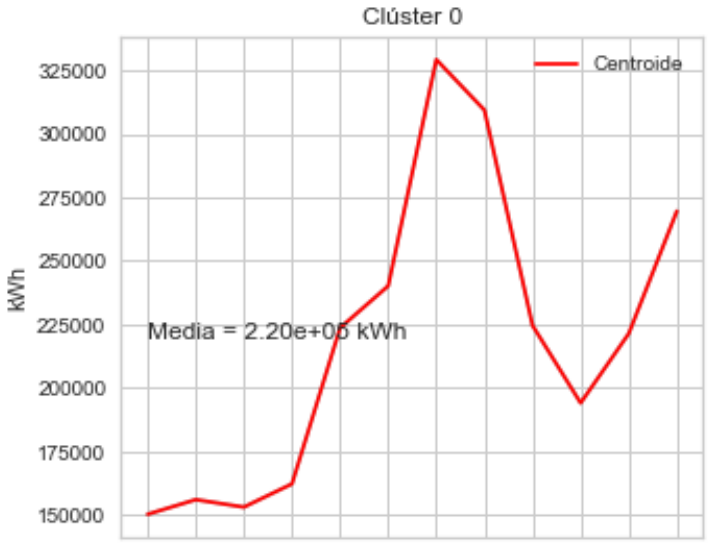
[76]:
```
_ = silhouette_visualizer(km_best, norm_data, colors='yellowbrick')

path = images_path / Path("ensayos/
 ↪clustering_energia_por_comunidad_observando_patrones/residencial")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("silhouette.svg"))
```

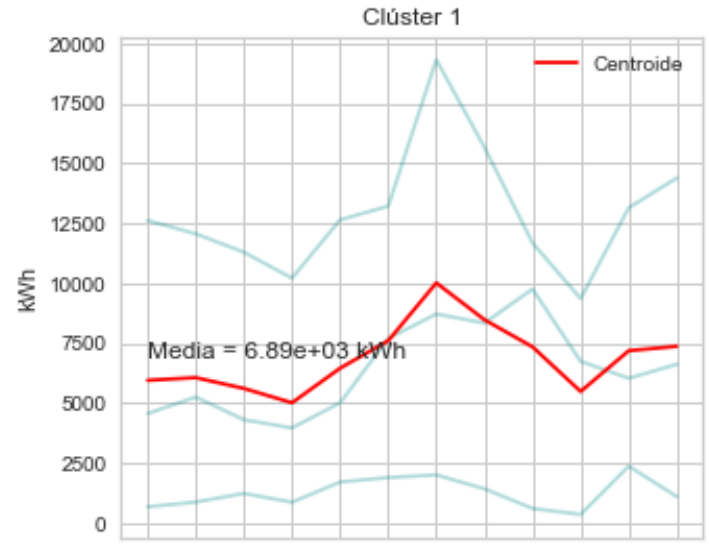


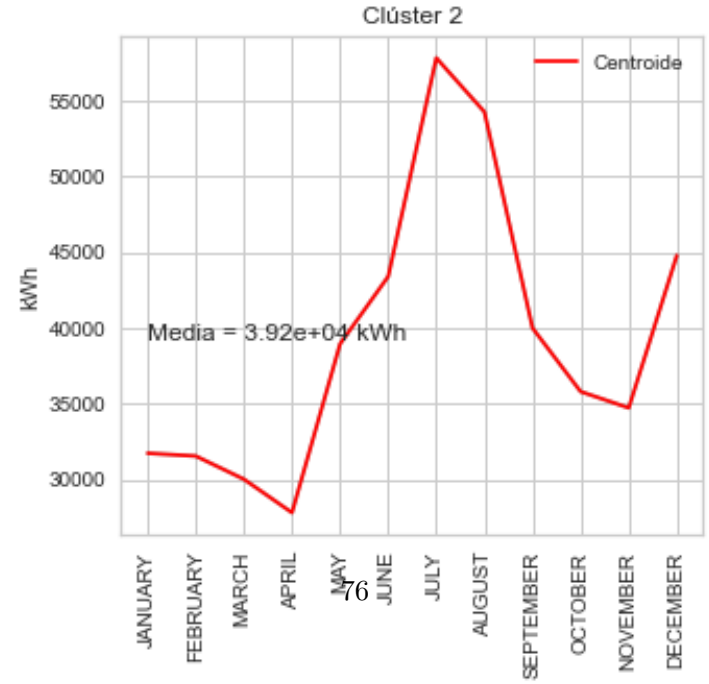

```
<Figure size 576x396 with 0 Axes>
```

[77]:
```
labels = pd.DataFrame(km_labels, columns=["label"])

k = km_centroids.shape[0]

f, axes = plt.subplots(k, 1, figsize=(5, 15), sharex=True)

for ki in range(k):
    cluster = residential_loop.iloc[labels.query(f'label == {ki}').index][dl.
 ↪energy_cols]
    palette={i:"darkcyan" for i in range(cluster.shape[0])}

    ax0 = sns.lineplot(data=cluster.values.T, ax=axes[ki], palette=palette,␣
 ↪alpha=0.3)
    _ = [line.set_linestyle("-") for line in ax0.lines]

    ax0c = sns.lineplot(data=km_centroids[ki, :].T, ax=axes[ki], color="red")
    _ = [line.set_linestyle("-") for line in ax0c.lines]

    ax0c.legend(ax0c.lines[-1:], ["Centroide"]);
```

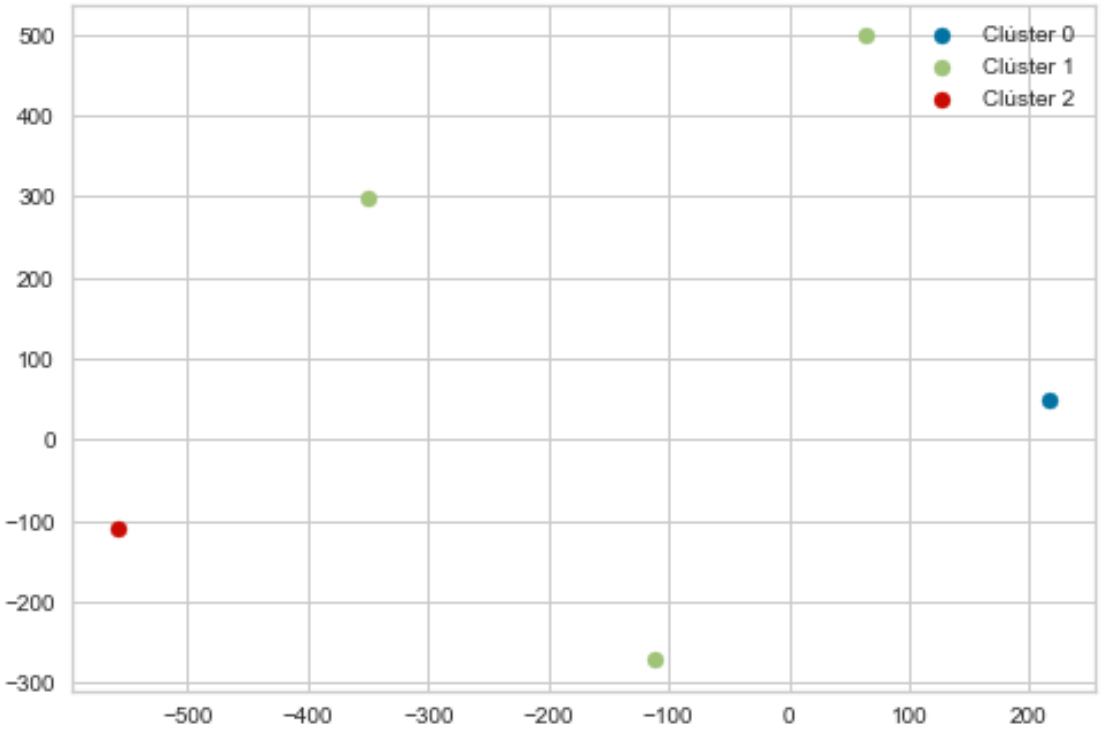
```
    ax0.set_title(f"Clúster {ki}")
    Media = km_centroids[ki, :].mean()
    ax0.text(0, Media, f"{Media = :0.2e} kWh")
    ax0.set(ylabel="kWh")

    ax0.set_xticks([e for e in range(0, 12)])
    ax0.set_xticklabels([f"{datetime.date(2008, i, 1).strftime('%B').upper()}"␣
 ↪for i in range(1, 13)], rotation=90)

path = images_path / Path("ensayos/
 ↪clustering_energia_por_comunidad_observando_patrones/residencial")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("cluster_grafica.svg"))
```


Clúster 0
Centroide
kWh
325000
300000
275000
250000
225000
200000
175000
150000
Media = 2.20e+05 kWh


Clúster 1
Centroide
kWh
20000
17500
15000
12500
10000
7500
5000
2500
0
Media = 6.89e+03 kWh


Clúster 2
Centroide
kWh
55000
50000
45000
40000
35000
30000
Media = 3.92e+04 kWh
JANUARY
FEBRUARY
MARCH
APRIL
MAY
JUNE
JULY
AUGUST
SEPTEMBER
OCTOBER
NOVEMBER
DECEMBER

[78]:
```
v = tsne(residential_loop[dl.energy_cols], 2)
for label in range(k):
    plt.scatter(
            x=v[(labels['label']==label).values, 0],
            y=v[(labels['label']==label).values, 1]
    )
plt.legend([f"Clúster {i}" for i in range(k)])

path = images_path / Path("ensayos/
 ↪clustering_energia_por_comunidad_observando_patrones/residencial")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("cluster_scatter.svg"))
```



[79]:
```
v = tsne(residential_loop[dl.energy_cols], 3)

fig = go.Figure()
for label in range(k):
    fig.add_trace(
        go.Scatter3d(
            x=v[(labels['label']==label).values, 0],
```

```
            y=v[(labels['label']==label).values, 1],
            z=v[(labels['label']==label).values, 2]
        )
    )
fig.show()

path = images_path / Path("ensayos/
 ↪clustering_energia_por_comunidad_observando_patrones/residencial")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("cluster_scatter3d.svg"))
```

```
<Figure size 576x396 with 0 Axes>
```

[ ]:
```
comunities_by_cluster = {}

for nlab in range(k):
    com_index = np.where(labels==nlab)[0]
    com_list = []
    for c in residential_loop.values[com_index][:,(0,1,3)]:
        com_list.append(c)
    comunities_by_cluster[nlab] = com_list

for k in sorted(comunities_by_cluster, key=lambda k:␣
 ↪len(comunities_by_cluster[k])):
    print(f"Al clúster {k} pertenecen las comunidades:")
    for e in comunities_by_cluster[k]:
        print("\t", e)
```

Industrial En el caso de las zonas industriales, *Near West Side* fue una de las que más consumió electricidad.

[81]:
```
industrial_loop = energy_df[(energy_df["COMMUNITY AREA NAME"] == "Near West␣
 ↪Side") & (energy_df["BUILDING TYPE"] == "Industrial")]
industrial_loop
```

[81]:

	COMMUNITY AREA NAME	CENSUS BLOCK	BUILDING TYPE	BUILDING_SUBTYPE	\
16299	Near West Side	1.703128e+14	Industrial	Industrial	
42543	Near West Side	1.703183e+14	Industrial	Industrial	
42571	Near West Side	1.703183e+14	Industrial	Industrial	

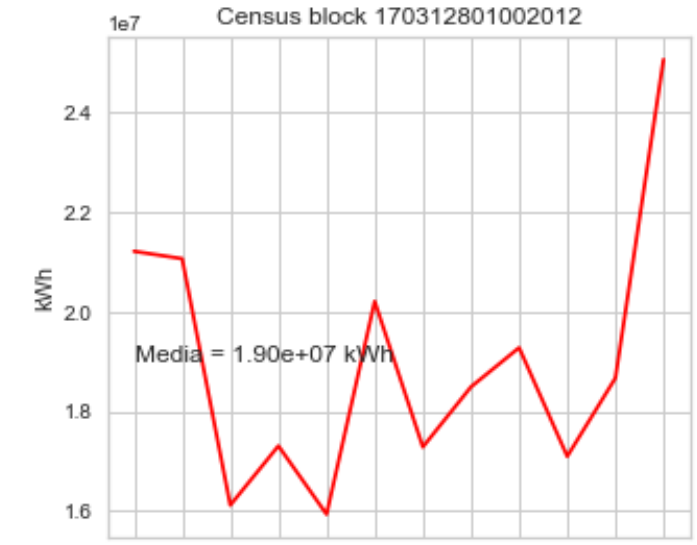
	KWH JANUARY 2010	KWH FEBRUARY 2010	KWH MARCH 2010	KWH APRIL 2010	\
16299	21214017.0	21065500.0	16121105.0	17310058.0	
42543	25445.0	22534.0	20547.0	30382.0	
42571	5966.0	3779.0	2893.0	2219.0	

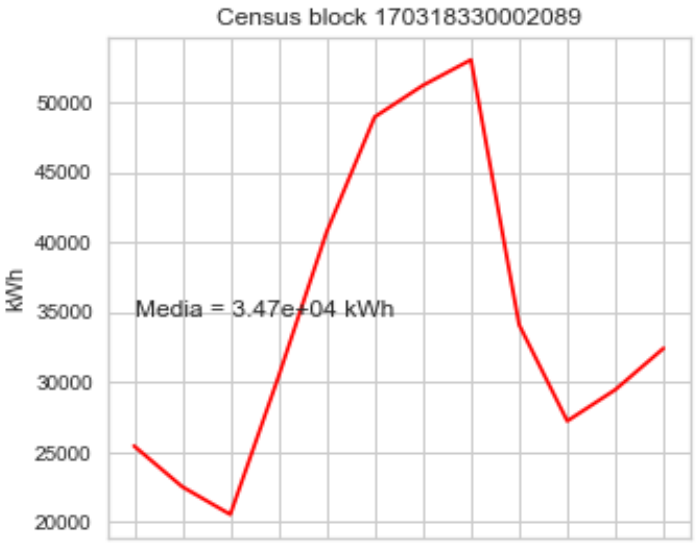
	KWH MAY 2010	KWH JUNE 2010	KWH JULY 2010	KWH AUGUST 2010	\
16299	15938553.0	20209197.0	17287870.0	18493853.0	

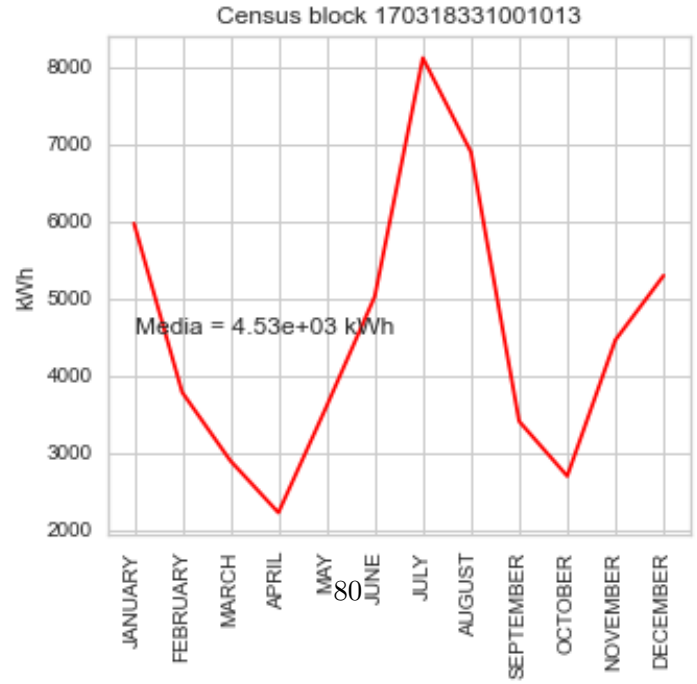
```
42543       40722.0        48960.0        51211.0          53044.0
42571        3597.0         5018.0         8110.0           6890.0

       KWH SEPTEMBER 2010  KWH OCTOBER 2010  KWH NOVEMBER 2010  \
16299          19280342.0        17100282.0         18671279.0
42543             34103.0           27225.0            29469.0
42571              3403.0            2693.0             4456.0

       KWH DECEMBER 2010
16299         25060008.0
42543            32424.0
42571             5290.0
```

Existen pocas muestras para aplicar clustering, por lo que mostramos cada una de las gráficas directamente.

[82]:
```
f, axes = plt.subplots(3, 1, figsize=(5, 15), sharex=True)

for i, (census_num, (index, row)) in enumerate(zip(industrial_loop['CENSUS␣
 ↪BLOCK'], industrial_loop[dl.energy_cols].iterrows())):
    ax0 = sns.lineplot(data=row.T, ax=axes[i], color="red")

    ax0.set_xticks([e for e in range(0, 12)])
    ax0.set_xticklabels([f"{datetime.date(2008, i, 1).strftime('%B').upper()}"␣
 ↪for i in range(1, 13)], rotation=90)

    ax0.set_title(f"Census block {int(census_num)}")
    Media = row.mean()
    ax0.text(0, Media, f"{Media = :0.2e} kWh")

    ax0.set(ylabel="kWh")

path = images_path / Path("ensayos/
 ↪clustering_energia_por_comunidad_observando_patrones/industrial")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("cluster_grafica.svg"))
```


Census block 170312801002012
1e7
2.4
2.2
2.0
1.8
1.6
kWh
Media = 1.90e+07 kWh


Census block 170318330002089
50000
45000
40000
35000
30000
25000
20000
kWh
Media = 3.47e+04 kWh


Census block 170318331001013
8000
7000
6000
5000
4000
3000
2000
kWh
Media = 4.53e+03 kWh
JANUARY
FEBRUARY
MARCH
APRIL
MAY
JUNE
JULY
AUGUST
SEPTEMBER
OCTOBER
NOVEMBER
DECEMBER

1.3.6 Volver al inicio

1.3.7 Clustering de gas para cada comunidad observando patrones en cada bloque

Igual que en el caso anterior, seleccionaremos la comunidad con más consumo para cada tipo encontrada, y estudiaremos los patrones de consumo entre todos los bloques de dicha comunidad para cada tipo de comunidad (industrial, comercial y residencial).

[83]:
```
gas_df = dl[["COMMUNITY AREA NAME", "CENSUS BLOCK", "BUILDING TYPE",␣
 ↪"BUILDING_SUBTYPE"] + dl.gas_cols]
gas_df
```

[83]:

	COMMUNITY AREA NAME	CENSUS BLOCK	BUILDING TYPE	BUILDING_SUBTYPE	\
0	Archer Heights	1.703157e+14	Residential	Multi < 7	
6	Austin	1.703125e+14	Commercial	Commercial	
11	Austin	1.703125e+14	Commercial	Multi < 7	
13	Avondale	1.703121e+14	Commercial	Multi < 7	
19	Avondale	1.703121e+14	Residential	Multi < 7	
…	…	…	…	…	
67046	Woodlawn	1.703184e+14	Residential	Single Family	
67047	Woodlawn	1.703184e+14	Commercial	Multi < 7	
67048	Woodlawn	1.703184e+14	Residential	Multi < 7	
67049	Woodlawn	1.703184e+14	Residential	Single Family	
67050	Woodlawn	1.703184e+14	Residential	Multi < 7	

	THERM JANUARY 2010	THERM FEBRUARY 2010	THERM MARCH 2010	\
0	2326.0	2131.0	1400.0	
6	3041.0	2680.0	1151.0	
11	910.0	738.0	632.0	
13	535.0	458.0	481.0	
19	285.0	285.0	244.0	
…	…	…	…	
67046	2166.0	1681.0	1858.0	
67047	985.0	1152.0	1238.0	
67048	2202.0	1874.0	1647.0	
67049	95.0	11.0	47.0	
67050	2372.0	1787.0	1449.0	

	THERM APRIL 2010	THERM MAY 2010	THERM JUNE 2010	THERM JULY 2010	\
0	620.0	502.0	224.0	222.0	
6	373.0	124.0	26.0	29.0	
11	448.0	254.0	113.0	42.0	
13	307.0	194.0	145.0	119.0	

```
19                129.0            73.0             46.0             39.0
…                 …                …                …                …
67046            1172.0           708.0            360.0             72.0
67047             630.0           475.0            192.0            141.0
67048             906.0           645.0            346.0             84.0
67049               9.0            45.0             18.0             22.0
67050             718.0           572.0            286.0            155.0

       THERM AUGUST 2010  THERM SEPTEMBER 2010  THERM OCTOBER 2010  \
0                  187.0                 197.0               252.0
6                   25.0                  49.0               177.0
11                  29.0                  33.0                36.0
13                 102.0                  88.0               109.0
19                  27.0                  39.0                29.0
…                   …                     …                   …
67046               67.0                  77.0               185.0
67047              162.0                 144.0               210.0
67048              150.0                 150.0               260.0
67049                9.0                  17.0                11.0
67050              134.0                 161.0               303.0

       THERM NOVEMBER 2010  THERM DECEMBER 2010
0                    744.0               2112.0
6                    670.0               3895.0
11                   166.0                633.0
13                   141.0                249.0
19                    50.0                144.0
…                     …                   …
67046                623.0               1800.0
67047                653.0               1744.0
67048                694.0               1335.0
67049                 18.0                 13.0
67050                588.0               1469.0

[60736 rows x 16 columns]
```

Comercial Entre las zonas comerciales, seleccionamos a *Loop*.

[84]:
```
commercial_loop = gas_df[(gas_df["COMMUNITY AREA NAME"] == "Loop") &␣
↪(gas_df["BUILDING TYPE"] == "Commercial")]
commercial_loop
```

[84]:
```
      COMMUNITY AREA NAME  CENSUS BLOCK BUILDING TYPE BUILDING_SUBTYPE  \
38278                Loop  1.703132e+14    Commercial       Commercial
38421                Loop  1.703132e+14    Commercial        Municipal
38441                Loop  1.703132e+14    Commercial       Commercial
38442                Loop  1.703132e+14    Commercial        Municipal
```

38443	Loop	1.703132e+14	Commercial	Commercial
…	…	…	…	…
38590	Loop	1.703184e+14	Commercial	Commercial
38591	Loop	1.703184e+14	Commercial	Commercial
38592	Loop	1.703184e+14	Commercial	Commercial
38593	Loop	1.703184e+14	Commercial	Commercial
38595	Loop	1.703184e+14	Commercial	Commercial

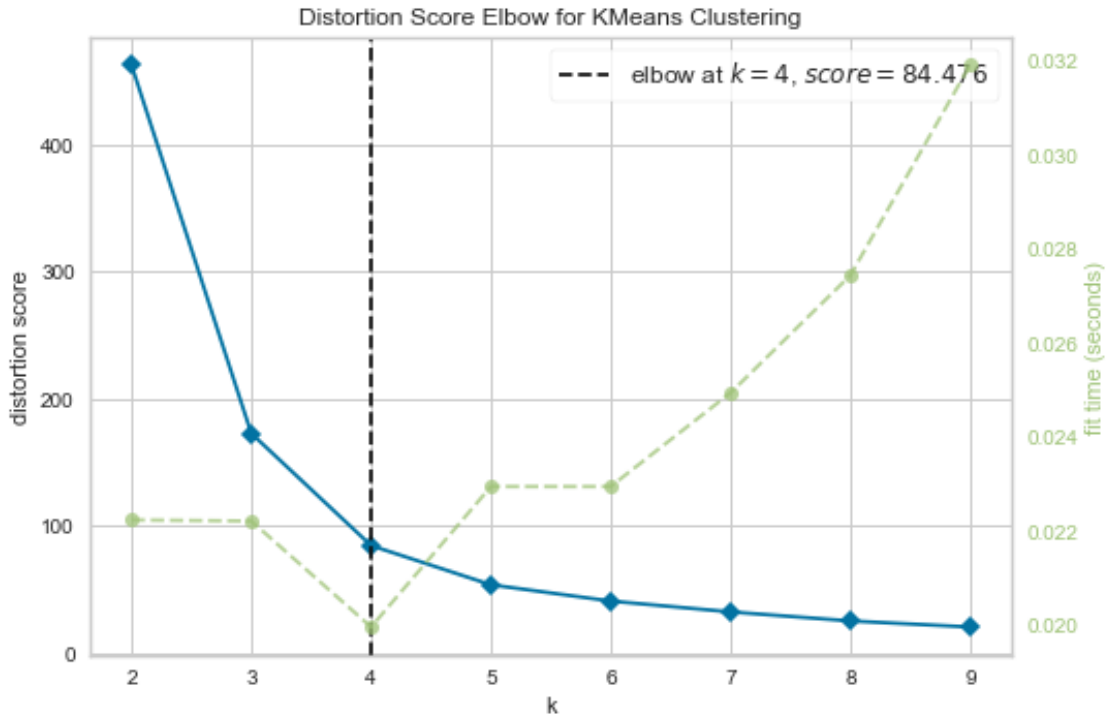
	THERM JANUARY 2010	THERM FEBRUARY 2010	THERM MARCH 2010	\
38278	3442.0	2174.0	2078.0	
38421	547.0	4673.0	2140.0	
38441	26533.0	22582.0	19806.0	
38442	37793.0	33336.0	23741.0	
38443	33314.0	22873.0	7441.0	
…	…	…	…	
38590	61037.0	64632.0	42911.0	
38591	30412.0	30380.0	23747.0	
38592	12883.0	11253.0	10453.0	
38593	25955.0	20581.0	13942.0	
38595	5362.0	7580.0	3591.0	

	THERM APRIL 2010	THERM MAY 2010	THERM JUNE 2010	THERM JULY 2010	\
38278	555.0	519.0	523.0	481.0	
38421	1982.0	1712.0	1687.0	1977.0	
38441	16489.0	10517.0	6900.0	16185.0	
38442	7320.0	4528.0	1217.0	1173.0	
38443	3285.0	2171.0	396.0	474.0	
…	…	…	…	…	
38590	21841.0	13181.0	6048.0	5479.0	
38591	5349.0	152.0	258.0	108.0	
38592	4450.0	1684.0	103.0	124.0	
38593	8535.0	6481.0	3076.0	2967.0	
38595	2828.0	1361.0	679.0	709.0	

	THERM AUGUST 2010	THERM SEPTEMBER 2010	THERM OCTOBER 2010	\
38278	543.0	515.0	473.0	
38421	1873.0	1779.0	2129.0	
38441	12766.0	12725.0	14340.0	
38442	1146.0	898.0	4245.0	
38443	311.0	576.0	3903.0	
…	…	…	…	
38590	6179.0	5863.0	8865.0	
38591	93.0	287.0	603.0	
38592	140.0	101.0	1222.0	
38593	1428.0	4041.0	7933.0	
38595	687.0	641.0	1037.0	

```
       THERM NOVEMBER 2010  THERM DECEMBER 2010
38278                768.0                1941.0
38421               2126.0                2720.0
38441              16231.0               25151.0
38442              18462.0               35026.0
38443               8656.0               28522.0
…                      …                    …
38590              22988.0               60728.0
38591               3774.0               48667.0
38592               6670.0               11681.0
38593              13596.0               25341.0
38595               2247.0                4626.0

[132 rows x 16 columns]
```

[85]:
```
kmax = 10
k_values = (2, kmax)

scaler = preprocessing.StandardScaler()
data = commercial_loop[dl.gas_cols]
norm_data = scaler.fit_transform(data)

km = KMeansCluster(norm_data)
km.cluster(k_values)

path = images_path / Path("ensayos/
 ↪clustering_gas_por_comunidad_observando_patrones/comercial")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("elbow.svg"))
```

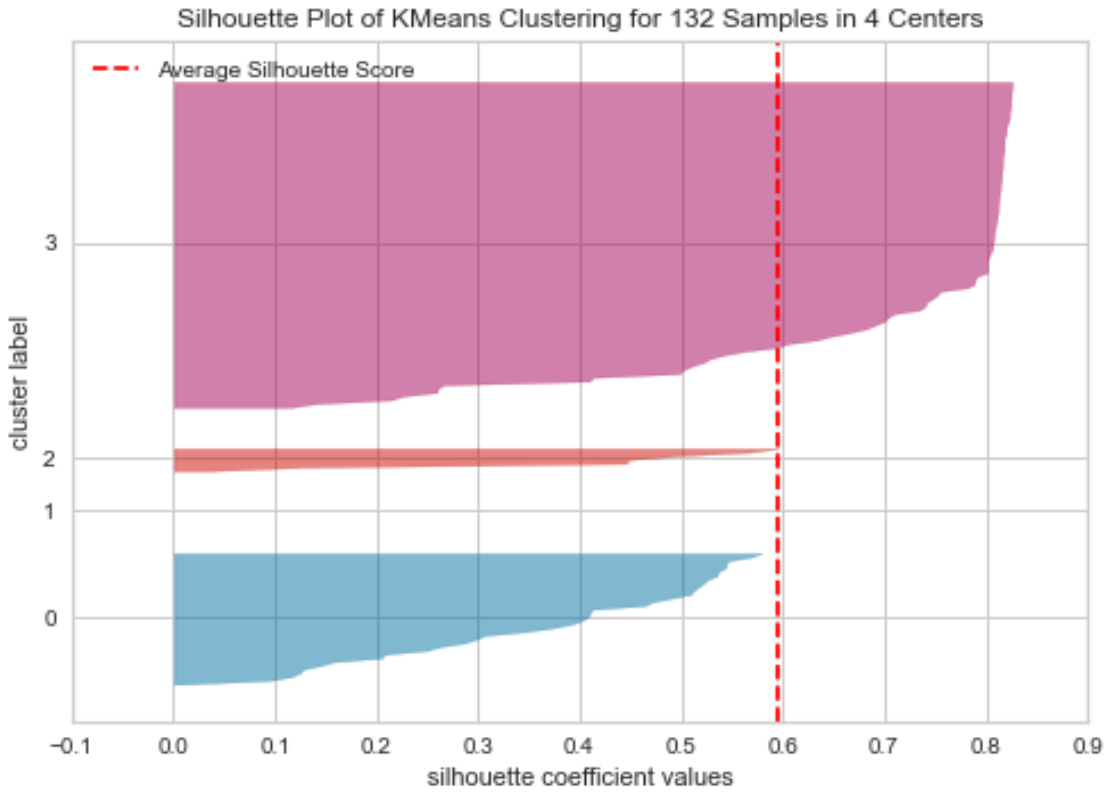



```
<Figure size 576x396 with 0 Axes>
```

[86]:
```
km_best = km.get_kmeans_of(4)
km_labels = km_best.labels_
km_centroids = scaler.inverse_transform(km_best.cluster_centers_)

km_labels
```

[86]:
```
array([3, 3, 3, 3, 3, 3, 0, 0, 2, 3, 2, 3, 3, 3, 3, 3, 3, 0, 0, 0, 3,
       1, 0, 3, 2, 0, 2, 3, 0, 0, 2, 2, 0, 3, 3, 3, 0, 3, 3, 0, 0, 3, 0,
       3, 3, 3, 3, 0, 3, 3, 3, 0, 3, 3, 3, 0, 3, 3, 0, 3, 3, 0, 3, 3, 3,
       3, 3, 3, 3, 3, 3, 3, 3, 0, 3, 3, 3, 3, 0, 3, 0, 3, 0, 3, 3, 3, 0,
       0, 3, 3, 0, 3, 3, 3, 0, 3, 0, 0, 0, 0, 3, 3, 3, 3, 3, 3, 3, 0, 0,
       3, 2, 3, 3, 3, 0, 3, 3, 3, 0, 3, 3, 3, 3, 3, 3, 3, 0, 3, 3, 3, 3])
```

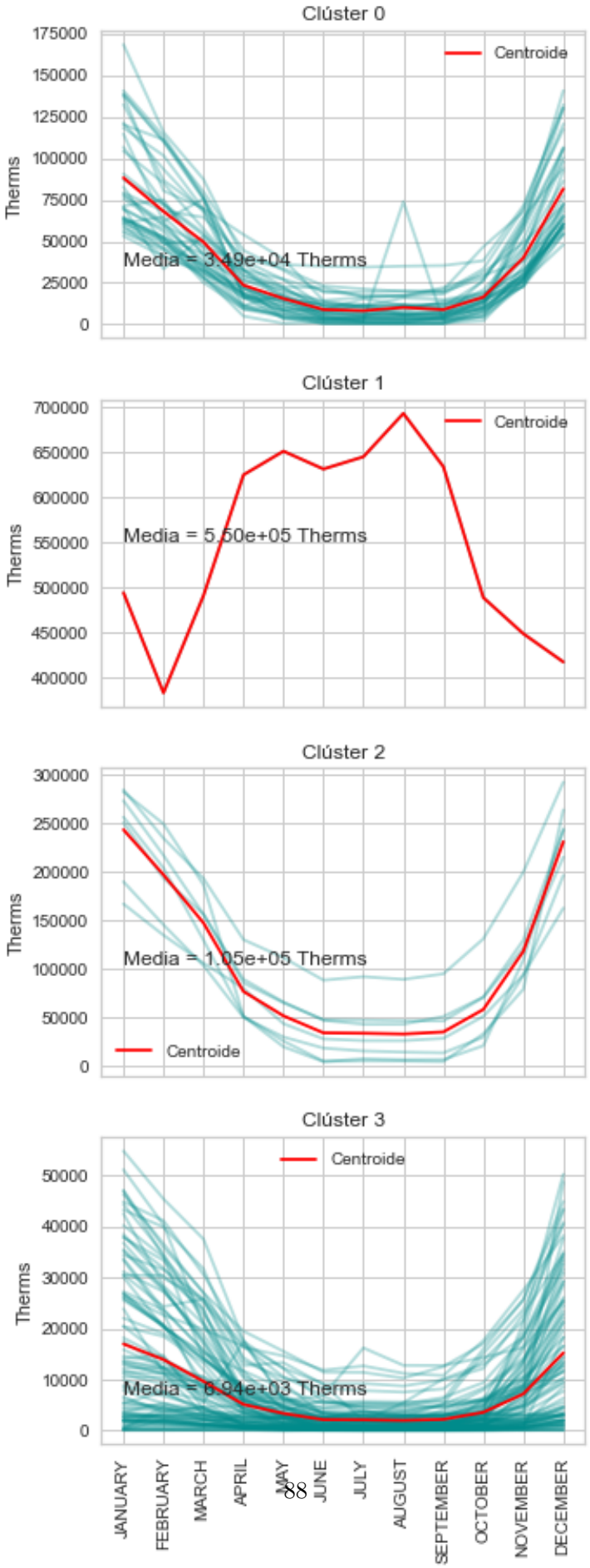
[87]:
```
_ = silhouette_visualizer(km_best, norm_data, colors='yellowbrick')

path = images_path / Path("ensayos/
 ↪clustering_gas_por_comunidad_observando_patrones/comercial")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("silhouette.svg"))
```



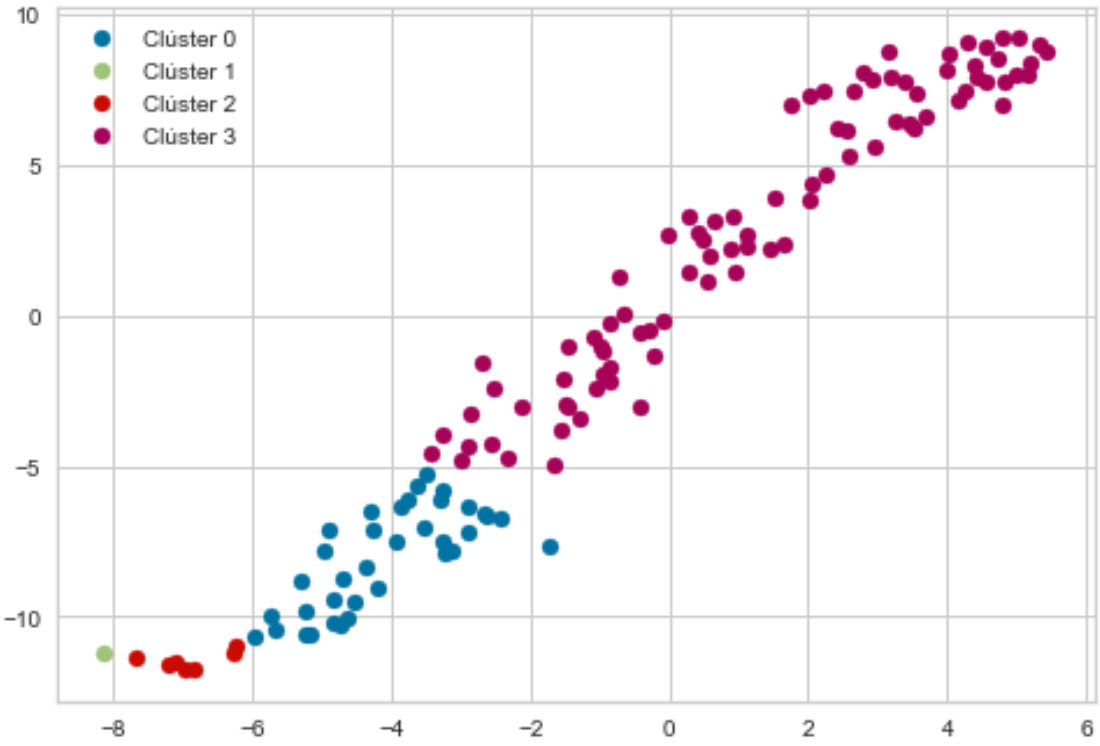

```
<Figure size 576x396 with 0 Axes>
```

[88]:
```
labels = pd.DataFrame(km_labels, columns=["label"])

k = km_centroids.shape[0]

f, axes = plt.subplots(k, 1, figsize=(5, 15), sharex=True)

for ki in range(k):
    cluster = commercial_loop.iloc[labels.query(f'label == {ki}').index][dl.
 ↪gas_cols]
    palette={i:"darkcyan" for i in range(cluster.shape[0])}

    ax0 = sns.lineplot(data=cluster.values.T, ax=axes[ki], palette=palette,␣
 ↪alpha=0.3)
    _ = [line.set_linestyle("-") for line in ax0.lines]

    ax0c = sns.lineplot(data=km_centroids[ki, :].T, ax=axes[ki], color="red")
    _ = [line.set_linestyle("-") for line in ax0c.lines]

    ax0c.legend(ax0c.lines[-1:], ["Centroide"]);
```

```
    ax0.set_title(f"Clúster {ki}")
    Media = km_centroids[ki, :].mean()
    ax0.text(0, Media, f"{Media = :0.2e} Therms")
    ax0.set(ylabel="Therms")

    ax0.set_xticks([e for e in range(0, 12)])
    ax0.set_xticklabels([f"{datetime.date(2008, i, 1).strftime('%B').upper()}"␣
 ↪for i in range(1, 13)], rotation=90)

path = images_path / Path("ensayos/
 ↪clustering_gas_por_comunidad_observando_patrones/comercial")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("cluster_grafica.svg"))
```


Clúster 0
Centroide
Therms
Media = 3.49e+04 Therms
Clúster 1
Centroide
Therms
Media = 5.50e+05 Therms
Clúster 2
Centroide
Therms
Media = 1.05e+05 Therms
Clúster 3
Centroide
Therms
Media = 6.94e+03 Therms
JANUARY
FEBRUARY
MARCH
APRIL
MAY
JUNE
JULY
AUGUST
SEPTEMBER
OCTOBER
NOVEMBER
DECEMBER

[89]:
```
v = tsne(commercial_loop[dl.gas_cols], 2)
for label in range(k):
    plt.scatter(
            x=v[(labels['label']==label).values, 0],
            y=v[(labels['label']==label).values, 1]
    )
plt.legend([f"Clúster {i}" for i in range(k)])

path = images_path / Path("ensayos/
 ↪clustering_gas_por_comunidad_observando_patrones/comercial")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("cluster_scatter.svg"))
```



[ ]:
```
comunities_by_cluster = {}

for nlab in range(k):
    com_index = np.where(labels==nlab)[0]
    com_list = []
    for c in commercial_loop.values[com_index][:,(0,1,3)]:
        com_list.append(c)
```

```
    comunities_by_cluster[nlab] = com_list

for k in sorted(comunities_by_cluster, key=lambda k:␣
↪len(comunities_by_cluster[k])):
    print(f"Al clúster {k} pertenecen las comunidades:")
    for e in comunities_by_cluster[k]:
        print("\t", e)
```

Industrial En este caso, la zona industrial con mayor consumo de gas es *Ashburn*

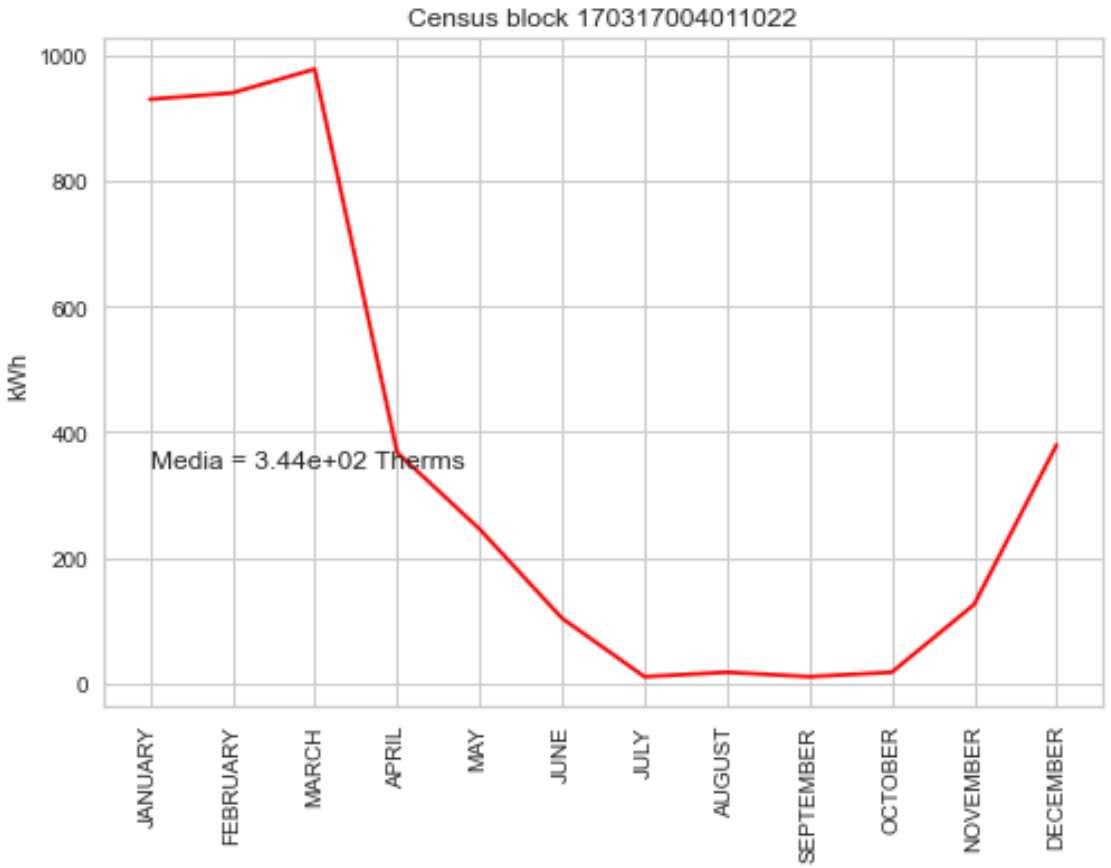
[91]:
```
industrial_loop = gas_df[(gas_df["COMMUNITY AREA NAME"] == "Ashburn") &␣
↪(gas_df["BUILDING TYPE"] == "Industrial")]
industrial_loop
```

[91]:
```
     COMMUNITY AREA NAME  CENSUS BLOCK BUILDING TYPE BUILDING_SUBTYPE  \
3589             Ashburn  1.703170e+14    Industrial       Industrial

      THERM JANUARY 2010  THERM FEBRUARY 2010  THERM MARCH 2010  \
3589               930.0                940.0             978.0

      THERM APRIL 2010  THERM MAY 2010  THERM JUNE 2010  THERM JULY 2010  \
3589             368.0           246.0            104.0             11.0

      THERM AUGUST 2010  THERM SEPTEMBER 2010  THERM OCTOBER 2010  \
3589               18.0                  11.0                18.0

      THERM NOVEMBER 2010  THERM DECEMBER 2010
3589                126.0                380.0
```

Al ser una única muestra, graficamos el resultado directamente.

[92]:
```
for i, (census_num, (index, row)) in enumerate(zip(industrial_loop['CENSUS␣
↪BLOCK'], industrial_loop[dl.gas_cols].iterrows())):
    ax0 = sns.lineplot(data=row.T, color="red")

    ax0.set_xticks([e for e in range(0, 12)])
    ax0.set_xticklabels([f"{datetime.date(2008, i, 1).strftime('%B').upper()}"␣
↪for i in range(1, 13)], rotation=90)

    ax0.set_title(f"Census block {int(census_num)}")
    Media = row.mean()
    ax0.text(0, Media, f"{Media = :0.2e} Therms")

    ax0.set(ylabel="kWh")

path = images_path / Path("ensayos/
↪clustering_gas_por_comunidad_observando_patrones/industrial")
```

```
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("cluster_grafica.svg"))
```



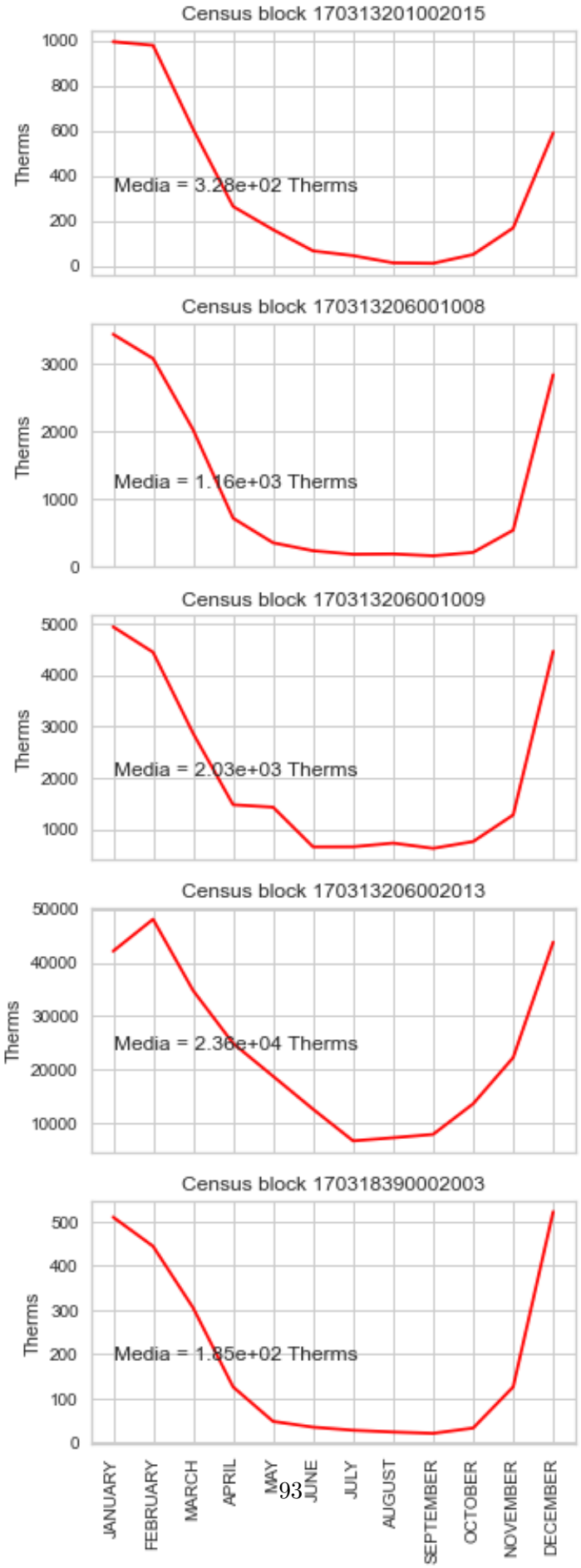
Residencial Entre las zonas residenciales elegimos a *Loop*.

[93]:
```
residential_loop = gas_df[(gas_df["COMMUNITY AREA NAME"] == "Loop") &␣
 ↪(gas_df["BUILDING TYPE"] == "Residential")]
residential_loop
```

[93]:
```
       COMMUNITY AREA NAME  CENSUS BLOCK BUILDING TYPE BUILDING_SUBTYPE  \
21904                 Loop  1.703132e+14   Residential        Multi < 7
38489                 Loop  1.703132e+14   Residential         Multi 7+
38491                 Loop  1.703132e+14   Residential         Multi 7+
38501                 Loop  1.703132e+14   Residential         Multi 7+
38513                 Loop  1.703184e+14   Residential    Single Family

       THERM JANUARY 2010  THERM FEBRUARY 2010  THERM MARCH 2010  \
21904               993.0                977.0             606.0
38489              3432.0               3072.0            2020.0
```

```
38491              4936.0               4440.0            2865.0
38501             42155.0              48132.0           34820.0
38513               510.0                444.0             305.0

       THERM APRIL 2010  THERM MAY 2010  THERM JUNE 2010  THERM JULY 2010  \
21904             263.0           160.0             66.0             45.0
38489             710.0           346.0            228.0            176.0
38491            1475.0          1427.0            655.0            657.0
38501           24927.0         18791.0          12621.0           6711.0
38513             126.0            48.0             35.0             28.0

       THERM AUGUST 2010  THERM SEPTEMBER 2010  THERM OCTOBER 2010  \
21904               13.0                  11.0                50.0
38489              181.0                 153.0               204.0
38491              729.0                 627.0               759.0
38501             7289.0                7924.0             13642.0
38513               24.0                  21.0                33.0

       THERM NOVEMBER 2010  THERM DECEMBER 2010
21904                168.0                589.0
38489                533.0               2834.0
38491               1276.0               4458.0
38501              22279.0              43853.0
38513                126.0                522.0
```

[94]:
```
f, axes = plt.subplots(5, 1, figsize=(5, 15), sharex=True)

for i, (census_num, (index, row)) in enumerate(zip(residential_loop['CENSUS␣
 ↪BLOCK'], residential_loop[dl.gas_cols].iterrows())):
    ax0 = sns.lineplot(data=row.T, ax=axes[i], color="red")

    ax0.set_xticks([e for e in range(0, 12)])
    ax0.set_xticklabels([f"{datetime.date(2008, i, 1).strftime('%B').upper()}"␣
 ↪for i in range(1, 13)], rotation=90)

    ax0.set_title(f"Census block {int(census_num)}")
    Media = row.mean()
    ax0.text(0, Media, f"{Media = :0.2e} Therms")

    ax0.set(ylabel="Therms")

path = images_path / Path("ensayos/
 ↪clustering_gas_por_comunidad_observando_patrones/residencial")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("cluster_grafica.svg"))
```


Census block 170313201002015
Therms
1000
800
600
400
200
0
Media = 3.28e+02 Therms
Census block 170313206001008
Therms
3000
2000
1000
0
Media = 1.16e+03 Therms
Census block 170313206001009
Therms
5000
4000
3000
2000
1000
Media = 2.03e+03 Therms
Census block 170313206002013
Therms
50000
40000
30000
20000
10000
Media = 2.36e+04 Therms
Census block 170318390002003
Therms
500
400
300
200
100
0
Media = 1.85e+02 Therms
JANUARY
FEBRUARY
MARCH
APRIL
MAY
JUNE
JULY
AUGUST
SEPTEMBER
OCTOBER
NOVEMBER
DECEMBER

1.3.8 Volver al inicio

1.3.9 Clustering de energia por media consumida en cada contador por comunidad

En este caso vamos a calcular el promedio consumido por cada contador en cada comunidad, dividiendo el consumo por la columna **ELECTRICITY ACCOUNTS**.

[95]:
```
total_energy_by_com = dl[["COMMUNITY AREA NAME"] + dl.energy_cols +␣
↪["ELECTRICITY ACCOUNTS"]]
total_energy_by_com
```

[95]:

	COMMUNITY AREA NAME	KWH JANUARY 2010	KWH FEBRUARY 2010	\
1	Ashburn	7334.0	7741.0	
5	Austin	0.0	0.0	
7	Austin	1470.0	1325.0	
8	Austin	2461.0	4888.0	
9	Austin	0.0	0.0	
…	…	…	…	
67046	Woodlawn	2705.0	1318.0	
67047	Woodlawn	1005.0	1760.0	
67048	Woodlawn	3567.0	3031.0	
67049	Woodlawn	1208.0	1055.0	
67050	Woodlawn	2717.0	3057.0	

	KWH MARCH 2010	KWH APRIL 2010	KWH MAY 2010	KWH JUNE 2010	\
1	4214.0	4284.0	2518.0	4273.0	
5	0.0	0.0	0.0	0.0	
7	294.0	391.0	366.0	2204.0	
8	2893.0	2737.0	2350.0	3037.0	
9	0.0	0.0	0.0	511.0	
…	…	…	…	…	
67046	1582.0	1465.0	1494.0	2990.0	
67047	1521.0	1832.0	2272.0	2361.0	
67048	2582.0	2295.0	7902.0	4987.0	
67049	1008.0	1109.0	1591.0	1367.0	
67050	2695.0	3793.0	4237.0	5383.0	

	KWH JULY 2010	KWH AUGUST 2010	KWH SEPTEMBER 2010	KWH OCTOBER 2010	\
1	4566.0	2787.0	3357.0	5540.0	
5	0.0	819.0	619.0	416.0	
7	2345.0	2032.0	920.0	586.0	
8	3874.0	4861.0	5180.0	2984.0	
9	904.0	1818.0	1968.0	738.0	

```
…              …              …                …                …
67046        2449.0         2351.0           1213.0           2174.0
67047        3018.0         3030.0           2886.0           3833.0
67048        5773.0         3996.0           3050.0           3103.0
67049        1569.0         1551.0           1376.0           1236.0
67050        5544.0         6929.0           5280.0           5971.0

       KWH NOVEMBER 2010  KWH DECEMBER 2010  ELECTRICITY ACCOUNTS
1                15774.0            19676.0                   8.0
5                  138.0                2.0                   3.0
7                  705.0              785.0                   3.0
8                 2635.0             3597.0                   6.0
9                  450.0             2207.0                   3.0
…                    …                  …                     …
67046             2888.0             5025.0                   6.0
67047             6290.0            12169.0                   9.0
67048             3880.0             4684.0                   7.0
67049             2108.0             2529.0                   7.0
67050             6986.0             5144.0                  12.0

[66180 rows x 14 columns]
```

Dividimos el consumo mensual de energía por el número de contadores.

[96]:
```
total_energy_by_com[dl.energy_cols] = total_energy_by_com[dl.energy_cols].
 ↪div(total_energy_by_com["ELECTRICITY ACCOUNTS"], axis=0)
del total_energy_by_com["ELECTRICITY ACCOUNTS"]
total_energy_by_com
```

[96]:
```
      COMMUNITY AREA NAME  KWH JANUARY 2010  KWH FEBRUARY 2010  \
1                 Ashburn        916.750000         967.625000
5                  Austin          0.000000           0.000000
7                  Austin        490.000000         441.666667
8                  Austin        410.166667         814.666667
9                  Austin          0.000000           0.000000
…                     …               …                  …
67046            Woodlawn        450.833333         219.666667
67047            Woodlawn        111.666667         195.555556
67048            Woodlawn        509.571429         433.000000
67049            Woodlawn        172.571429         150.714286
67050            Woodlawn        226.416667         254.750000

       KWH MARCH 2010  KWH APRIL 2010  KWH MAY 2010  KWH JUNE 2010  \
1          526.750000      535.500000    314.750000     534.125000
5            0.000000        0.000000      0.000000       0.000000
7           98.000000      130.333333    122.000000     734.666667
8          482.166667      456.166667    391.666667     506.166667
9            0.000000        0.000000      0.000000     170.333333
```

```
…              …             …             …             …
67046     263.666667    244.166667    249.000000    498.333333
67047     169.000000    203.555556    252.444444    262.333333
67048     368.857143    327.857143   1128.857143    712.428571
67049     144.000000    158.428571    227.285714    195.285714
67050     224.583333    316.083333    353.083333    448.583333

       KWH JULY 2010  KWH AUGUST 2010  KWH SEPTEMBER 2010  KWH OCTOBER 2010  \
1          570.750000       348.375000          419.625000        692.500000
5            0.000000       273.000000          206.333333        138.666667
7          781.666667       677.333333          306.666667        195.333333
8          645.666667       810.166667          863.333333        497.333333
9          301.333333       606.000000          656.000000        246.000000
…                 …              …                   …                 …
67046      408.166667       391.833333          202.166667        362.333333
67047      335.333333       336.666667          320.666667        425.888889
67048      824.714286       570.857143          435.714286        443.285714
67049      224.142857       221.571429          196.571429        176.571429
67050      462.000000       577.416667          440.000000        497.583333

       KWH NOVEMBER 2010  KWH DECEMBER 2010
1            1971.750000        2459.500000
5              46.000000           0.666667
7             235.000000         261.666667
8             439.166667         599.500000
9             150.000000         735.666667
…                    …                  …
67046         481.333333         837.500000
67047         698.888889        1352.111111
67048         554.285714         669.142857
67049         301.142857         361.285714
67050         582.166667         428.666667

[66180 rows x 13 columns]
```

Y posteriormente calculamos la media para cada comunidad.

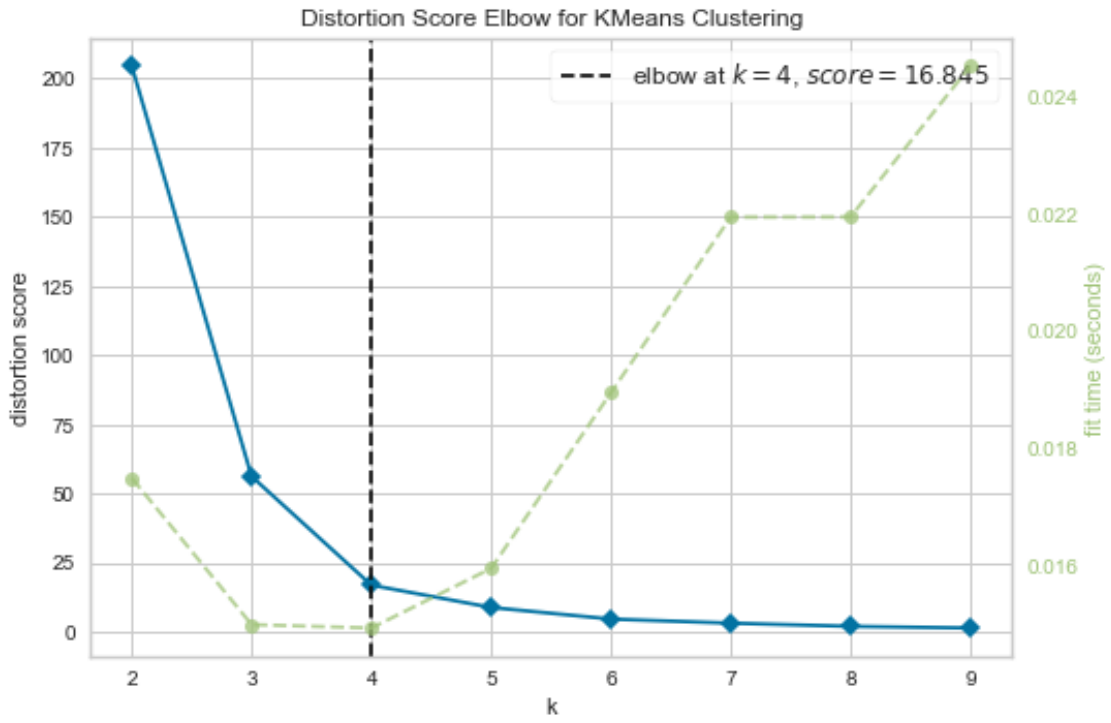
[97]:
```
mean_energy = total_energy_by_com.groupby('COMMUNITY AREA NAME',␣
↪as_index=False).mean()
mean_energy
```

[97]:
```
   COMMUNITY AREA NAME  KWH JANUARY 2010  KWH FEBRUARY 2010  KWH MARCH 2010  \
0          Albany Park        588.960230         620.076751      599.319381
1       Archer Heights       4060.960537        4162.342241     3963.701645
2        Armour Square       1742.539516        1624.745099     1512.367518
3              Ashburn        780.696213         745.711491      704.729112
4       Auburn Gresham        586.611070         590.521216      581.013643
..                 …                 …                  …               …
```

```
72           West Lawn       1599.801869        1761.060292     1676.913074
73        West Pullman        336.840340         505.430414      494.363589
74          West Ridge        631.225589         633.842594      606.552057
75           West Town        554.456614         601.668226      571.151123
76            Woodlawn       7110.092941        6651.934074     4857.665927

    KWH APRIL 2010  KWH MAY 2010  KWH JUNE 2010  KWH JULY 2010  \
0        608.100158    787.060110    1062.593329    1124.960334
1       3882.010044   4056.491226    4694.134467    4813.099926
2       1574.774637   1571.659308    1762.314045    2198.402000
3        717.169818    960.698133    1346.862270    1441.881978
4        573.039983    706.920065     921.194964     971.684396
..              …             …              …              …
72      1675.510384   2034.169928    2506.009592    2512.312836
73       517.450046    575.883072     692.737455     884.599784
74       679.627136    901.395931    1193.336081    1153.017542
75       570.600857    693.789572     896.494764    1049.318744
76      4664.439473   4299.869910    5068.639403    4576.080946

    KWH AUGUST 2010  KWH SEPTEMBER 2010  KWH OCTOBER 2010  KWH NOVEMBER 2010  \
0         902.121725          681.448159        682.711545         910.421071
1        4811.352907         4152.028865       3892.331308        4081.929164
2        2139.742148         1774.726173       1223.895629        1602.952279
3        1121.033684          808.984312        775.900670        1065.874121
4         820.333114          636.278627        667.405742         888.332942
..               …                   …                 …                  …
72       2422.593366         1906.256466       1680.336738        1806.449746
73        846.449675          645.562179        586.981731         700.753697
74        936.103499          684.780656        712.970023         945.586482
75       1041.675432          811.528727        688.255666         781.993540
76       5510.521041         6255.137064       5391.800905        5675.902200

    KWH DECEMBER 2010
0           975.996995
1          4534.551003
2          2310.849074
3          1191.412444
4          1013.297035
..                 …
72         2138.588987
73          921.851030
74          948.583799
75          956.159867
76         8230.214213

[77 rows x 13 columns]
```

Aplicamos clustering con KMeans

[98]:
```
kmax = 10
k_values = (2, kmax)

scaler = preprocessing.StandardScaler()
data = mean_energy[dl.energy_cols]
norm_data = scaler.fit_transform(data)

km = KMeansCluster(norm_data)
km.cluster(k_values)

path = images_path / Path("ensayos/clustering_energia_media_en_contador/total")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("elbow.svg"))
```



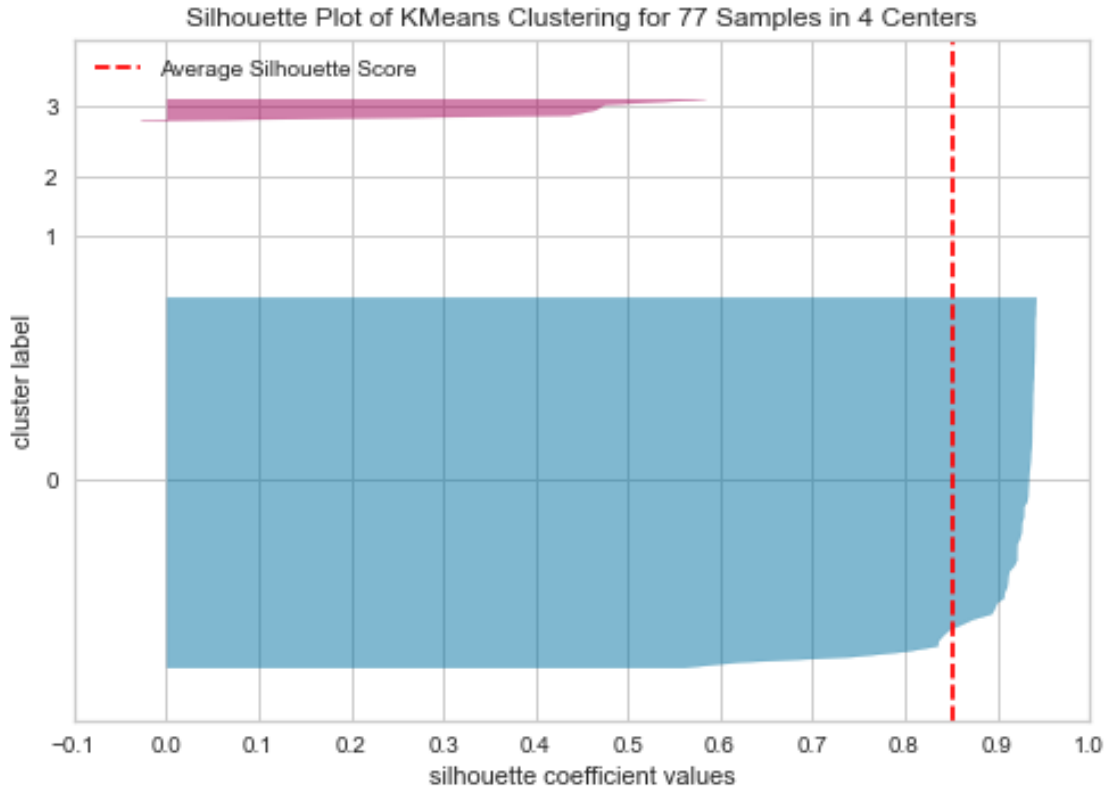

```
<Figure size 576x396 with 0 Axes>
```

Con k=4 tenemos la división óptima.

[99]:
```
km_best = km.get_kmeans_of(4)
km_labels = km_best.labels_
km_centroids = scaler.inverse_transform(km_best.cluster_centers_)

km_labels
```

[99]:
```
array([0, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0,
       0, 0, 0, 3, 3, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3])
```

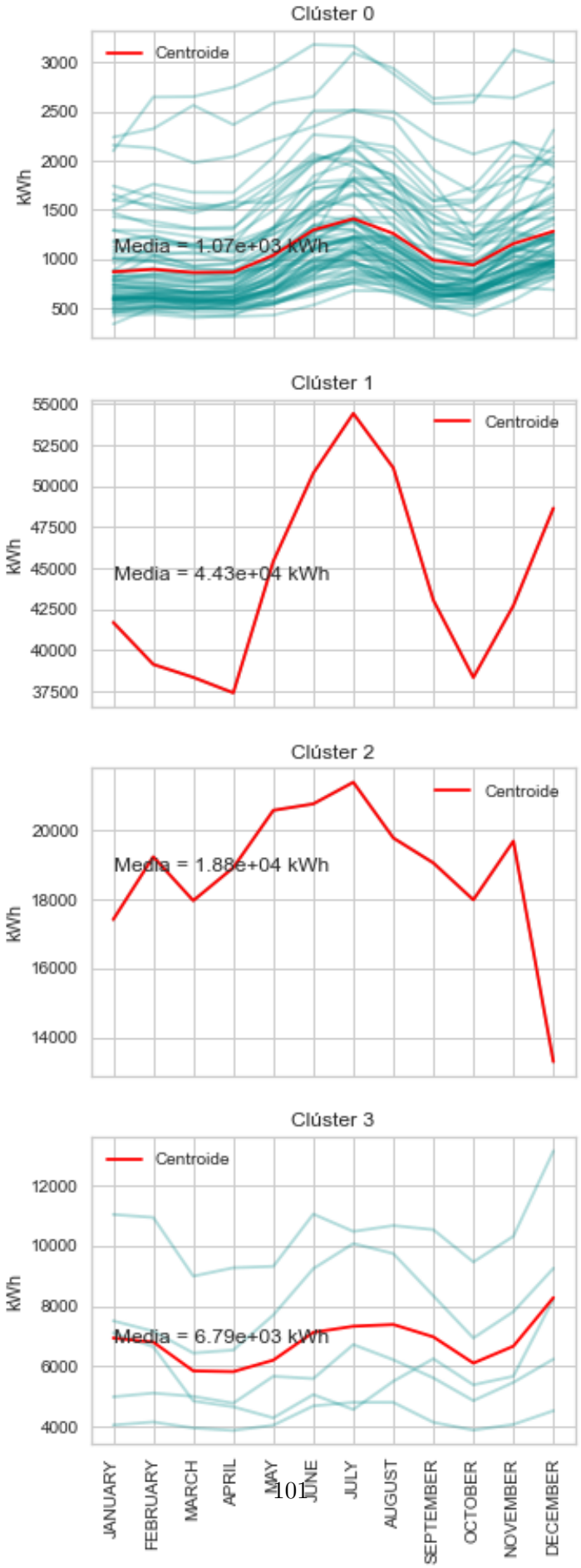
[100]:
```
_ = silhouette_visualizer(km_best, norm_data, colors='yellowbrick')

path = images_path / Path("ensayos/clustering_energia_media_en_contador/total")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("silhouette.svg"))
```




```
<Figure size 576x396 with 0 Axes>
```

[101]:
```
labels = pd.DataFrame(km_labels, columns=["label"])

k = km_centroids.shape[0]

f, axes = plt.subplots(k, 1, figsize=(5, 15), sharex=True)

for ki in range(k):
    cluster = mean_energy.iloc[labels.query(f'label == {ki}').index][dl.
 ↪energy_cols]
```

```
    palette={i:"darkcyan" for i in range(cluster.shape[0])}

    ax0 = sns.lineplot(data=cluster.values.T, ax=axes[ki], palette=palette,␣
↪alpha=0.3)
    _ = [line.set_linestyle("-") for line in ax0.lines]

    ax0c = sns.lineplot(data=km_centroids[ki, :].T, ax=axes[ki], color="red")
    _ = [line.set_linestyle("-") for line in ax0c.lines]

    ax0c.legend(ax0c.lines[-1:], ["Centroide"]);

    ax0.set_title(f"Clúster {ki}")
    Media = km_centroids[ki, :].mean()
    ax0.text(0, Media, f"{Media = :0.2e} kWh")
    ax0.set(ylabel="kWh")

    ax0.set_xticks([e for e in range(0, 12)])
    ax0.set_xticklabels([f"{datetime.date(2008, i, 1).strftime('%B').upper()}"␣
↪for i in range(1, 13)], rotation=90)

path = images_path / Path("ensayos/clustering_energia_media_en_contador/total")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("grafica.svg"))
```


Clúster 0
Centroide
kWh
3000
2500
2000
1500
1000
500
Media = 1.07e+03 kWh
Clúster 1
Centroide
kWh
55000
52500
50000
47500
45000
42500
40000
37500
Media = 4.43e+04 kWh
Clúster 2
Centroide
kWh
20000
18000
16000
14000
Media = 1.88e+04 kWh
Clúster 3
Centroide
kWh
12000
10000
8000
6000
4000
Media = 6.79e+03 kWh
JANUARY
FEBRUARY
MARCH
APRIL
MAY
JUNE
JULY
AUGUST
SEPTEMBER
OCTOBER
NOVEMBER
DECEMBER

¿Qué comunidad hay en cada clúster?

[ ]:
```
comunities_by_cluster = {}

for nlab in range(k):
    com_index = np.where(labels==nlab)[0]
    com_list = []
    for c in mean_energy.values[com_index][:,0]:
        com_list.append(c)
    comunities_by_cluster[nlab] = com_list

for k in sorted(comunities_by_cluster, key=lambda k:␣
↪len(comunities_by_cluster[k])):
    print(f"Al clúster {k} pertenecen las comunidades:")
    for e in comunities_by_cluster[k]:
        print("\t", e)
```

Separación por cuartiles Vamos a separar las muestras muy alejadas del valor medio mediante la eliminación de outliers. Dividiremos en 3 porciones: los que están muy por abajo de la media, en ***mean_q1***; los que están muy por encima de la media, en ***mean_q4***; y los que se encuentran sobre la media, en ***mean_qm***. Con este tratamiento previo trataremos de encontrar de mejor modo patrones dentro de cada grupo.

[103]:
```
zmin_score = 0.26

z_mean_energy = zscore(mean_energy[dl.energy_cols])
mean_energy_qm = mean_energy[np.logical_and(-zmin_score <= z_mean_energy,␣
↪z_mean_energy <= zmin_score).all(axis=1)]
mean_energy_q1 = mean_energy[(-zmin_score > z_mean_energy).all(axis=1)]
mean_energy_q4 = mean_energy[(zmin_score < z_mean_energy).all(axis=1)]

mean_energy_qm
```

[103]:

	COMMUNITY AREA NAME	KWH JANUARY 2010	KWH FEBRUARY 2010	KWH MARCH 2010	\
2	Armour Square	1742.539516	1624.745099	1512.367518	
3	Ashburn	780.696213	745.711491	704.729112	
5	Austin	881.393662	1143.496215	1142.900845	
7	Avondale	825.904543	831.643551	816.281594	
8	Belmont Cragin	1185.054781	1236.872753	1123.117644	
9	Beverly	819.153565	843.576326	780.968340	
11	Brighton Park	962.527296	975.527314	893.291030	
12	Burnside	1659.831583	1511.910669	1522.988559	
16	Clearing	1006.018576	971.486217	908.137255	
17	Douglas	804.330181	1028.136170	1072.324103	
21	Edgewater	787.956060	765.620076	757.785832	

24	Forest Glen	998.415636	1029.897147	930.546497
27	Garfield Ridge	1124.419351	1047.446808	925.297925
31	Hermosa	704.177132	718.416720	727.294078
32	Humboldt Park	880.847058	904.685828	855.461803
33	Hyde Park	860.503676	891.304441	824.430828
34	Irving Park	787.311409	804.157295	760.386940
37	Lakeview	1010.291796	989.955995	986.119385
38	Lincoln Park	1430.381825	1381.343191	1312.933273
39	Lincoln Square	1289.470311	1197.343900	1154.690204
42	Lower West Side	2238.734970	2324.962051	2563.762259
43	McKinley Park	1290.580346	1284.954295	1226.908180
46	Mount Greenwood	1166.047926	1227.524962	1103.540173
50	New City	2158.034873	2128.007395	1979.528847
52	North Lawndale	769.911303	848.381662	857.489972
53	North Park	1597.105270	1550.847146	1466.172978
55	O'Hare	1471.117048	1338.369856	1301.571878
57	Portage Park	743.415492	754.489564	715.730941
58	Pullman	2100.235399	2647.225804	2651.496623
63	South Deering	1497.917369	1666.825278	1562.935376
64	South Lawndale	1083.949928	1159.269373	1137.217089
66	Uptown	822.961222	812.293937	776.984800
69	West Elsdon	1050.124284	1075.118131	1051.426395
72	West Lawn	1599.801869	1761.060292	1676.913074

	KWH APRIL 2010	KWH MAY 2010	KWH JUNE 2010	KWH JULY 2010	\
2	1574.774637	1571.659308	1762.314045	2198.402000	
3	717.169818	960.698133	1346.862270	1441.881978	
5	1061.795445	1256.132520	1484.143676	1639.071569	
7	789.980707	950.883245	1282.302588	1387.953706	
8	1101.673213	1304.915316	1531.453974	1669.345291	
9	847.318220	1133.758536	1584.180429	1546.303610	
11	889.663229	1002.113841	1197.505350	1419.792567	
12	1500.095776	1773.631201	2003.929350	2109.256792	
16	881.398771	1136.920999	1462.454409	1656.843804	
17	965.833775	1068.902056	1234.492284	1899.938112	
21	808.236524	1013.694592	1337.047880	1292.964239	
24	958.553462	1314.293749	1723.190553	1744.001857	
27	946.223852	1165.278231	1545.691703	1802.146002	
31	692.283694	800.147358	1129.596609	1218.280040	
32	815.365556	947.857923	1097.151848	1229.160301	
33	858.400679	941.763228	1181.456530	1410.719011	
34	767.441701	958.748132	1292.566553	1361.615334	
37	1007.950787	1289.291345	1712.388964	1787.969638	
38	1323.007504	1620.290012	1988.384519	2155.631354	
39	1226.145015	1493.415860	1777.453604	1811.905439	
42	2365.772965	2584.996838	2652.793231	3096.807976	
43	1166.402172	1315.028792	1508.648989	1830.942342	

46	1150.895012	1594.086521	1975.182248	2139.973906
50	2041.684600	2209.034956	2346.285104	2517.115221
52	859.260080	920.269797	972.349859	1220.240197
53	1585.892708	1825.844100	2264.106142	2234.551794
55	1302.013968	1724.254600	2069.873388	1940.255138
57	732.006116	945.573452	1270.487066	1380.737304
58	2748.754931	2934.033984	3181.305584	3166.381242
63	1525.794170	1663.397625	2033.259323	2009.443261
64	1135.061677	1246.770658	1424.521890	1813.676266
66	795.939722	1045.913039	1334.576922	1396.770914
69	1081.294813	1321.606673	1844.828496	1990.117801
72	1675.510384	2034.169928	2506.009592	2512.312836

	KWH AUGUST 2010	KWH SEPTEMBER 2010	KWH OCTOBER 2010	KWH NOVEMBER 2010	\
2	2139.742148	1774.726173	1223.895629	1602.952279	
3	1121.033684	808.984312	775.900670	1065.874121	
5	1672.210766	1369.155481	1197.265957	1405.209377	
7	1167.133834	916.402788	865.263593	1107.338454	
8	1593.902853	1244.509159	1140.136696	1415.916247	
9	1147.048840	919.139609	861.511896	1283.501531	
11	1421.544570	1102.163523	948.540256	1092.090235	
12	1780.390617	1527.236072	1577.450641	1935.369574	
16	1493.196089	1065.599291	960.189565	1175.699334	
17	1635.922526	1441.719261	1134.206334	1304.732489	
21	1106.686453	898.508448	926.612126	1261.304178	
24	1327.503928	987.360230	1031.827405	1403.344445	
27	1671.712309	1116.258401	984.279463	1191.441888	
31	1131.554421	841.572864	779.246675	928.565326	
32	1228.869095	933.532279	871.939696	1056.782991	
33	1354.890042	957.197711	888.457646	1082.287230	
34	1182.035092	891.198331	852.952136	1081.173961	
37	1551.799873	1166.202127	1086.955514	1294.617318	
38	2079.804864	1650.778356	1474.125476	1718.213343	
39	1484.892539	1199.797229	1202.759471	1566.712233	
42	2938.655204	2632.480765	2665.254262	2639.321548	
43	1750.085608	1496.456765	1330.582396	1418.612205	
46	1763.025987	1267.471487	1239.876325	1451.904435	
50	2498.114585	2225.133858	2066.290594	2194.410749	
52	1214.814409	1094.312193	932.652531	1120.387548	
53	1983.246059	1590.103452	1602.898242	2052.842765	
55	1641.615568	1356.539485	1371.032938	1845.774332	
57	1095.934655	796.496626	792.842362	1018.084277	
58	2884.105464	2583.234329	2593.965732	3128.337629	
63	1851.622628	1590.819372	1734.424423	2186.510849	
64	1823.113284	1440.972191	1247.026243	1321.758601	
66	1124.679495	853.774196	839.126860	1085.585506	
69	1849.691242	1267.647503	1130.864014	1210.435294	

```
72      2422.593366         1906.256466       1680.336738        1806.449746

    KWH DECEMBER 2010
2          2310.849074
3          1191.412444
5          1630.487812
7          1239.955574
8          1619.806319
9          1187.529238
11         1247.987735
12         1969.445154
16         1349.713413
17         1830.144795
21         1167.221496
24         1414.124895
27         1421.864072
31         1153.127346
32         1275.612207
33         1202.991025
34         1229.086884
37         1535.295256
38         2043.712750
39         1739.853156
42         2796.924253
43         1661.874531
46         1575.857425
50         1934.417876
52         1251.061516
53         1996.183347
55         1758.726383
57         1128.016885
58         3009.082393
63         2079.765557
64         1485.961881
66         1211.331702
69         1396.508081
72         2138.588987
```

[104]:
```
mean_energy_q1
```

[104]:
```
        COMMUNITY AREA NAME  KWH JANUARY 2010  KWH FEBRUARY 2010  \
14                  Chatham        548.736974         563.949036
19       East Garfield Park        450.959905         518.789042
20                East Side        418.123974         438.244319
23                Englewood        504.576513         525.370981
25              Fuller Park        485.376146         507.384843
28         Grand Boulevard        612.172127         602.778075
```

```
29  Greater Grand Crossing        540.691549         563.919851
56                 Oakland        462.199800         473.069618
62           South Chicago        447.606205         466.036084
65             South Shore        486.540287         498.248891
70          West Englewood        618.129398         594.753803
71      West Garfield Park        518.252198         598.153529
73            West Pullman        336.840340         505.430414

    KWH MARCH 2010  KWH APRIL 2010  KWH MAY 2010  KWH JUNE 2010  \
14      546.049245      538.207488    668.152943     878.640122
19      528.272949      529.217821    573.993923     679.083028
20      401.374926      423.017437    617.825313     897.358812
23      499.136593      480.369505    547.117476     675.751533
25      517.840736      497.809375    550.832397     628.657527
28      509.061568      493.853598    547.317564     662.937125
29      538.917779      543.887588    634.444308     820.860347
56      423.576991      413.032848    426.406716     530.717199
62      462.377413      444.617988    530.067131     693.512976
65      488.881648      486.547177    542.520038     726.045306
70      563.928640      544.140699    626.366991     785.657575
71      600.395014      585.378133    668.000861     797.586222
73      494.363589      517.450046    575.883072     692.737455

    KWH JULY 2010  KWH AUGUST 2010  KWH SEPTEMBER 2010  KWH OCTOBER 2010  \
14     938.656570       761.947701          605.753641        633.530900
19     825.308169       823.947782          735.957543        613.447287
20     937.610734       685.952468          495.418421        505.783496
23     757.998640       730.072981          565.111700        590.747475
25     769.049517       776.020786          585.774026        542.345548
28     795.866038       809.540480          638.899086        582.338788
29     869.311150       814.108248          634.695363        645.272314
56     676.657687       671.607153          533.147311        420.315195
62     748.631958       649.540879          510.929136        546.391435
65     796.639112       713.272992          561.745431        573.613358
70     848.069335       796.693279          618.995212        650.060245
71     936.586922       904.084362          746.388751        636.364346
73     884.599784       846.449675          645.562179        586.981731

    KWH NOVEMBER 2010  KWH DECEMBER 2010
14         836.728679         922.702208
19         729.301227         905.244509
20         712.306064         686.252539
23         785.415277         941.913156
25         686.766241         819.261537
28         723.965683         981.641705
29         827.015719         982.295835
56         576.774318         801.655847
```

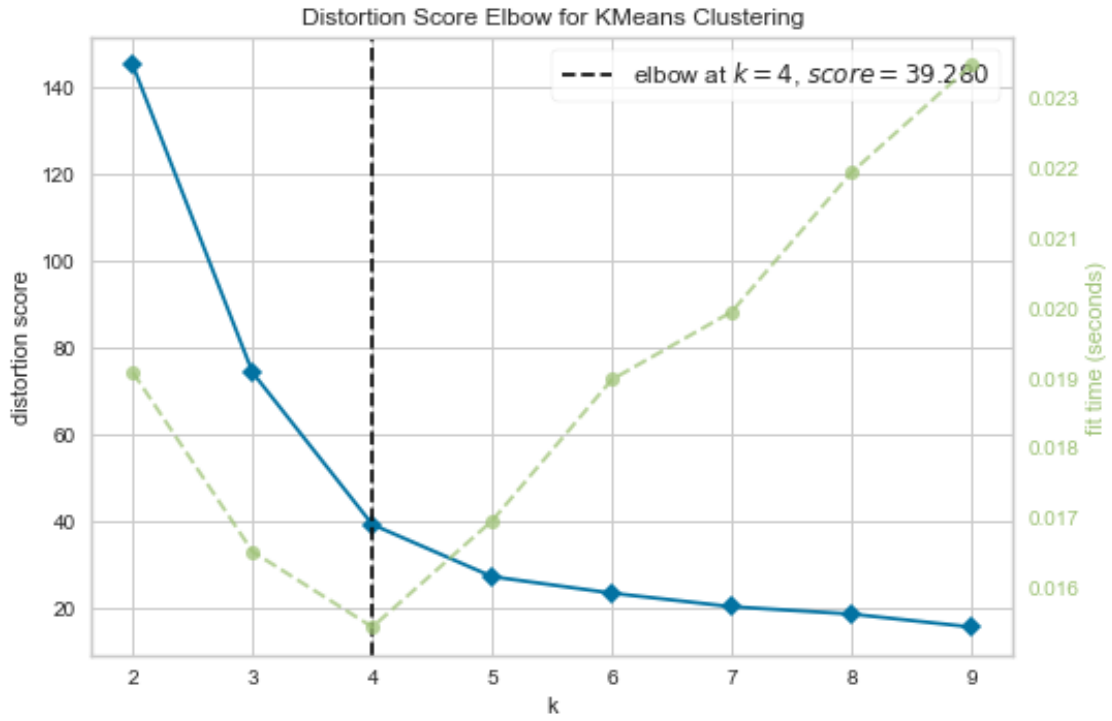
```
62         724.367365         856.924134
65         729.811821         876.291037
70         825.636205         964.519868
71         794.316026         985.578056
73         700.753697         921.851030
```

[105]:
```
mean_energy_q4
```

[105]:
```
    COMMUNITY AREA NAME  KWH JANUARY 2010  KWH FEBRUARY 2010  KWH MARCH 2010  \
1         Archer Heights       4060.960537        4162.342241     3963.701645
41                  Loop      41703.327760       39149.935395    38359.267550
47      Near North Side       7508.711888        7166.403554     6443.371826
48      Near South Side       4995.516429        5119.402869     5011.130793
49       Near West Side      11028.762534       10930.151571     8987.540767
59            Riverdale      17407.626092       19220.115392    17953.735688
76             Woodlawn       7110.092941        6651.934074     4857.665927

     KWH APRIL 2010  KWH MAY 2010  KWH JUNE 2010  KWH JULY 2010  \
1       3882.010044   4056.491226    4694.134467    4813.099926
41     37418.408781  45420.173772   50777.961084   54413.264670
47      6542.160299   7701.709251    9235.782100   10067.107065
48      4784.420322   5674.484652    5597.203584    6722.953663
49      9264.634207   9310.610341   11041.463763   10466.831510
59     18906.783214  20560.624665   20747.169334   21371.818762
76      4664.439473   4299.869910    5068.639403    4576.080946

     KWH AUGUST 2010  KWH SEPTEMBER 2010  KWH OCTOBER 2010  KWH NOVEMBER 2010  \
1        4811.352907         4152.028865       3892.331308        4081.929164
41      51098.644304        43053.542073      38350.661591       42724.906583
47       9733.517137         8336.834124       6941.555301        7812.584585
48       6220.897687         5621.622801       4868.820165        5472.779093
49      10660.692877        10525.429915       9457.175345       10307.927959
59      19762.272862        19041.743449      17976.516134       19671.468081
76       5510.521041         6255.137064       5391.800905        5675.902200

     KWH DECEMBER 2010
1          4534.551003
41        48648.280433
47         9245.165316
48         6241.668529
49        13126.378343
59        13303.468699
76         8230.214213
```

Aplicamos clustering a los valores en ***mean_qm***.

[106]:
```
kmax = 10
k_values = (2, kmax)

scaler = preprocessing.StandardScaler()
data = mean_energy_qm[dl.energy_cols]
norm_data = scaler.fit_transform(data)

km = KMeansCluster(norm_data)
km.cluster(k_values)

path = images_path / Path("ensayos/clustering_energia_media_en_contador/
 ↪mean_qm")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("elbow.svg"))
```

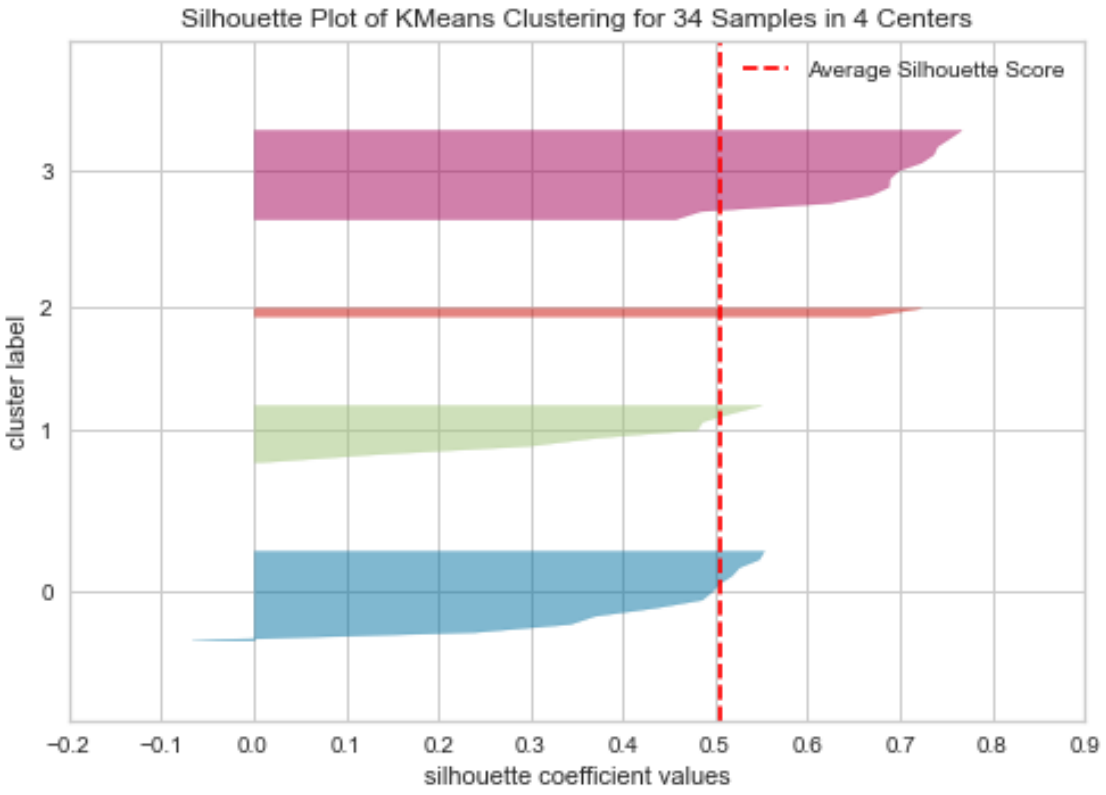



```
<Figure size 576x396 with 0 Axes>
```

[107]:
```
km_best = km.get_kmeans_of(4)
km_labels = km_best.labels_
km_centroids = scaler.inverse_transform(km_best.cluster_centers_)

km_labels
```

[107]:
```
array([1, 3, 0, 3, 0, 3, 3, 1, 0, 0, 3, 0, 0, 3, 3, 3, 3, 0, 1, 0, 2, 0,
       0, 1, 3, 1, 1, 3, 2, 1, 0, 3, 0, 1])
```

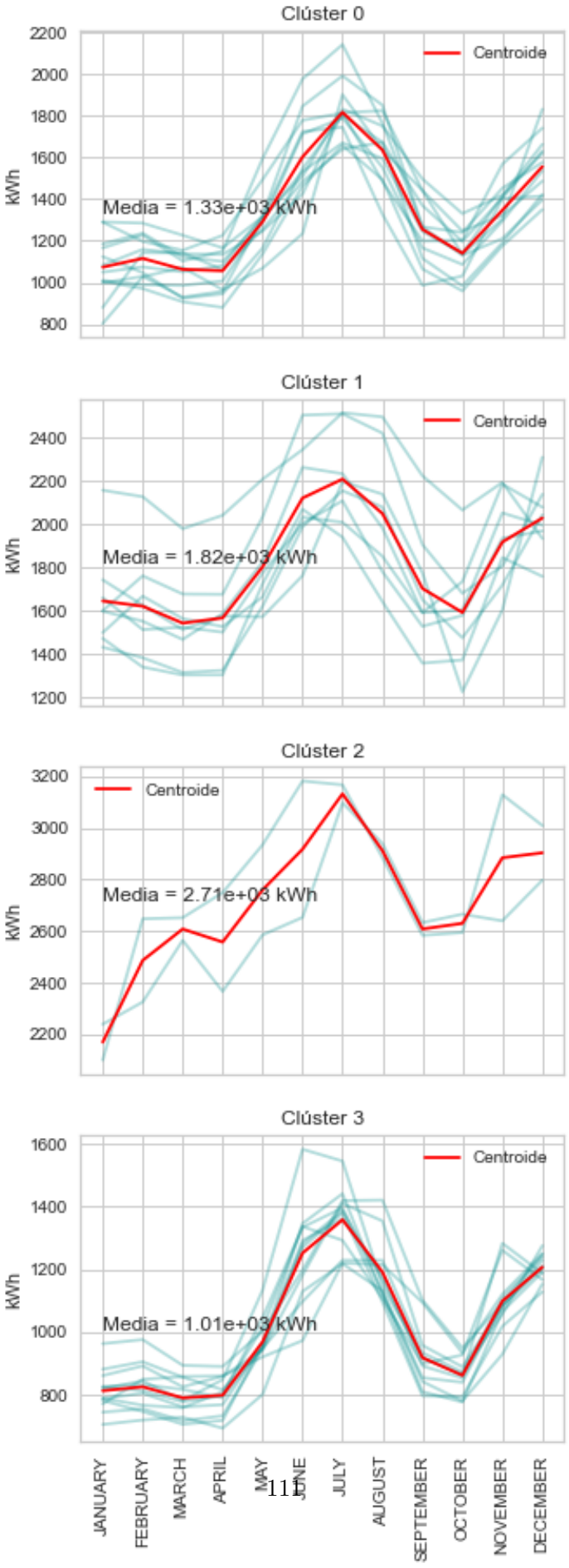
[108]:
```
_ = silhouette_visualizer(km_best, norm_data, colors='yellowbrick')

path = images_path / Path("ensayos/clustering_energia_media_en_contador/
 ↪mean_qm")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("silhouette.svg"))
```



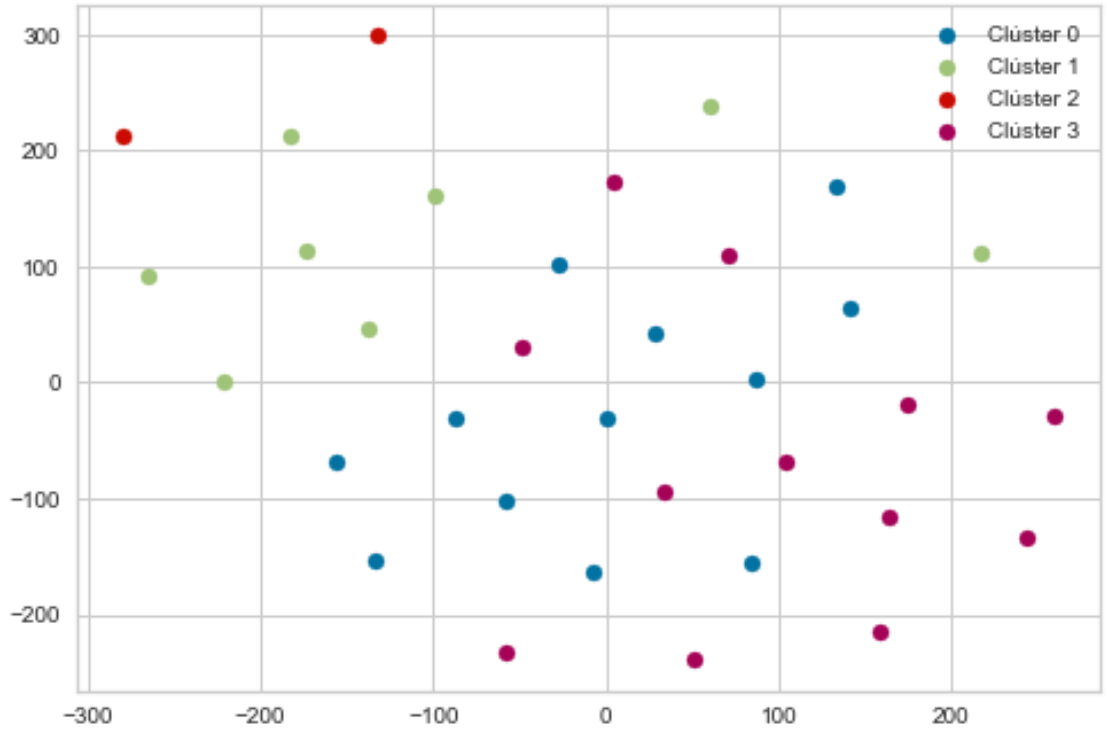
<Figure size 576x396 with 0 Axes>

[109]:
```
labels = pd.DataFrame(km_labels, columns=["label"])

k = km_centroids.shape[0]

f, axes = plt.subplots(k, 1, figsize=(5, 15), sharex=True)

for ki in range(k):
    cluster = mean_energy_qm.iloc[labels.query(f'label == {ki}').index][dl.
 ↪energy_cols]
    palette={i:"darkcyan" for i in range(cluster.shape[0])}
```

```
    ax0 = sns.lineplot(data=cluster.values.T, ax=axes[ki], palette=palette,␣
 ↪alpha=0.3)
    _ = [line.set_linestyle("-") for line in ax0.lines]

    ax0c = sns.lineplot(data=km_centroids[ki, :].T, ax=axes[ki], color="red")
    _ = [line.set_linestyle("-") for line in ax0c.lines]

    ax0c.legend(ax0c.lines[-1:], ["Centroide"]);

    ax0.set_title(f"Clúster {ki}")
    Media = km_centroids[ki, :].mean()
    ax0.text(0, Media, f"{Media = :0.2e} kWh")
    ax0.set(ylabel="kWh")

    ax0.set_xticks([e for e in range(0, 12)])
    ax0.set_xticklabels([f"{datetime.date(2008, i, 1).strftime('%B').upper()}"␣
 ↪for i in range(1, 13)], rotation=90)

path = images_path / Path("ensayos/clustering_energia_media_en_contador/
 ↪mean_qm")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("cluster_grafica.svg"))
```


Clúster 0
Centroide
Media = 1.33e+03 kWh
kWh
2200
2000
1800
1600
1400
1200
1000
800
Clúster 1
Centroide
Media = 1.82e+03 kWh
kWh
2400
2200
2000
1800
1600
1400
1200
Clúster 2
Centroide
Media = 2.71e+03 kWh
kWh
3200
3000
2800
2600
2400
2200
Clúster 3
Centroide
Media = 1.01e+03 kWh
kWh
1600
1400
1200
1000
800
JANUARY
FEBRUARY
MARCH
APRIL
MAY
JUNE
JULY
AUGUST
SEPTEMBER
OCTOBER
NOVEMBER
DECEMBER

[110]:
```
v = tsne(mean_energy_qm[dl.energy_cols], 2)
for label in range(k):
    plt.scatter(
            x=v[(labels['label']==label).values, 0],
            y=v[(labels['label']==label).values, 1]
    )
plt.legend([f"Clúster {i}" for i in range(k)])

path = images_path / Path("ensayos/clustering_energia_media_en_contador/
 ↪mean_qm")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("scatter.svg"))
```



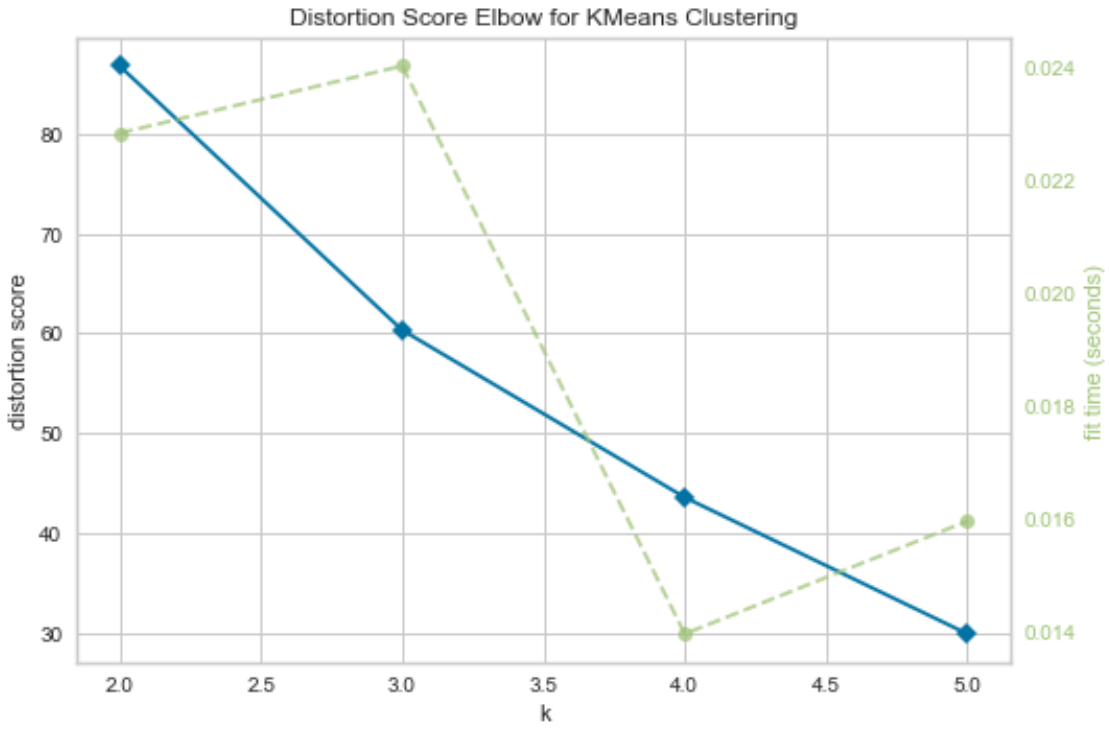
[ ]:
```
comunities_by_cluster = {}

for nlab in range(k):
    com_index = np.where(labels==nlab)[0]
    com_list = []
    for c in mean_energy_qm.values[com_index][:,0]:
        com_list.append(c)
```

```
    comunities_by_cluster[nlab] = com_list

for k in sorted(comunities_by_cluster, key=lambda k:␣
 ↪len(comunities_by_cluster[k])):
    print(f"Al clúster {k} pertenecen las comunidades:")
    for e in comunities_by_cluster[k]:
        print("\t", e)
```

Aplicamos clustering a los valores en ***mean_q1***.

[112]:
```
kmax = 6
k_values = (2, kmax)

scaler = preprocessing.StandardScaler()
data = mean_energy_q1[dl.energy_cols]
norm_data = scaler.fit_transform(data)

km = KMeansCluster(norm_data)
km.cluster(k_values)

path = images_path / Path("ensayos/clustering_energia_media_en_contador/
 ↪mean_q1")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("elbow.svg"))
```

```
C:\Users\Alberto\miniconda3\envs\cdi-musiani-env\lib\site-
packages\yellowbrick\utils\kneed.py:140: YellowbrickWarning:

No 'knee' or 'elbow point' detected This could be due to bad clustering, no
actual clusters being formed etc.

C:\Users\Alberto\miniconda3\envs\cdi-musiani-env\lib\site-
packages\yellowbrick\cluster\elbow.py:343: YellowbrickWarning:

No 'knee' or 'elbow' point detected, pass `locate_elbow=False` to remove the
warning
```

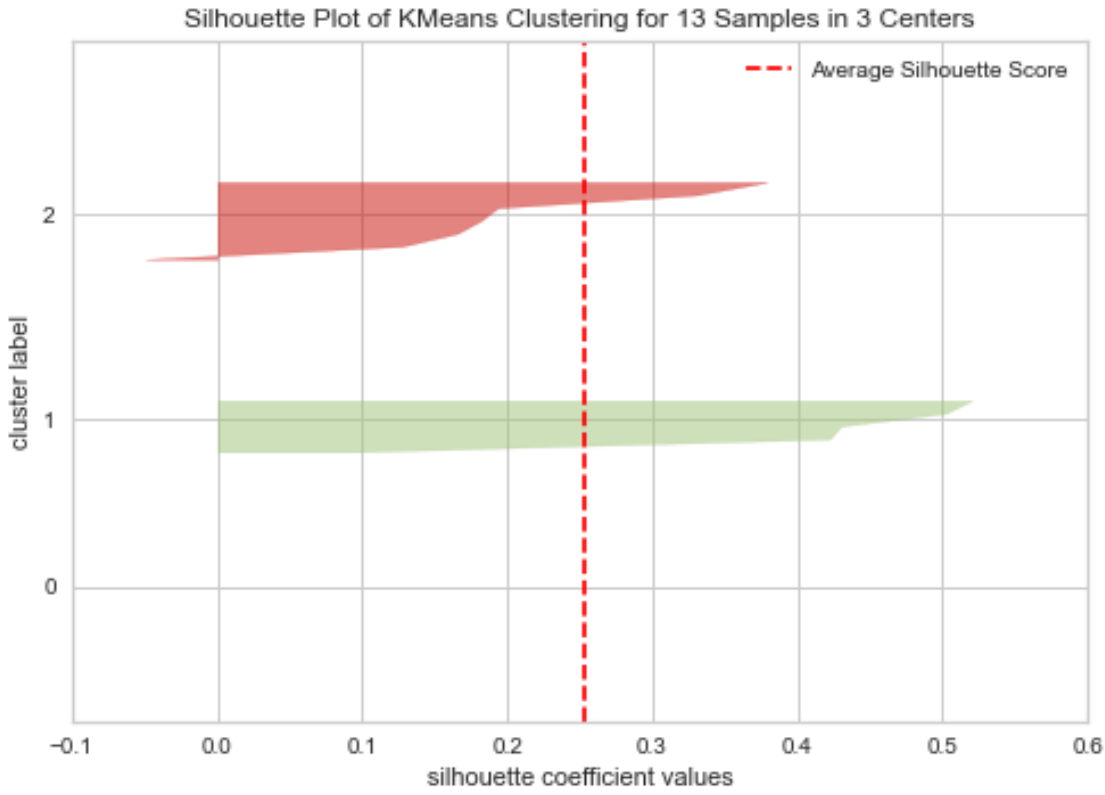


<Figure size 576x396 with 0 Axes>

[113]:
```
km_best = km.get_kmeans_of(3)
km_labels = km_best.labels_
km_centroids = scaler.inverse_transform(km_best.cluster_centers_)

km_labels
```

[113]: array([1, 2, 2, 2, 2, 1, 1, 0, 2, 2, 1, 1, 2])

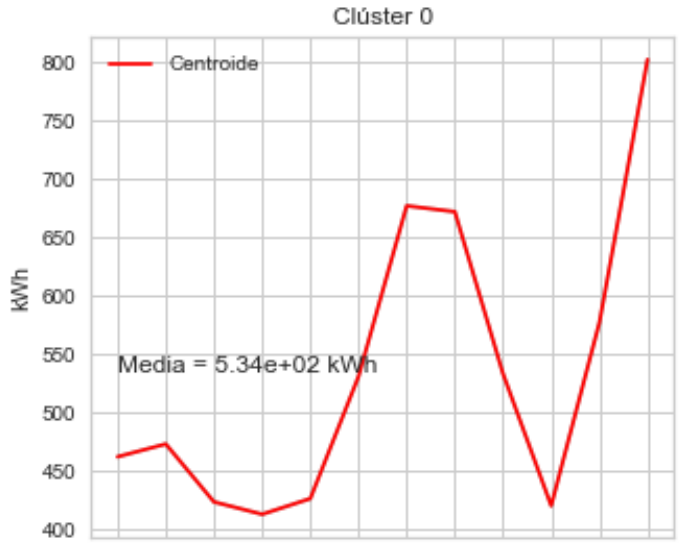
[114]:
```
_ = silhouette_visualizer(km_best, norm_data, colors='yellowbrick')

path = images_path / Path("ensayos/clustering_energia_media_en_contador/
 ↪mean_q1")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("silhouette.svg"))
```

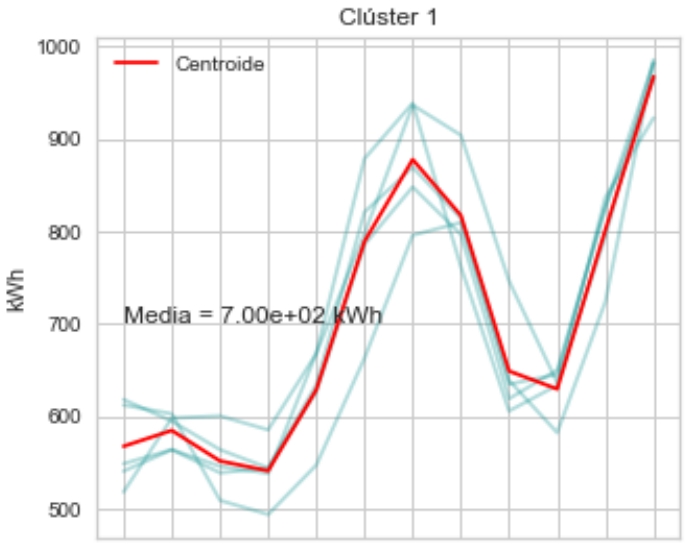


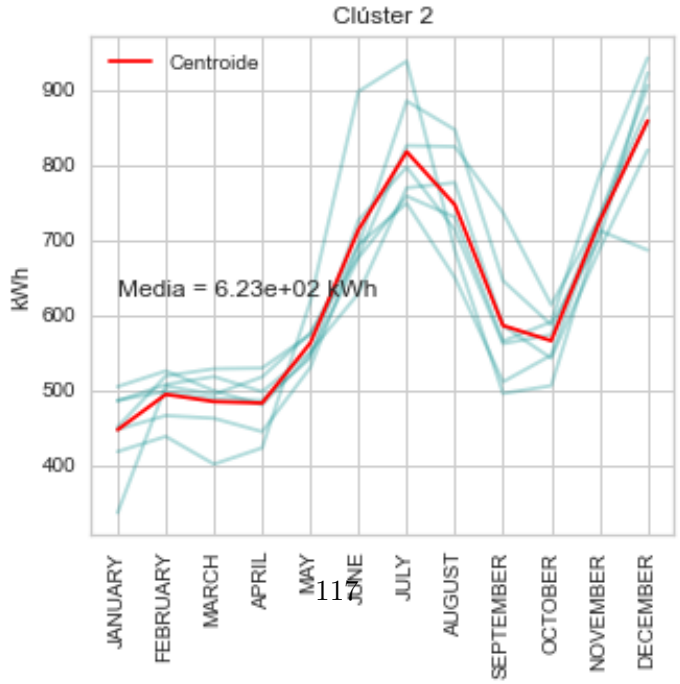
<Figure size 576x396 with 0 Axes>

[115]:
```
labels = pd.DataFrame(km_labels, columns=["label"])

k = km_centroids.shape[0]

f, axes = plt.subplots(k, 1, figsize=(5, 15), sharex=True)

for ki in range(k):
    cluster = mean_energy_q1.iloc[labels.query(f'label == {ki}').index][dl.
 ↪energy_cols]
    palette={i:"darkcyan" for i in range(cluster.shape[0])}

    ax0 = sns.lineplot(data=cluster.values.T, ax=axes[ki], palette=palette,␣
 ↪alpha=0.3)
    _ = [line.set_linestyle("-") for line in ax0.lines]

    ax0c = sns.lineplot(data=km_centroids[ki, :].T, ax=axes[ki], color="red")
    _ = [line.set_linestyle("-") for line in ax0c.lines]

    ax0c.legend(ax0c.lines[-1:], ["Centroide"]);
```

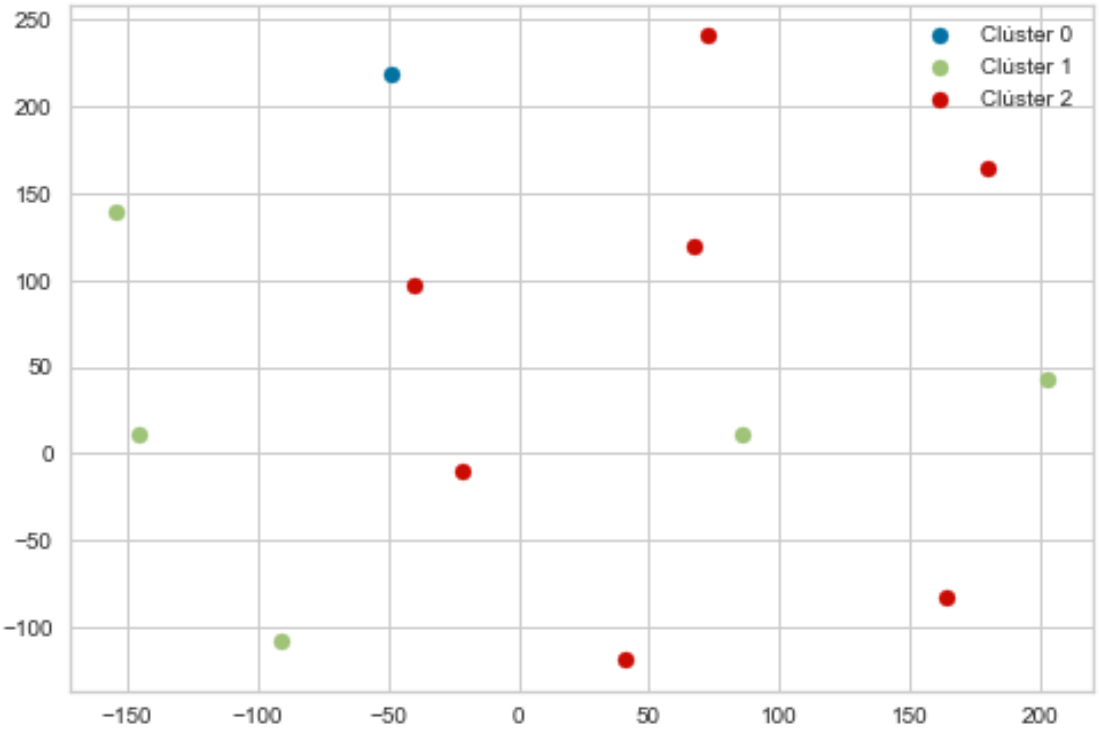
```
    ax0.set_title(f"Clúster {ki}")
    Media = km_centroids[ki, :].mean()
    ax0.text(0, Media, f"{Media = :0.2e} kWh")
    ax0.set(ylabel="kWh")

    ax0.set_xticks([e for e in range(0, 12)])
    ax0.set_xticklabels([f"{datetime.date(2008, i, 1).strftime('%B').upper()}"␣
 ↪for i in range(1, 13)], rotation=90)

path = images_path / Path("ensayos/clustering_energia_media_en_contador/
 ↪mean_q1")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("cluster_grafica.svg"))
```


Clúster 0
Centroide
kWh
Media = 5.34e+02 kWh


Clúster 1
Centroide
kWh
Media = 7.00e+02 kWh


Clúster 2
Centroide
kWh
Media = 6.23e+02 kWh
JANUARY
FEBRUARY
MARCH
APRIL
MAY
JUNE
JULY
AUGUST
SEPTEMBER
OCTOBER
NOVEMBER
DECEMBER

[116]:
```
v = tsne(mean_energy_q1[dl.energy_cols], 2)
for label in range(k):
    plt.scatter(
            x=v[(labels['label']==label).values, 0],
            y=v[(labels['label']==label).values, 1]
    )
plt.legend([f"Clúster {i}" for i in range(k)])

path = images_path / Path("ensayos/clustering_energia_media_en_contador/
 ↪mean_q1")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("scatter.svg"))
```



[117]:
```
v = tsne(mean_energy_q1[dl.energy_cols], 3)
fig = go.Figure()
for label in range(k):
    fig.add_trace(
        go.Scatter3d(
            x=v[(labels['label']==label).values, 0],
            y=v[(labels['label']==label).values, 1],
```

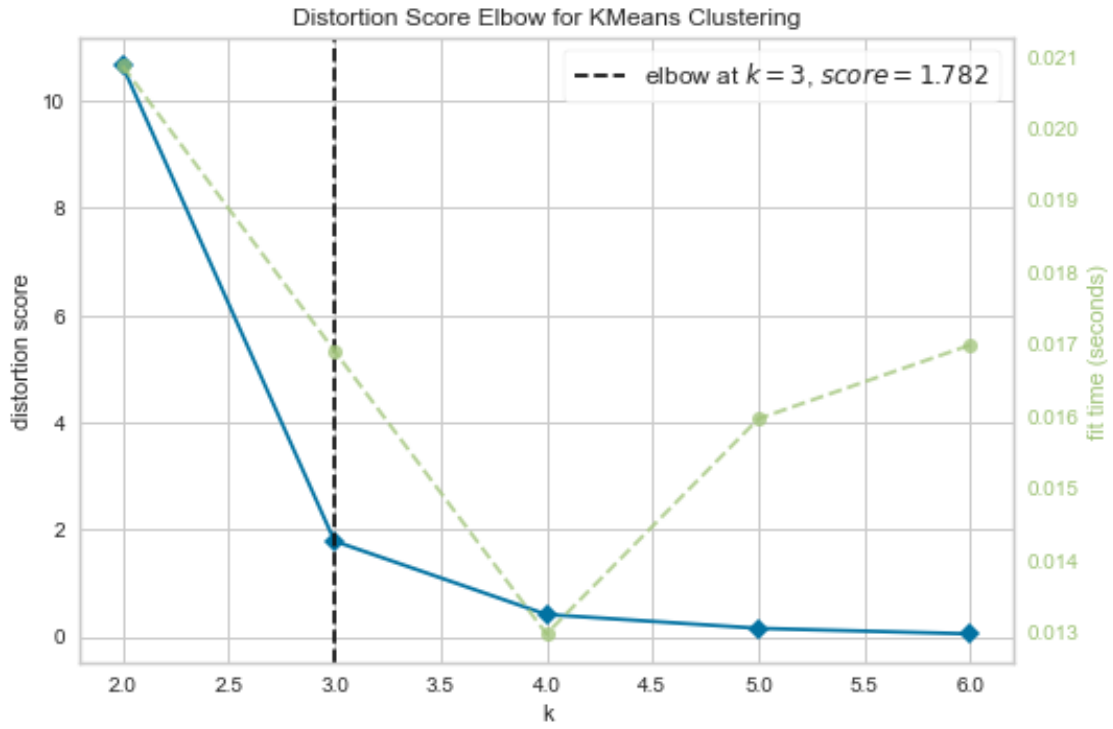
```
            z=v[(labels['label']==label).values, 2]
        )
    )
fig.show()

path = images_path / Path("ensayos/clustering_energia_media_en_contador/
 ↪mean_q1")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("scatter3d.svg"))
```

```
<Figure size 576x396 with 0 Axes>
```

[ ]:
```
comunities_by_cluster = {}

for nlab in range(k):
    com_index = np.where(labels==nlab)[0]
    com_list = []
    for c in mean_energy_q1.values[com_index][:,0]:
        com_list.append(c)
    comunities_by_cluster[nlab] = com_list

for k in sorted(comunities_by_cluster, key=lambda k:␣
 ↪len(comunities_by_cluster[k])):
    print(f"Al clúster {k} pertenecen las comunidades:")
    for e in comunities_by_cluster[k]:
        print("\t", e)
```

Aplicamos clustering a los valores en ***mean_q4***.

[119]:
```
kmax = 7
k_values = (2, kmax)

scaler = preprocessing.StandardScaler()
data = mean_energy_q4[dl.energy_cols]
norm_data = scaler.fit_transform(data)

km = KMeansCluster(norm_data)
km.cluster(k_values)

path = images_path / Path("ensayos/clustering_energia_media_en_contador/
 ↪mean_q4")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("elbow.svg"))
```

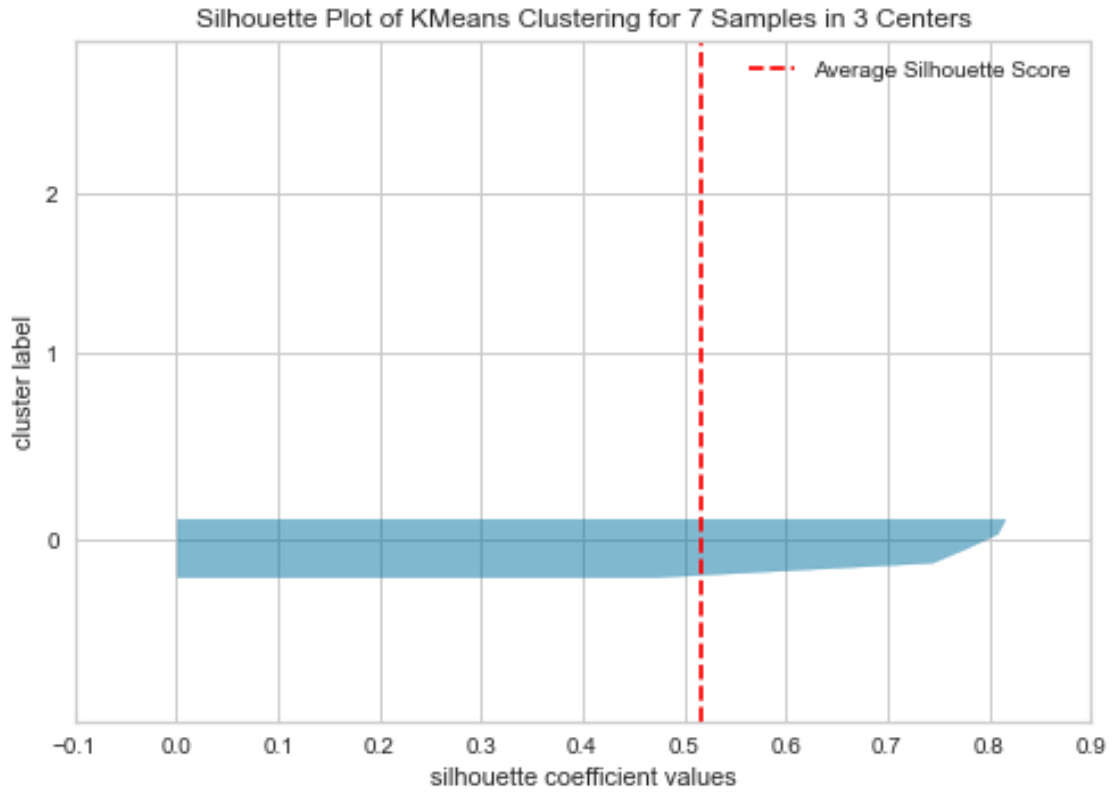



```
<Figure size 576x396 with 0 Axes>
```

[120]:
```
km_best = km.get_kmeans_of(3)
km_labels = km_best.labels_
km_centroids = scaler.inverse_transform(km_best.cluster_centers_)

km_labels
```

[120]:
```
array([0, 1, 0, 0, 0, 2, 0])
```

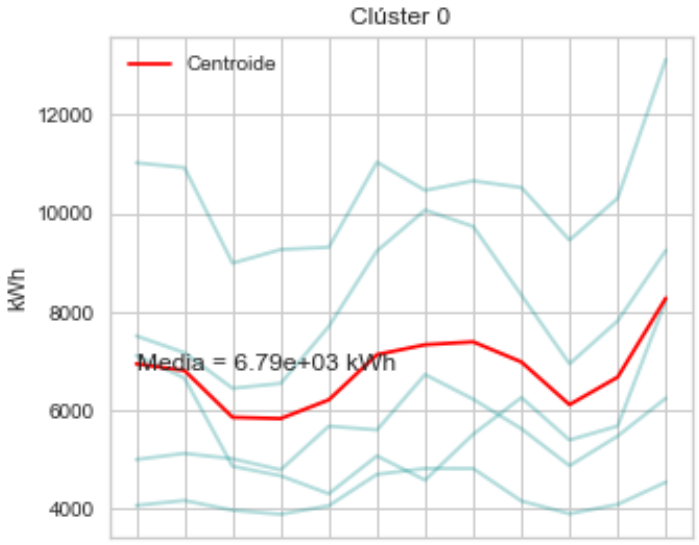
[121]:
```
_ = silhouette_visualizer(km_best, norm_data, colors='yellowbrick')

path = images_path / Path("ensayos/clustering_energia_media_en_contador/
 ↪mean_q4")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("silhouette.svg"))
```

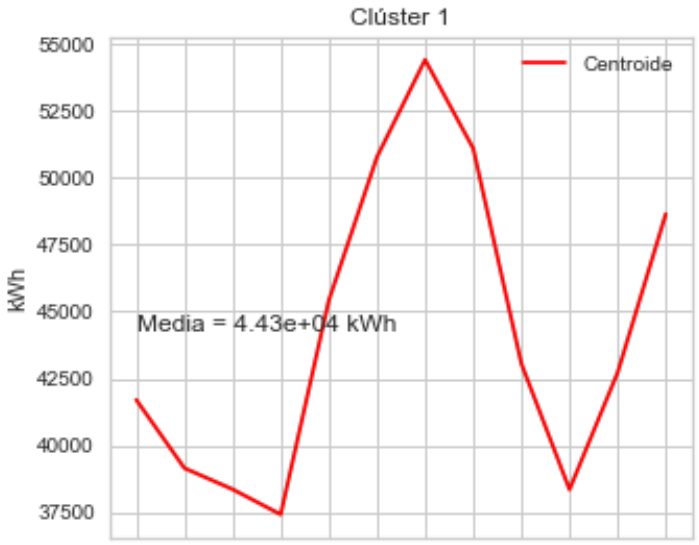


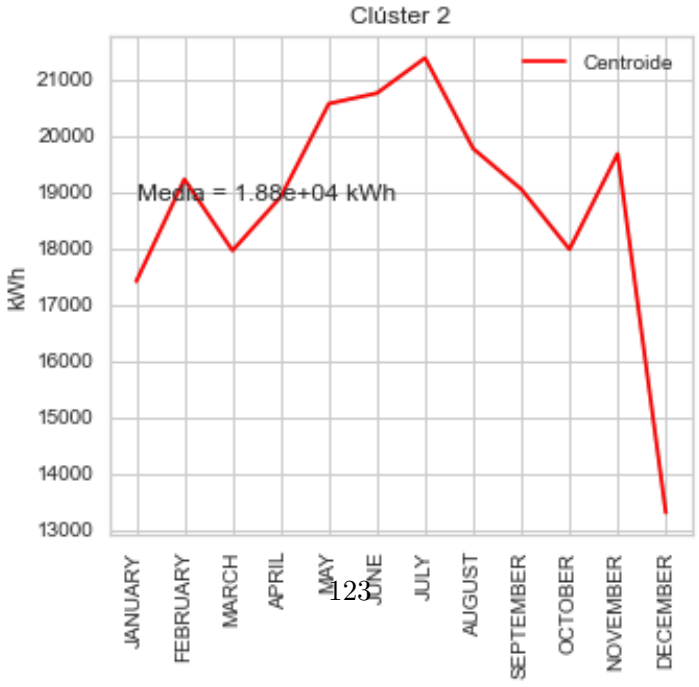
<Figure size 576x396 with 0 Axes>

[122]:
```
labels = pd.DataFrame(km_labels, columns=["label"])

k = km_centroids.shape[0]

f, axes = plt.subplots(k, 1, figsize=(5, 15), sharex=True)

for ki in range(k):
    cluster = mean_energy_q4.iloc[labels.query(f'label == {ki}').index][dl.
 ↪energy_cols]
    palette={i:"darkcyan" for i in range(cluster.shape[0])}

    ax0 = sns.lineplot(data=cluster.values.T, ax=axes[ki], palette=palette,␣
 ↪alpha=0.3)
    _ = [line.set_linestyle("-") for line in ax0.lines]

    ax0c = sns.lineplot(data=km_centroids[ki, :].T, ax=axes[ki], color="red")
    _ = [line.set_linestyle("-") for line in ax0c.lines]

    ax0c.legend(ax0c.lines[-1:], ["Centroide"]);
```

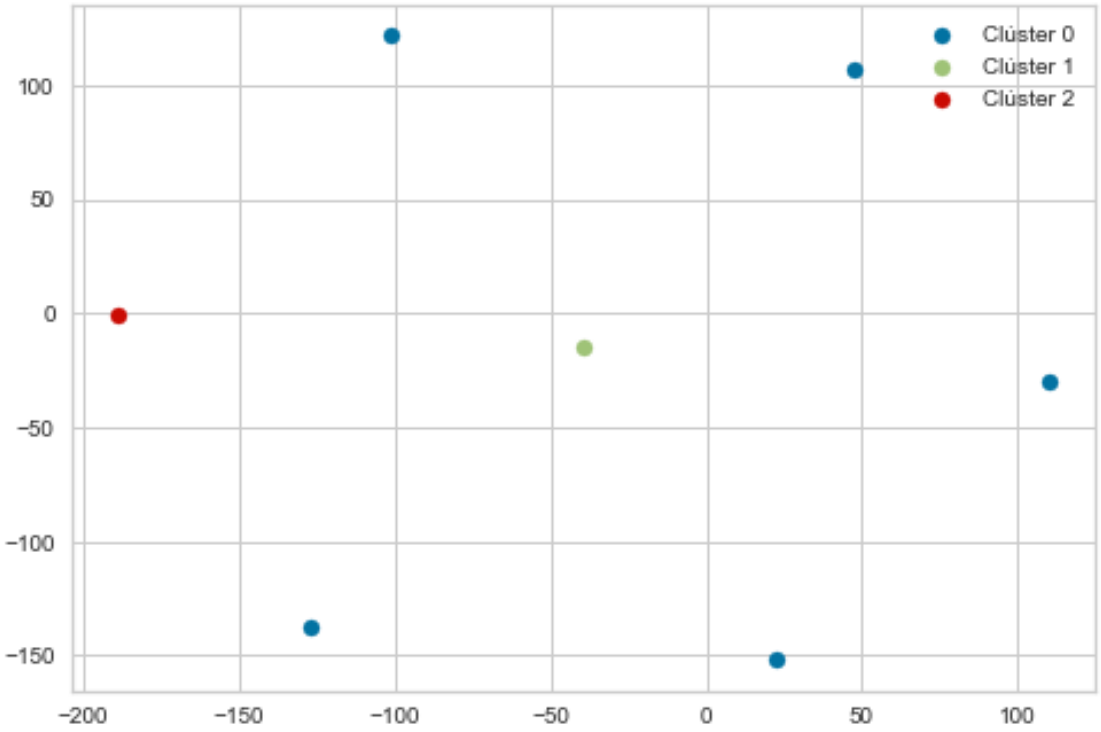
```
    ax0.set_title(f"Clúster {ki}")
    Media = km_centroids[ki, :].mean()
    ax0.text(0, Media, f"{Media = :0.2e} kWh")
    ax0.set(ylabel="kWh")

    ax0.set_xticks([e for e in range(0, 12)])
    ax0.set_xticklabels([f"{datetime.date(2008, i, 1).strftime('%B').upper()}"␣
 ↪for i in range(1, 13)], rotation=90)

path = images_path / Path("ensayos/clustering_energia_media_en_contador/
 ↪mean_q4")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("graficas.svg"))
```


Clúster 0
Centroide
12000
10000
8000
6000
4000
kWh
Media = 6.79e+03 kWh


Clúster 1
Centroide
55000
52500
50000
47500
45000
42500
40000
37500
kWh
Media = 4.43e+04 kWh


Clúster 2
Centroide
21000
20000
19000
18000
17000
16000
15000
14000
13000
kWh
Media = 1.88e+04 kWh
JANUARY
FEBRUARY
MARCH
APRIL
MAY
JUNE
JULY
AUGUST
SEPTEMBER
OCTOBER
NOVEMBER
DECEMBER

[123]:
```
v = tsne(mean_energy_q4[dl.energy_cols], 2)
for label in range(k):
    plt.scatter(
            x=v[(labels['label']==label).values, 0],
            y=v[(labels['label']==label).values, 1]
    )
plt.legend([f"Clúster {i}" for i in range(k)])

path = images_path / Path("ensayos/clustering_energia_media_en_contador/
 ↪mean_q4")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("scatter.svg"))
```



[ ]:
```
comunities_by_cluster = {}

for nlab in range(k):
    com_index = np.where(labels==nlab)[0]
    com_list = []
    for c in mean_energy_q4.values[com_index][:,0]:
        com_list.append(c)
```

```
    comunities_by_cluster[nlab] = com_list

for k in sorted(comunities_by_cluster, key=lambda k:␣
 ↪len(comunities_by_cluster[k])):
    print(f"Al clúster {k} pertenecen las comunidades:")
    for e in comunities_by_cluster[k]:
        print("\t", e)
```

1.3.10 Volver al inicio

1.3.11 Clustering de gas por media consumida en cada contador por comunidad

En este caso vamos a calcular el promedio consumido por cada contador en cada comunidad, dividiendo el consumo por la columna **GAS ACCOUNTS**.

[125]:
```
total_gas_by_com = dl[["COMMUNITY AREA NAME"] + dl.gas_cols + ["GAS ACCOUNTS"]]
total_gas_by_com
```

[125]:

	COMMUNITY AREA NAME	THERM JANUARY 2010	THERM FEBRUARY 2010	\
0	Archer Heights	2326.0	2131.0	
6	Austin	3041.0	2680.0	
11	Austin	910.0	738.0	
13	Avondale	535.0	458.0	
19	Avondale	285.0	285.0	
…	…	…	…	
67046	Woodlawn	2166.0	1681.0	
67047	Woodlawn	985.0	1152.0	
67048	Woodlawn	2202.0	1874.0	
67049	Woodlawn	95.0	11.0	
67050	Woodlawn	2372.0	1787.0	

	THERM MARCH 2010	THERM APRIL 2010	THERM MAY 2010	THERM JUNE 2010	\
0	1400.0	620.0	502.0	224.0	
6	1151.0	373.0	124.0	26.0	
11	632.0	448.0	254.0	113.0	
13	481.0	307.0	194.0	145.0	
19	244.0	129.0	73.0	46.0	
…	…	…	…	…	
67046	1858.0	1172.0	708.0	360.0	
67047	1238.0	630.0	475.0	192.0	
67048	1647.0	906.0	645.0	346.0	
67049	47.0	9.0	45.0	18.0	
67050	1449.0	718.0	572.0	286.0	

	THERM JULY 2010	THERM AUGUST 2010	THERM SEPTEMBER 2010	\
0	222.0	187.0	197.0	

```
6                 29.0              25.0                 49.0
11                42.0              29.0                 33.0
13               119.0             102.0                 88.0
19                39.0              27.0                 39.0
…                  …                 …                    …
67046             72.0              67.0                 77.0
67047            141.0             162.0                144.0
67048             84.0             150.0                150.0
67049             22.0               9.0                 17.0
67050            155.0             134.0                161.0

       THERM OCTOBER 2010  THERM NOVEMBER 2010  THERM DECEMBER 2010  \
0                   252.0                744.0                2112.0
6                   177.0                670.0                3895.0
11                   36.0                166.0                 633.0
13                  109.0                141.0                 249.0
19                   29.0                 50.0                 144.0
…                     …                   …                    …
67046               185.0                623.0                1800.0
67047               210.0                653.0                1744.0
67048               260.0                694.0                1335.0
67049                11.0                 18.0                  13.0
67050               303.0                588.0                1469.0

       GAS ACCOUNTS
0              11.0
6               3.0
11              3.0
13              4.0
19              3.0
…               …
67046           9.0
67047           8.0
67048           5.0
67049           5.0
67050          13.0

[60813 rows x 14 columns]
```

Dividimos el consumo de gas por el número de contadores.

[126]:
```
total_gas_by_com[dl.gas_cols] = total_gas_by_com[dl.gas_cols].
 ↪div(total_gas_by_com["GAS ACCOUNTS"], axis=0)
del total_gas_by_com["GAS ACCOUNTS"]
total_gas_by_com
```

[126]:

	COMMUNITY AREA NAME	THERM JANUARY 2010	THERM FEBRUARY 2010	\
0	Archer Heights	211.454545	193.727273	
6	Austin	1013.666667	893.333333	
11	Austin	303.333333	246.000000	
13	Avondale	133.750000	114.500000	
19	Avondale	95.000000	95.000000	
…	…	…	…	
67046	Woodlawn	240.666667	186.777778	
67047	Woodlawn	123.125000	144.000000	
67048	Woodlawn	440.400000	374.800000	
67049	Woodlawn	19.000000	2.200000	
67050	Woodlawn	182.461538	137.461538	

	THERM MARCH 2010	THERM APRIL 2010	THERM MAY 2010	THERM JUNE 2010	\
0	127.272727	56.363636	45.636364	20.363636	
6	383.666667	124.333333	41.333333	8.666667	
11	210.666667	149.333333	84.666667	37.666667	
13	120.250000	76.750000	48.500000	36.250000	
19	81.333333	43.000000	24.333333	15.333333	
…	…	…	…	…	
67046	206.444444	130.222222	78.666667	40.000000	
67047	154.750000	78.750000	59.375000	24.000000	
67048	329.400000	181.200000	129.000000	69.200000	
67049	9.400000	1.800000	9.000000	3.600000	
67050	111.461538	55.230769	44.000000	22.000000	

	THERM JULY 2010	THERM AUGUST 2010	THERM SEPTEMBER 2010	\
0	20.181818	17.000000	17.909091	
6	9.666667	8.333333	16.333333	
11	14.000000	9.666667	11.000000	
13	29.750000	25.500000	22.000000	
19	13.000000	9.000000	13.000000	
…	…	…	…	
67046	8.000000	7.444444	8.555556	
67047	17.625000	20.250000	18.000000	
67048	16.800000	30.000000	30.000000	
67049	4.400000	1.800000	3.400000	
67050	11.923077	10.307692	12.384615	

	THERM OCTOBER 2010	THERM NOVEMBER 2010	THERM DECEMBER 2010
0	22.909091	67.636364	192.000000
6	59.000000	223.333333	1298.333333
11	12.000000	55.333333	211.000000
13	27.250000	35.250000	62.250000
19	9.666667	16.666667	48.000000
…	…	…	…
67046	20.555556	69.222222	200.000000

```
67047          26.250000           81.625000          218.000000
67048          52.000000          138.800000          267.000000
67049           2.200000            3.600000            2.600000
67050          23.307692           45.230769          113.000000

[60813 rows x 13 columns]
```

Y posteriormente calculamos el promedio para cada comunidad.

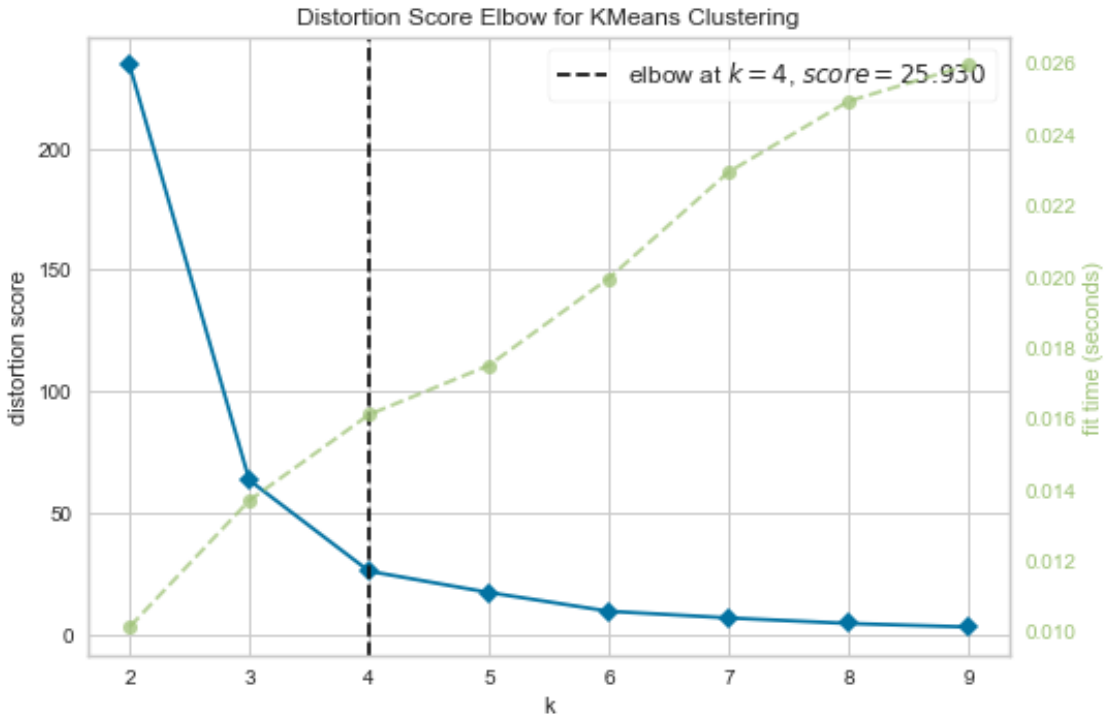
[127]:
```
mean_gas = total_gas_by_com.groupby('COMMUNITY AREA NAME', as_index=False).
↪mean()
mean_gas
```

[127]:
```
    COMMUNITY AREA NAME  THERM JANUARY 2010  THERM FEBRUARY 2010  \
0           Albany Park          284.552873           245.087784
1         Archer Heights          521.052922           401.826920
2          Armour Square          583.664740           451.322075
3                Ashburn          287.763035           247.270869
4         Auburn Gresham          323.149316           298.781050
..                   …                   …                   …
72            West Lawn          304.764895           266.360117
73          West Pullman          269.176338           251.701739
74            West Ridge          340.371726           278.831150
75            West Town          214.904567           183.123474
76             Woodlawn          371.700188           319.064021

    THERM MARCH 2010  THERM APRIL 2010  THERM MAY 2010  THERM JUNE 2010  \
0         200.396134        108.203023       70.542487        42.627449
1         315.615721        168.591343      108.174522        76.405301
2         487.174976        267.113941      193.111138       123.958385
3         235.673138        127.214171       65.933325        46.456245
4         232.171357        126.028303       77.288771        42.824096
..               …                 …               …                …
72        189.982854        107.564402       69.055657        45.487462
73        182.500750         97.649137       63.180883        28.492159
74        224.815387        133.599109       92.930962        64.734452
75        159.826261         78.785008       46.839160        30.773432
76        285.381080        160.449406      100.195986        57.049176

    THERM JULY 2010  THERM AUGUST 2010  THERM SEPTEMBER 2010  \
0         34.757723          30.944222             32.728462
1         73.128319          66.208327             66.431437
2        110.153765         108.135963            111.420355
3         28.085409          28.700234             26.699497
4         30.284555          29.387503             31.015608
..              …                  …                     …
72        41.903536          41.695916             40.238220
73        23.252768          22.823583             22.590791
```

```
74        57.671696          51.438850             45.241085
75        23.777846          21.285254             22.968842
76        39.392705          30.133532             41.408589

    THERM OCTOBER 2010  THERM NOVEMBER 2010  THERM DECEMBER 2010
0            39.958613            78.510739           194.184422
1            85.968940           163.912070           411.780009
2           190.671587           321.742738           590.266422
3            34.093083            63.663270           157.917898
4            50.249806           111.352088           274.396757
..                 …                    …                    …
72           49.723047            96.189277           247.669317
73           38.725627            91.067161           228.711587
74           51.940878           107.929262           247.420668
75           28.651958            53.462065           148.915897
76           60.148513           118.230107           246.791872

[77 rows x 13 columns]
```

Aplicamos clustering con KMeans.

[128]:
```
kmax = 10
k_values = (2, kmax)

scaler = preprocessing.StandardScaler()
data = mean_gas[dl.gas_cols]
norm_data = scaler.fit_transform(data)

km = KMeansCluster(norm_data)
km.cluster(k_values)

path = images_path / Path("ensayos/clustering_gas_media_en_contador/total")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("elbow.svg"))
```



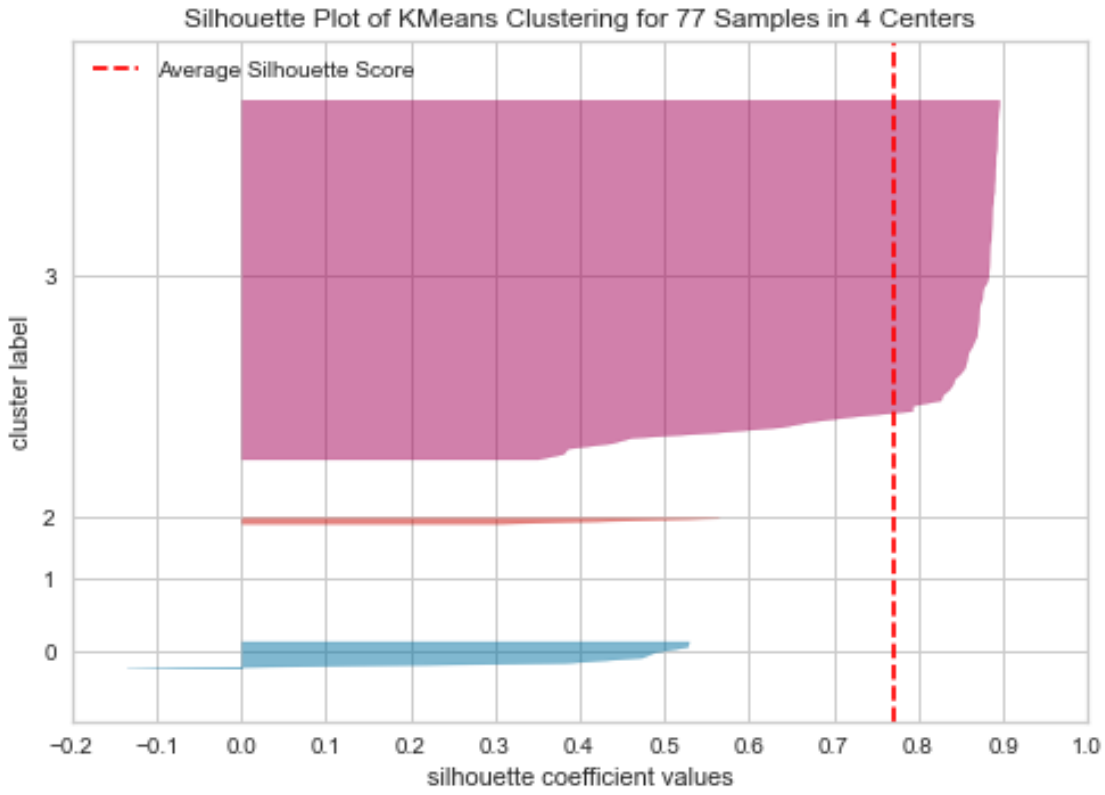

```
<Figure size 576x396 with 0 Axes>
```

Con k=4 tenemos la división óptima.

[129]:
```
km_best = km.get_kmeans_of(4)
km_labels = km_best.labels_
km_centroids = scaler.inverse_transform(km_best.cluster_centers_)

km_labels
```

[129]:
```
array([3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 0, 3, 3, 3, 3, 0, 3, 3, 3, 3,
       3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 0, 3, 3, 0, 3, 3, 3, 3, 1, 3, 3,
       3, 3, 3, 2, 0, 3, 3, 3, 3, 3, 3, 0, 3, 3, 3, 2, 3, 3, 3, 3, 3, 3,
       3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3])
```

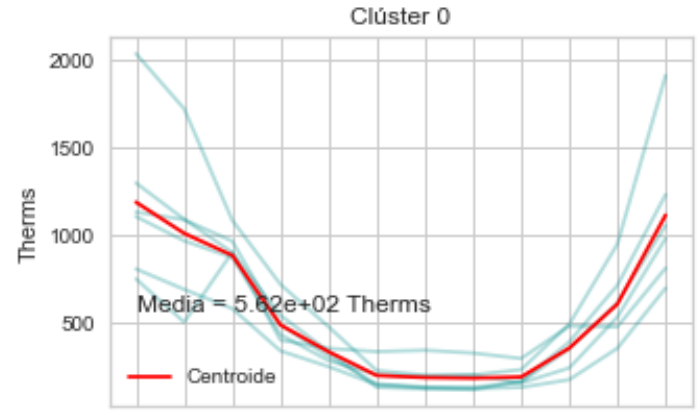
[130]:
```
_ = silhouette_visualizer(km_best, norm_data, colors='yellowbrick')

path = images_path / Path("ensayos/clustering_gas_media_en_contador/total")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("silhouette.svg"))
```

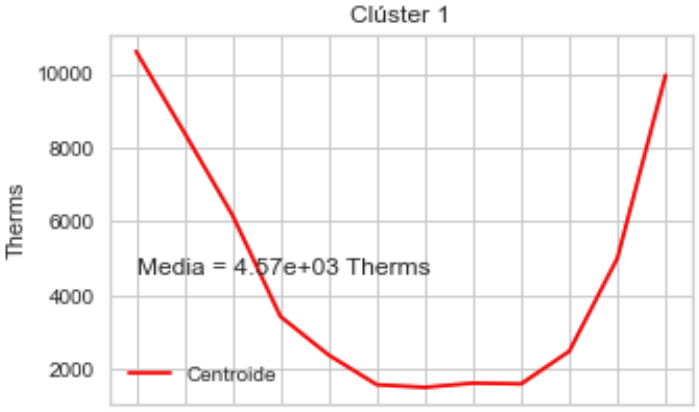


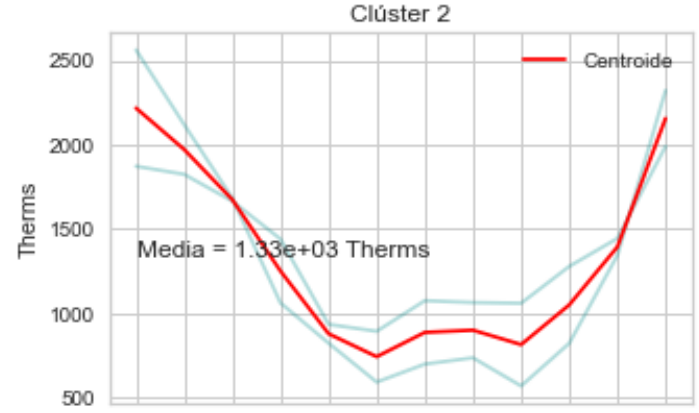

```
<Figure size 576x396 with 0 Axes>
```

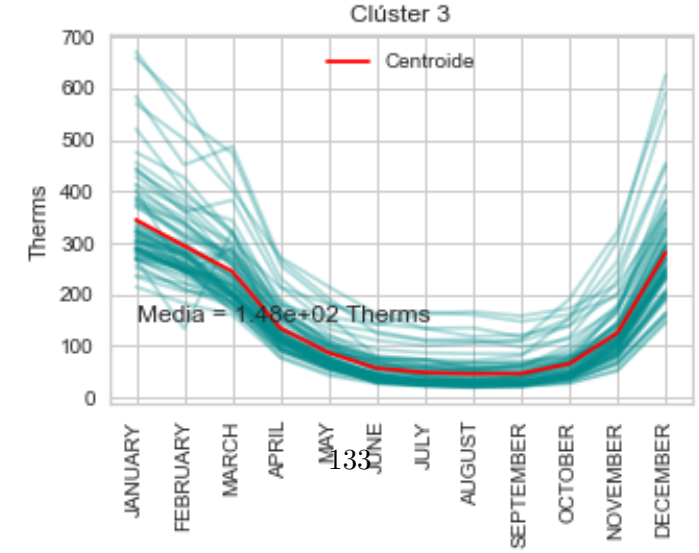
[131]:
```
labels = pd.DataFrame(km_labels, columns=["label"])

k = km_centroids.shape[0]

f, axes = plt.subplots(k, 1, figsize=(5, 15), sharex=True)

for ki in range(k):
    cluster = mean_gas.iloc[labels.query(f'label == {ki}').index][dl.gas_cols]
    palette={i:"darkcyan" for i in range(cluster.shape[0])}

    ax0 = sns.lineplot(data=cluster.values.T, ax=axes[ki], palette=palette,␣
 ↪alpha=0.3)
    _ = [line.set_linestyle("-") for line in ax0.lines]

    ax0c = sns.lineplot(data=km_centroids[ki, :].T, ax=axes[ki], color="red")
    _ = [line.set_linestyle("-") for line in ax0c.lines]

    ax0c.legend(ax0c.lines[-1:], ["Centroide"]);

    ax0.set_title(f"Clúster {ki}")
```

```
    Media = km_centroids[ki, :].mean()
    ax0.text(0, Media, f"{Media = :0.2e} Therms")
    ax0.set(ylabel="Therms")

    ax0.set_xticks([e for e in range(0, 12)])
    ax0.set_xticklabels([f"{datetime.date(2008, i, 1).strftime('%B').upper()}"␣
↪for i in range(1, 13)], rotation=90)

path = images_path / Path("ensayos/clustering_gas_media_en_contador/total")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("graficas.svg"))
```

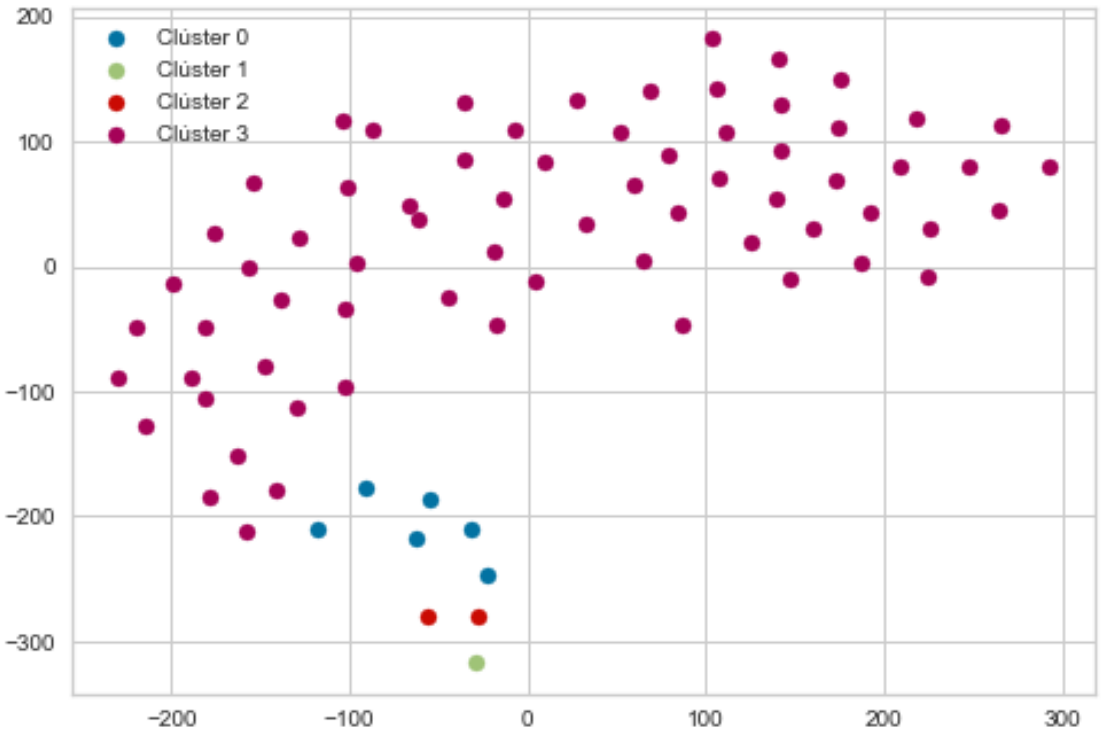

Clúster 0
Therms
2000
1500
1000
500
Media = 5.62e+02 Therms
Centroide


Clúster 1
Therms
10000
8000
6000
4000
2000
Media = 4.57e+03 Therms
Centroide


Clúster 2
Therms
2500
2000
1500
1000
500
Media = 1.33e+03 Therms
Centroide


Clúster 3
Therms
700
600
500
400
300
200
100
0
Media = 1.48e+02 Therms
Centroide
JANUARY
FEBRUARY
MARCH
APRIL
MAY
JUNE
JULY
AUGUST
SEPTEMBER
OCTOBER
NOVEMBER
DECEMBER

[133]:
```
v = tsne(mean_gas[dl.gas_cols], 2)
for label in range(k):
    plt.scatter(
            x=v[(labels['label']==label).values, 0],
            y=v[(labels['label']==label).values, 1]
    )
plt.legend([f"Clúster {i}" for i in range(k)])

path = images_path / Path("ensayos/clustering_gas_media_en_contador/total")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("scatter.svg"))
```



¿Qué comunidades hay en cada clúster?

[ ]:
```
comunities_by_cluster = {}

for nlab in range(k):
    com_index = np.where(labels==nlab)[0]
    com_list = []
    for c in mean_gas.values[com_index][:,0]:
        com_list.append(c)
```

```
    comunities_by_cluster[nlab] = com_list

for k in sorted(comunities_by_cluster, key=lambda k:␣
 ↪len(comunities_by_cluster[k])):
    print(f"Al clúster {k} pertenecen las comunidades:")
    for e in comunities_by_cluster[k]:
        print("\t", e)
```

Separación por cuartiles Vamos a separar las muestras muy alejadas del valor medio mediante la eliminación de outliers. Dividiremos en 3 porciones: los que están muy por abajo de la media, en *mean_q1*; los que están muy por encima de la media, en *mean_q4*; y los que se encuentran sobre la media, en *mean_qm*. Con este tratamiento previo trataremos de encontrar de mejor modo patrones dentro de cada grupo.

[135]:
```
zmin_score = 0.26

z_mean_gas = zscore(mean_gas[dl.gas_cols])
mean_gas_qm = mean_gas[np.logical_and(-zmin_score <= z_mean_gas, z_mean_gas <=␣
 ↪zmin_score).all(axis=1)]
mean_gas_q1 = mean_gas[(-zmin_score > z_mean_gas).all(axis=1)]
mean_gas_q4 = mean_gas[(zmin_score < z_mean_gas).all(axis=1)]

mean_gas_qm
```

[135]:

	COMMUNITY AREA NAME	THERM JANUARY 2010	THERM FEBRUARY 2010	\
1	Archer Heights	521.052922	401.826920	
2	Armour Square	583.664740	451.322075	
5	Austin	375.527418	340.070165	
11	Brighton Park	329.036687	270.078502	
21	Edgewater	658.917630	570.229261	
31	Hermosa	413.722371	368.280032	
37	Lakeview	437.751977	382.584217	
42	Lower West Side	443.226312	359.981507	
43	McKinley Park	323.566614	280.694309	
49	Near West Side	671.089125	541.241674	
50	New City	475.741516	426.900098	
53	North Park	411.453397	337.025392	
65	South Shore	384.221957	313.447754	
66	Uptown	456.350989	393.458826	

	THERM MARCH 2010	THERM APRIL 2010	THERM MAY 2010	THERM JUNE 2010	\
1	315.615721	168.591343	108.174522	76.405301	
2	487.174976	267.113941	193.111138	123.958385	
5	256.766228	147.349960	101.834839	64.277464	
11	239.283083	132.245368	91.166026	68.551441	
21	416.664880	231.148473	156.636148	79.454516	
31	286.225292	183.922083	133.238923	110.972275	

```
37        295.267978        166.219545      109.859130        67.378199
42        382.984658        216.159551      142.063576       128.130953
43        276.185830        140.544196      101.248356        80.469367
49        473.597290        257.104121      183.030549       143.469029
50        319.378215        161.227358      118.385827        93.777355
53        269.959717        155.258982       94.467697        61.219057
65        280.513328        169.539498      112.710766        73.521336
66        323.010425        190.977445      122.452670        76.596368

    THERM JULY 2010  THERM AUGUST 2010  THERM SEPTEMBER 2010  \
1         73.128319          66.208327             66.431437
2        110.153765         108.135963            111.420355
5         59.122924          56.433242             54.945450
11        51.058345          52.244289             62.327791
21        73.251229          71.783102             81.109346
31       100.823497         105.861043            109.505919
37        57.978375          56.030234             60.159221
42       111.958657         112.407813             98.313283
43        73.856148          57.956081             62.759469
49       134.390934         135.583177            117.382970
50        86.933380          85.099924             84.569976
53        58.709252          50.499208             47.940140
65        73.531265          54.686490             60.867523
66        64.350477          59.349936             60.648616

    THERM OCTOBER 2010  THERM NOVEMBER 2010  THERM DECEMBER 2010
1            85.968940           163.912070           411.780009
2           190.671587           321.742738           590.266422
5            69.581625           143.622471           315.235657
11           64.676049           104.926773           232.598242
21          154.394763           303.713912           625.323361
31          138.939544           202.431431           382.412413
37           92.558238           169.443953           352.299955
42          113.654632           172.630533           325.808930
43           75.724301           112.027159           234.817817
49          152.604912           264.358699           555.532796
50          117.744776           227.770992           447.706175
53           69.290616           135.998739           292.478673
65           77.075018           139.752888           267.479329
66          102.107306           179.623707           365.326735
```

[136]:
```
mean_gas_q1
```

[136]:
```
    COMMUNITY AREA NAME  THERM JANUARY 2010  THERM FEBRUARY 2010  \
20            East Side          234.096488           205.636718
30            Hegewisch          256.971420           209.963993
35      Jefferson Park           267.822697           239.742414
```

```
40        Logan Square          237.602956           220.901166
51        North Center          251.140221           211.428602
75           West Town          214.904567           183.123474

    THERM MARCH 2010  THERM APRIL 2010  THERM MAY 2010  THERM JUNE 2010  \
20        200.159872        104.559541       60.048371        40.656037
30        185.351310         99.199959       57.086837        33.176131
35        186.320303        110.622652       58.567879        37.180809
40        200.925168         95.981718       62.083632        42.810736
51        175.547574         92.889053       56.446202        32.115566
75        159.826261         78.785008       46.839160        30.773432

    THERM JULY 2010  THERM AUGUST 2010  THERM SEPTEMBER 2010  \
20        24.424548          21.883132             22.843186
30        25.057912          21.692122             23.137106
35        27.102508          24.842575             25.624770
40        34.941626          32.565628             33.249863
51        25.250018          20.648477             20.614367
75        23.777846          21.285254             22.968842

    THERM OCTOBER 2010  THERM NOVEMBER 2010  THERM DECEMBER 2010
20           27.464492            52.064187           144.921484
30           31.599408            69.430850           158.439756
35           36.801825            82.166749           190.918519
40           46.279649            87.237568           204.288151
51           32.650028            67.422616           164.334048
75           28.651958            53.462065           148.915897
```

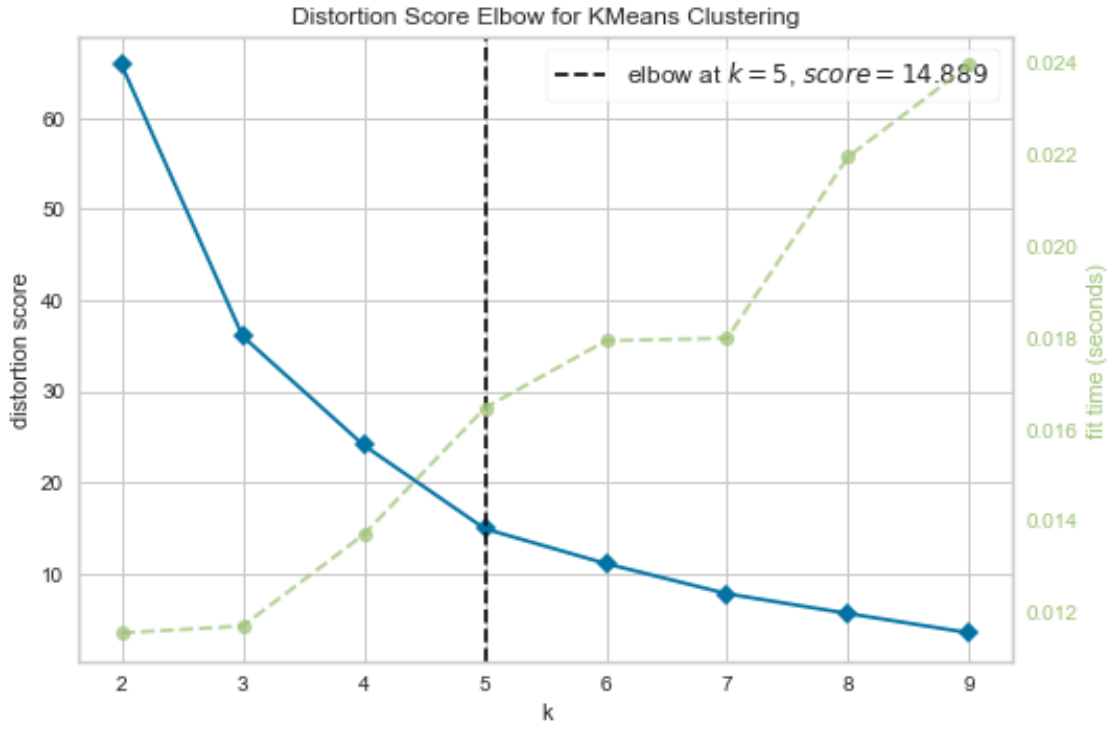
[137]:
```
mean_gas_q4
```

[137]:
```
   COMMUNITY AREA NAME  THERM JANUARY 2010  THERM FEBRUARY 2010  \
41                Loop        10605.645853          8416.081631
47     Near North Side         2561.680909          2119.106513
48     Near South Side         2039.334959          1723.952502
59           Riverdale         1872.426728          1824.674491

    THERM MARCH 2010  THERM APRIL 2010  THERM MAY 2010  THERM JUNE 2010  \
41       6183.463934       3435.484118     2396.747512      1592.728842
47       1683.767077       1062.919003      824.563958       593.747481
48       1085.831066        719.351076      476.540802       226.049091
59       1665.893264       1441.338703      935.926139       894.621183

    THERM JULY 2010  THERM AUGUST 2010  THERM SEPTEMBER 2010  \
41      1520.397669         1634.239188           1613.739611
47       700.338868          737.632438            570.851880
48       200.692879          200.828904            231.522441
59      1076.188708         1065.118085           1060.908624
```

	THERM OCTOBER 2010	THERM NOVEMBER 2010	THERM DECEMBER 2010
41	2495.810977	5013.781157	9956.751008
47	823.457823	1344.985954	2320.957193
48	491.562926	950.021815	1913.890225
59	1279.218855	1444.840866	1990.056394

Aplicamos clustering a los valores en ***mean_qm***.

[138]:
```
kmax = 10
k_values = (2, kmax)

scaler = preprocessing.StandardScaler()
data = mean_gas_qm[dl.gas_cols]
norm_data = scaler.fit_transform(data)

km = KMeansCluster(norm_data)
km.cluster(k_values)

path = images_path / Path("ensayos/clustering_gas_media_en_contador/mean_qm")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("elbow.svg"))
```

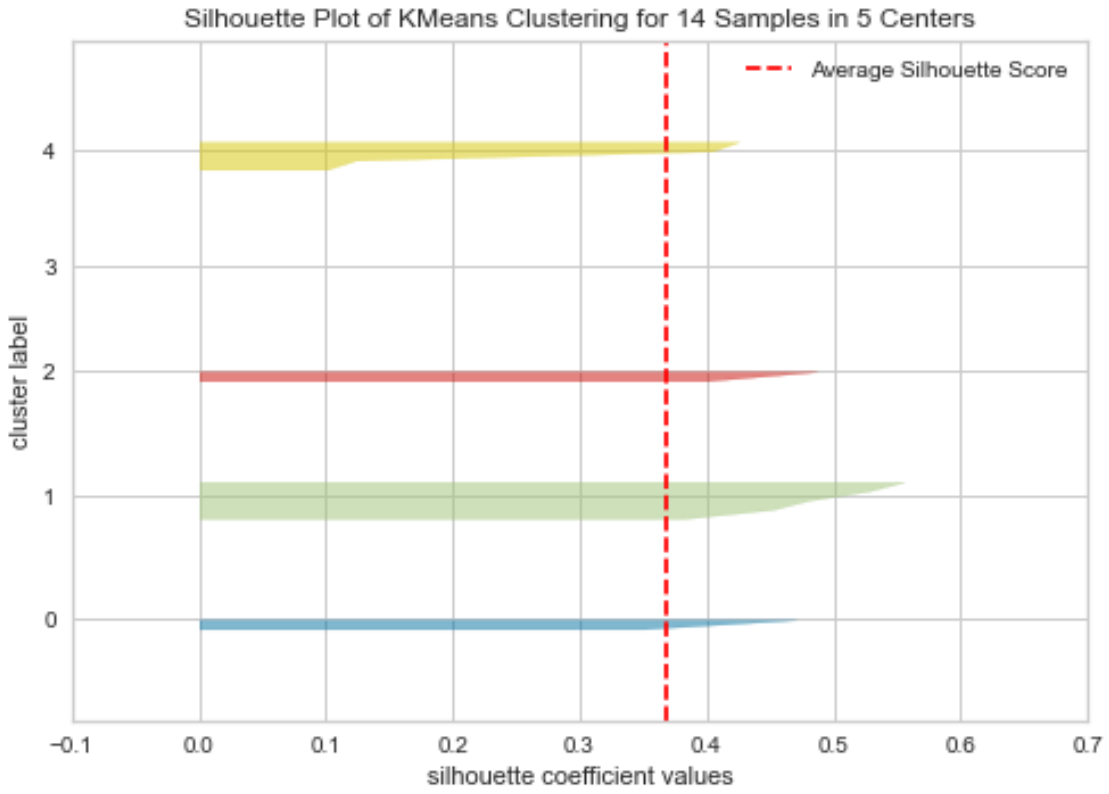


<Figure size 576x396 with 0 Axes>

[139]:
```
km_best = km.get_kmeans_of(5)
km_labels = km_best.labels_
km_centroids = scaler.inverse_transform(km_best.cluster_centers_)

km_labels
```

[139]:
```
array([4, 0, 1, 1, 3, 2, 4, 2, 1, 0, 4, 1, 1, 4])
```

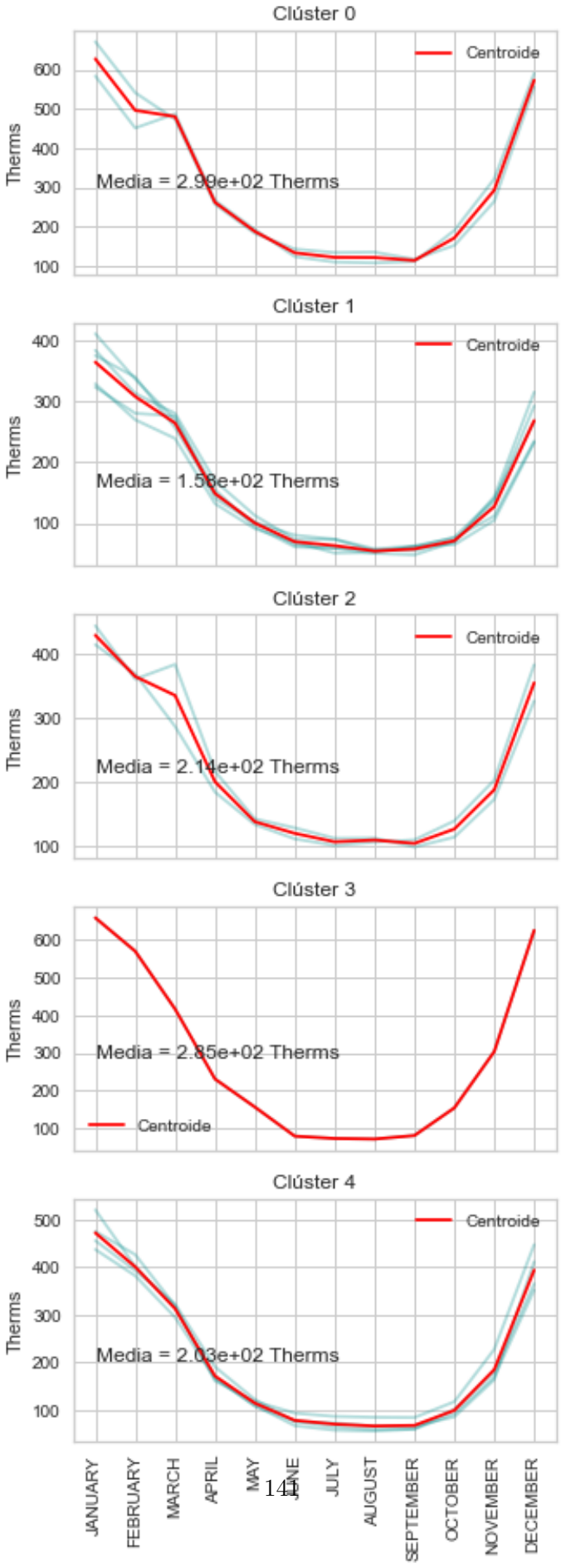
[140]:
```
_ = silhouette_visualizer(km_best, norm_data, colors='yellowbrick')

path = images_path / Path("ensayos/clustering_gas_media_en_contador/mean_qm")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("silhouette.svg"))
```



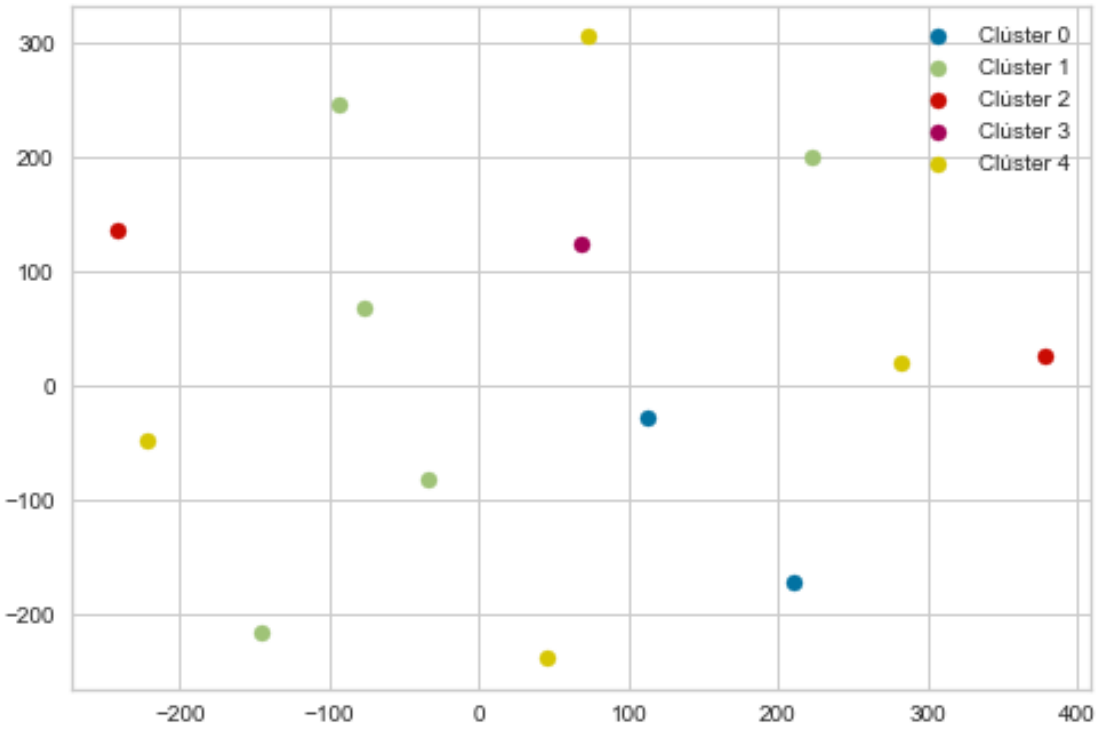
<Figure size 576x396 with 0 Axes>

[141]:
```
labels = pd.DataFrame(km_labels, columns=["label"])

k = km_centroids.shape[0]

f, axes = plt.subplots(k, 1, figsize=(5, 15), sharex=True)
```

```
for ki in range(k):
    cluster = mean_gas_qm.iloc[labels.query(f'label == {ki}').index][dl.
 ↪gas_cols]
    palette={i:"darkcyan" for i in range(cluster.shape[0])}

    ax0 = sns.lineplot(data=cluster.values.T, ax=axes[ki], palette=palette,␣
 ↪alpha=0.3)
    _ = [line.set_linestyle("-") for line in ax0.lines]

    ax0c = sns.lineplot(data=km_centroids[ki, :].T, ax=axes[ki], color="red")
    _ = [line.set_linestyle("-") for line in ax0c.lines]

    ax0c.legend(ax0c.lines[-1:], ["Centroide"]);

    ax0.set_title(f"Clúster {ki}")
    Media = km_centroids[ki, :].mean()
    ax0.text(0, Media, f"{Media = :0.2e} Therms")
    ax0.set(ylabel="Therms")

    ax0.set_xticks([e for e in range(0, 12)])
    ax0.set_xticklabels([f"{datetime.date(2008, i, 1).strftime('%B').upper()}"␣
 ↪for i in range(1, 13)], rotation=90)

path = images_path / Path("ensayos/clustering_gas_media_en_contador/mean_qm")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("grafica.svg"))
```


Clúster 0
Centroide
Therms
Media = 2.99e+02 Therms
Clúster 1
Centroide
Therms
Media = 1.58e+02 Therms
Clúster 2
Centroide
Therms
Media = 2.14e+02 Therms
Clúster 3
Therms
Media = 2.85e+02 Therms
Centroide
Clúster 4
Centroide
Therms
Media = 2.03e+02 Therms
JANUARY
FEBRUARY
MARCH
APRIL
MAY
JUNE
JULY
AUGUST
SEPTEMBER
OCTOBER
NOVEMBER
DECEMBER

[142]:
```
v = tsne(mean_gas_qm[dl.gas_cols], 2)
for label in range(k):
    plt.scatter(
            x=v[(labels['label']==label).values, 0],
            y=v[(labels['label']==label).values, 1]
    )
plt.legend([f"Clúster {i}" for i in range(k)])

path = images_path / Path("ensayos/clustering_gas_media_en_contador/mean_qm")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("scatter.svg"))
```

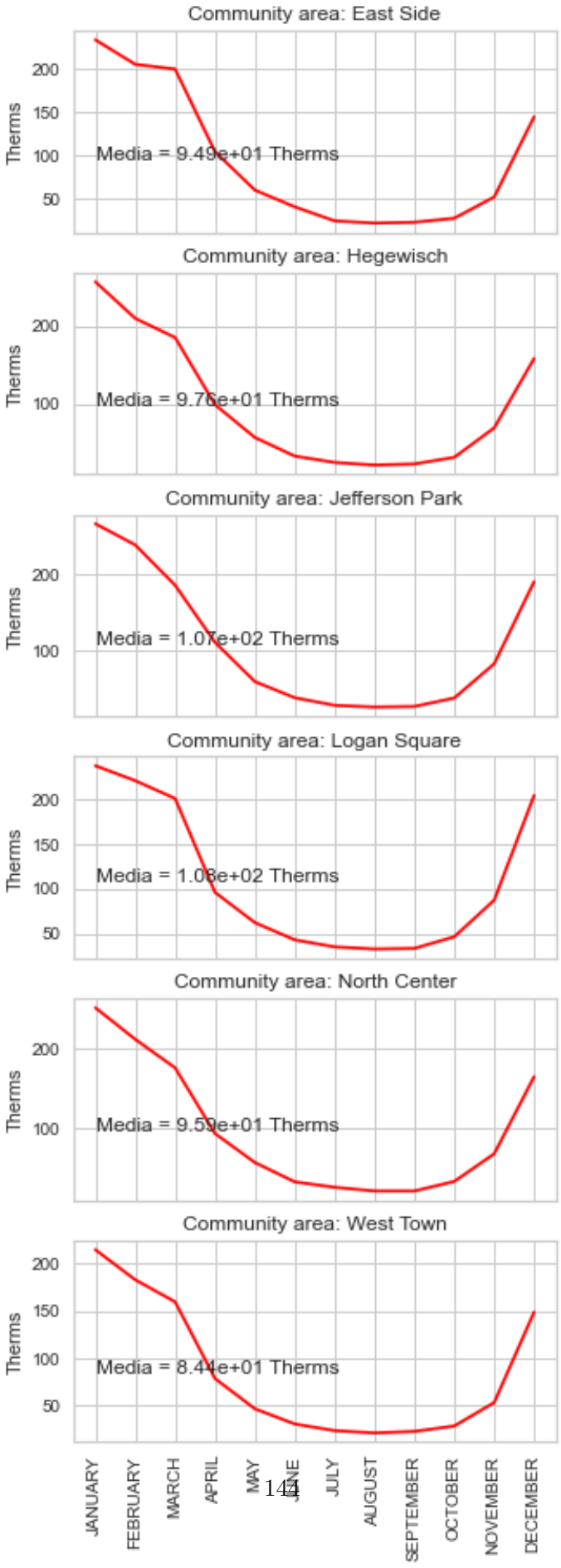


[ ]:
```
comunities_by_cluster = {}

for nlab in range(k):
    com_index = np.where(labels==nlab)[0]
    com_list = []
    for c in mean_gas_qm.values[com_index][:,0]:
        com_list.append(c)
    comunities_by_cluster[nlab] = com_list
```

```
for k in sorted(comunities_by_cluster, key=lambda k:␣
↪len(comunities_by_cluster[k])):
    print(f"Al clúster {k} pertenecen las comunidades:")
    for e in comunities_by_cluster[k]:
        print("\t", e)
```

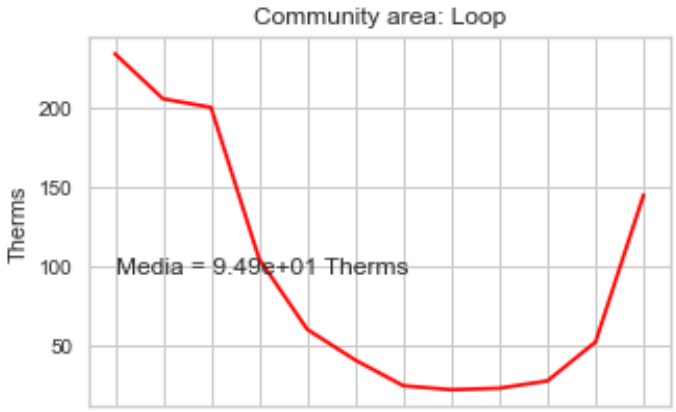
Visualizamos ***mean_q1***.

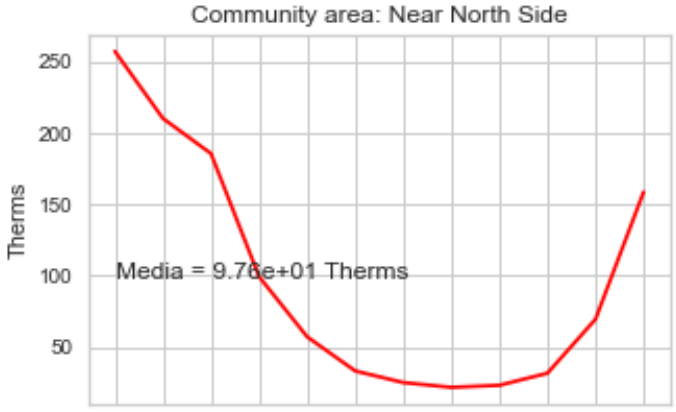
[144]:
```
f, axes = plt.subplots(6, 1, figsize=(5, 15), sharex=True)

for i, (com_name, (index, row)) in enumerate(zip(mean_gas_q1['COMMUNITY AREA␣
↪NAME'], mean_gas_q1[dl.gas_cols].iterrows())):
    ax0 = sns.lineplot(data=row.T, ax=axes[i], color="red")

    ax0.set_xticks([e for e in range(0, 12)])
    ax0.set_xticklabels([f"{datetime.date(2008, i, 1).strftime('%B').upper()}"␣
↪for i in range(1, 13)], rotation=90)

    ax0.set_title(f"Community area: {com_name}")
    Media = row.mean()
    ax0.text(0, Media, f"{Media = :0.2e} Therms")

    ax0.set(ylabel="Therms")

path = images_path / Path("ensayos/clustering_gas_media_en_contador/mean_q1")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("grafica.svg"))
```

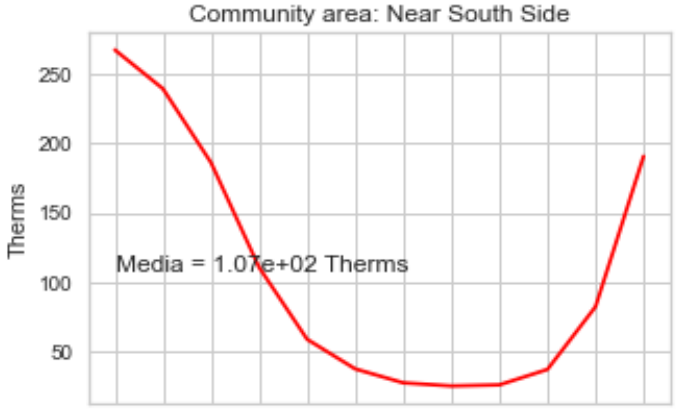

Community area: East Side
Therms
200
150
100
50
Media = 9.49e+01 Therms
Community area: Hegewisch
Therms
200
100
Media = 9.76e+01 Therms
Community area: Jefferson Park
Therms
200
100
Media = 1.07e+02 Therms
Community area: Logan Square
Therms
200
150
100
50
Media = 1.08e+02 Therms
Community area: North Center
Therms
200
100
Media = 9.59e+01 Therms
Community area: West Town
Therms
200
150
100
50
Media = 8.44e+01 Therms
JANUARY
FEBRUARY
MARCH
APRIL
MAY
JUNE
JULY
AUGUST
SEPTEMBER
OCTOBER
NOVEMBER
DECEMBER

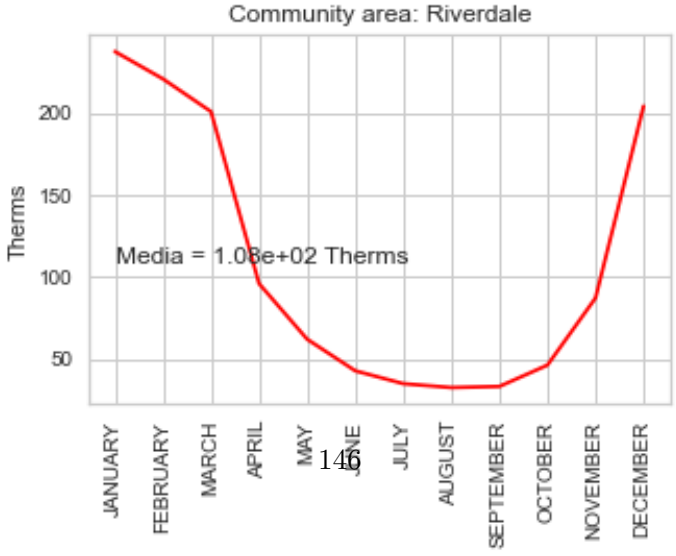
Visualizamos ***mean_q4***.

[145]:
```
f, axes = plt.subplots(4, 1, figsize=(5, 15), sharex=True)

for i, (com_name, (index, row)) in enumerate(zip(mean_gas_q4['COMMUNITY AREA␣
↪NAME'], mean_gas_q1[dl.gas_cols].iterrows())):
    ax0 = sns.lineplot(data=row.T, ax=axes[i], color="red")

    ax0.set_xticks([e for e in range(0, 12)])
    ax0.set_xticklabels([f"{datetime.date(2008, i, 1).strftime('%B').upper()}"␣
↪for i in range(1, 13)], rotation=90)

    ax0.set_title(f"Community area: {com_name}")
    Media = row.mean()
    ax0.text(0, Media, f"{Media = :0.2e} Therms")

    ax0.set(ylabel="Therms")

path = images_path / Path("ensayos/clustering_gas_media_en_contador/mean_q4")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("grafica.svg"))
```


Community area: Loop
Therms
200
150
100
50
Media = 9.49e+01 Therms


Community area: Near North Side
Therms
250
200
150
100
50
Media = 9.76e+01 Therms


Community area: Near South Side
Therms
250
200
150
100
50
Media = 1.07e+02 Therms


Community area: Riverdale
Therms
200
150
100
50
Media = 1.08e+02 Therms
JANUARY
FEBRUARY
MARCH
APRIL
MAY
JUNE
JULY
AUGUST
SEPTEMBER
OCTOBER
NOVEMBER
DECEMBER

1.3.12 Volver al inicio

1.3.13 Clustering por edad del edificio

En este caso vamos a buscar patrones según la edad promedio de los edificios de cada comunidad.

[146]:
```
building_age = dl[['AVERAGE BUILDING AGE']]
building_age
```

[146]:

	AVERAGE BUILDING AGE
0	71.33
1	41.00
2	86.00
3	84.00
4	85.00
…	…
67046	0.00
67047	104.50
67048	100.67
67049	0.00
67050	79.40

[67051 rows x 1 columns]

El rango de edad promedio es:

[147]:
```
f"La edad media de los edificios oscila entre {building_age.min().item()} y␣
↪{building_age.max().item()} años (incluyendo filas nulas)"
```

[147]:
```
'La edad media de los edificios oscila entre 0.0 y 158.0 años (incluyendo filas
nulas)'
```

[148]:
```
energy_building_age = dl[['AVERAGE BUILDING AGE', 'COMMUNITY AREA NAME',␣
↪'CENSUS BLOCK', 'BUILDING TYPE', 'BUILDING_SUBTYPE'] + dl.energy_cols]
gas_building_age = dl[['AVERAGE BUILDING AGE', 'COMMUNITY AREA NAME', 'CENSUS␣
↪BLOCK', 'BUILDING TYPE', 'BUILDING_SUBTYPE'] + dl.gas_cols]
```

[149]:
```
f"Para las filas no nulas de energía, la edad oscila entre␣
↪{energy_building_age[['AVERAGE BUILDING AGE']].min().item()} y␣
↪{energy_building_age[['AVERAGE BUILDING AGE']].max().item()}"
```

[149]:
```
'Para las filas no nulas de energía, la edad oscila entre 0.0 y 158.0'
```

[150]:
```
f"Para las filas no nulas de energía, la edad oscila entre␣
↪{gas_building_age[['AVERAGE BUILDING AGE']].min().item()} y␣
↪{gas_building_age[['AVERAGE BUILDING AGE']].max().item()}"
```

[150]:
```
'Para las filas no nulas de energía, la edad oscila entre 0.0 y 153.5'
```

Energía Aplicamos clustering a las muestras de consumo de electricidad.

[151]:
```
energy_building_age
```

[151]:

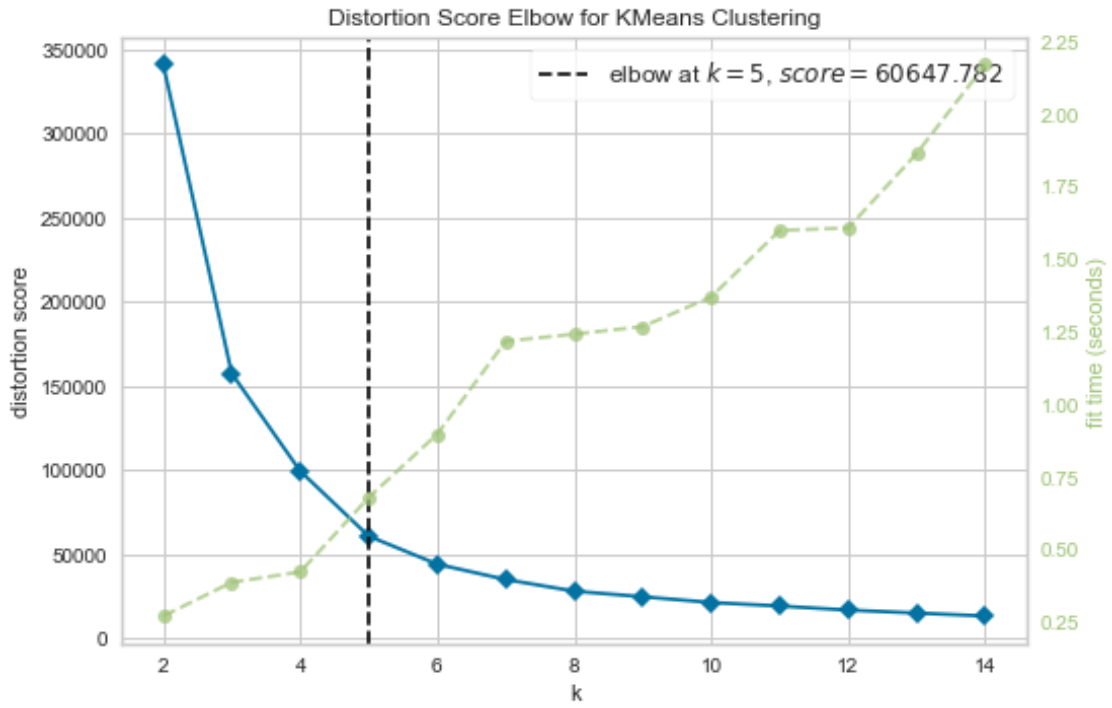
	AVERAGE BUILDING AGE	COMMUNITY AREA NAME	CENSUS BLOCK	BUILDING TYPE	\
1	41.00	Ashburn	1.703170e+14	Residential	
5	131.00	Austin	1.703125e+14	Commercial	
7	131.00	Austin	1.703125e+14	Commercial	
8	99.00	Austin	1.703125e+14	Residential	
9	99.00	Austin	1.703125e+14	Residential	
…	…	…	…	…	
67046	0.00	Woodlawn	1.703184e+14	Residential	
67047	104.50	Woodlawn	1.703184e+14	Commercial	
67048	100.67	Woodlawn	1.703184e+14	Residential	
67049	0.00	Woodlawn	1.703184e+14	Residential	
67050	79.40	Woodlawn	1.703184e+14	Residential	

	BUILDING_SUBTYPE	KWH JANUARY 2010	KWH FEBRUARY 2010	KWH MARCH 2010	\
1	Multi 7+	7334.0	7741.0	4214.0	
5	Commercial	0.0	0.0	0.0	
7	Multi < 7	1470.0	1325.0	294.0	
8	Multi 7+	2461.0	4888.0	2893.0	
9	Multi 7+	0.0	0.0	0.0	
…	…	…	…	…	
67046	Single Family	2705.0	1318.0	1582.0	
67047	Multi < 7	1005.0	1760.0	1521.0	
67048	Multi < 7	3567.0	3031.0	2582.0	
67049	Single Family	1208.0	1055.0	1008.0	
67050	Multi < 7	2717.0	3057.0	2695.0	

	KWH APRIL 2010	KWH MAY 2010	KWH JUNE 2010	KWH JULY 2010	\
1	4284.0	2518.0	4273.0	4566.0	
5	0.0	0.0	0.0	0.0	
7	391.0	366.0	2204.0	2345.0	
8	2737.0	2350.0	3037.0	3874.0	
9	0.0	0.0	511.0	904.0	
…	…	…	…	…	
67046	1465.0	1494.0	2990.0	2449.0	
67047	1832.0	2272.0	2361.0	3018.0	
67048	2295.0	7902.0	4987.0	5773.0	
67049	1109.0	1591.0	1367.0	1569.0	

```
67050          3793.0        4237.0         5383.0         5544.0

       KWH AUGUST 2010  KWH SEPTEMBER 2010  KWH OCTOBER 2010  \
1               2787.0              3357.0            5540.0
5                819.0               619.0             416.0
7               2032.0               920.0             586.0
8               4861.0              5180.0            2984.0
9               1818.0              1968.0             738.0
…                  …                   …                 …
67046           2351.0              1213.0            2174.0
67047           3030.0              2886.0            3833.0
67048           3996.0              3050.0            3103.0
67049           1551.0              1376.0            1236.0
67050           6929.0              5280.0            5971.0

       KWH NOVEMBER 2010  KWH DECEMBER 2010
1                15774.0            19676.0
5                  138.0                2.0
7                  705.0              785.0
8                 2635.0             3597.0
9                  450.0             2207.0
…                    …                  …
67046             2888.0             5025.0
67047             6290.0            12169.0
67048             3880.0             4684.0
67049             2108.0             2529.0
67050             6986.0             5144.0

[66103 rows x 17 columns]
```

[152]:
```
kmax = 15
k_values = (2, kmax)

scaler = preprocessing.StandardScaler()
norm_data = scaler.fit_transform(energy_building_age[dl.energy_cols])

km = KMeansCluster(norm_data)
km.cluster(k_values)

path = images_path / Path("ensayos/clustering_por_edad_edificio/energia")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("elbow.svg"))
```



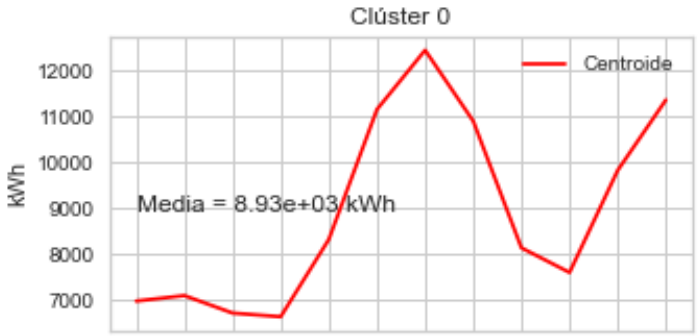
<Figure size 576x396 with 0 Axes>

[153]:
```
km_best = km.get_kmeans_of(5)
km_labels = km_best.labels_
km_centroids = scaler.inverse_transform(km_best.cluster_centers_)

np.unique(km_labels, return_counts=True)
```

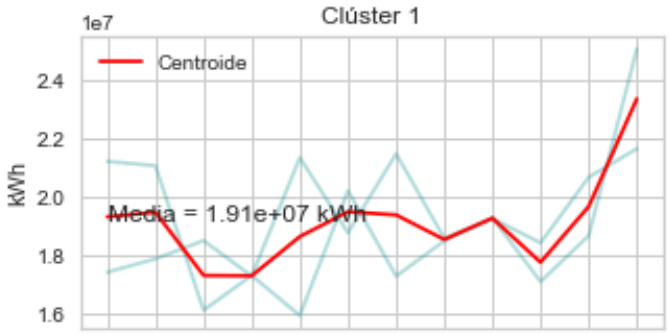
[153]:
```
(array([0, 1, 2, 3, 4]),
 array([65812,     2,    53,     6,   230], dtype=int64))
```

[154]:
```
labels = pd.DataFrame(km_labels, columns=["label"])

k = km_centroids.shape[0]

f, axes = plt.subplots(k, 1, figsize=(5, 15), sharex=True)

for ki in range(k):
    cluster = energy_building_age.iloc[labels.query(f'label == {ki}').index][dl.
 ↪energy_cols]

    palette={i:"darkcyan" for i in range(cluster.shape[0])}
```

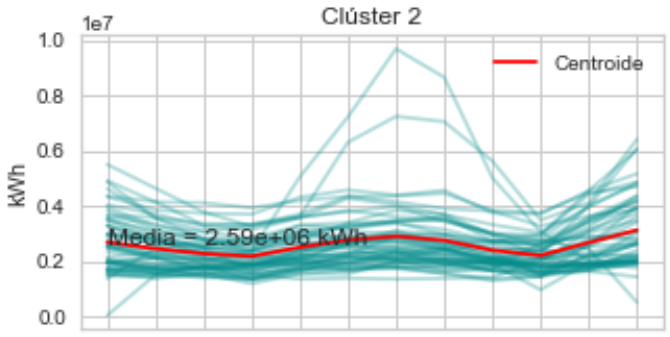
```
    if len(cluster) < 10000: #para reducir el tiempo de computo no mostramos␣
↪para un máximo de muestras
        ax0 = sns.lineplot(data=cluster.values.T, ax=axes[ki], palette=palette,␣
↪alpha=0.3)
        _ = [line.set_linestyle("-") for line in ax0.lines]

    ax0c = sns.lineplot(data=km_centroids[ki, :].T, ax=axes[ki], color="red")
    _ = [line.set_linestyle("-") for line in ax0c.lines]

    ax0c.legend(ax0c.lines[-1:], ["Centroide"]);

    axes[ki].set_title(f"Clúster {ki}")
    Media = km_centroids[ki, :].mean()
    axes[ki].text(0, Media, f"{Media = :0.2e} kWh")

    axes[ki].set_xticks([e for e in range(0, 12)])
    axes[ki].set_xticklabels([f"{datetime.date(2008, i, 1).strftime('%B').
↪upper()}" for i in range(1, 13)], rotation=90)
    axes[ki].set(ylabel="kWh")
    #print(f"Gráfica {ki} realizada")

    cluster_age = energy_building_age.iloc[labels.query(f'label == {ki}').
↪index]["AVERAGE BUILDING AGE"]

    txt = f"Edad máxima: {cluster_age.max(): 0.2f} años\n" + \
          f"Edad mínima: {cluster_age.min(): 0.2f} años\n" + \
          f"Edad media: {cluster_age.mean(): 0.2f} años\n"
    axes[ki].text(12, Media, txt)


path = images_path / Path("ensayos/clustering_por_edad_edificio/energia")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("grafica.svg"))
```



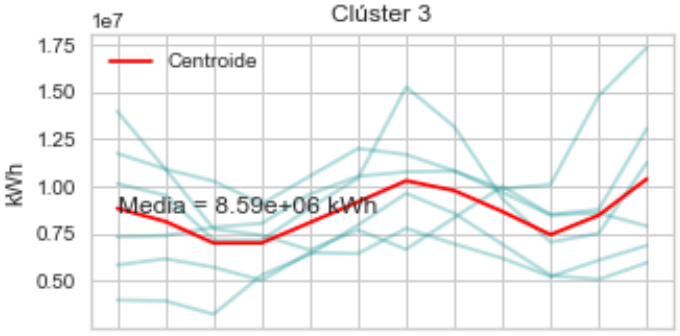
Edad máxima: 158.00 años
Edad mínima: 0.00 años
Edad media: 71.94 años



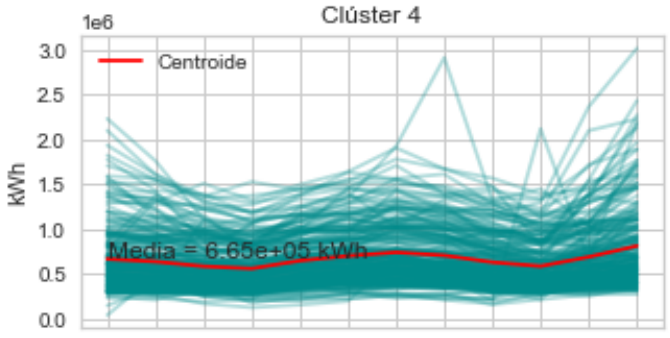
Edad máxima: 98.00 años
Edad mínima: 0.00 años
Edad media: 49.00 años



Edad máxima: 104.00 años
Edad mínima: 0.00 años
Edad media: 33.69 años



Edad máxima: 70.50 años
Edad mínima: 0.00 años
Edad media: 27.95 años



Edad máxima: 123.00 años
Edad mínima: 0.00 años
Edad media: 35.55 años

[ ]:
```
comunities_by_cluster = {}

for nlab in range(k):
    com_index = np.where(labels==nlab)[0]
    com_list = []
    for c in energy_building_age.values[com_index][:,1:5]:
        com_list.append(c)
    comunities_by_cluster[nlab] = com_list

for k in sorted(comunities_by_cluster, key=lambda k:␣
 ↪len(comunities_by_cluster[k])):
    print(f"Al clúster {k} pertenecen las comunidades:")
    for e in comunities_by_cluster[k]:
        print("\t", e)
```

Gas Aplicamos clustering a las muestras de consumo de gas.

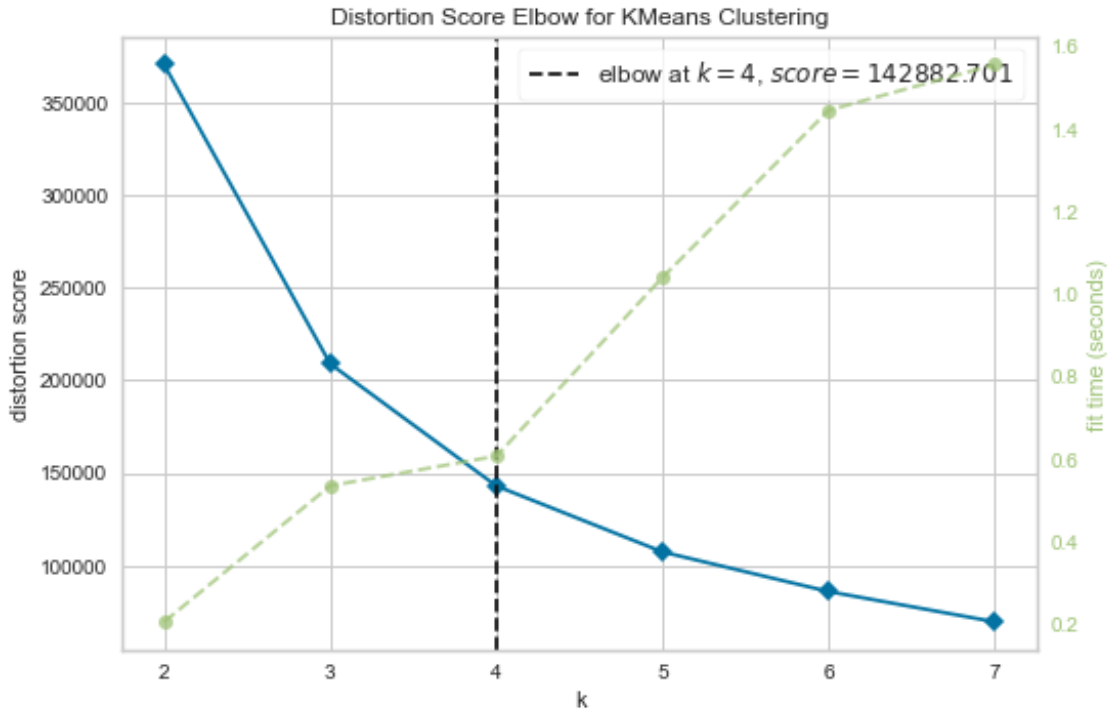
[156]:
```
gas_building_age
```

[156]:
```
       AVERAGE BUILDING AGE COMMUNITY AREA NAME  CENSUS BLOCK BUILDING TYPE  \
0                     71.33      Archer Heights  1.703157e+14   Residential
6                     54.00              Austin  1.703125e+14    Commercial
11                    87.00              Austin  1.703125e+14    Commercial
13                     0.00            Avondale  1.703121e+14    Commercial
19                   101.00            Avondale  1.703121e+14   Residential
…                       …                   …             …             …
67046                  0.00            Woodlawn  1.703184e+14   Residential
67047                104.50            Woodlawn  1.703184e+14    Commercial
67048                100.67            Woodlawn  1.703184e+14   Residential
67049                  0.00            Woodlawn  1.703184e+14   Residential
67050                 79.40            Woodlawn  1.703184e+14   Residential

      BUILDING_SUBTYPE  THERM JANUARY 2010  THERM FEBRUARY 2010  \
0            Multi < 7              2326.0               2131.0
6           Commercial              3041.0               2680.0
11           Multi < 7               910.0                738.0
13           Multi < 7               535.0                458.0
19           Multi < 7               285.0                285.0
…                  …                   …                    …
67046   Single Family              2166.0               1681.0
67047        Multi < 7               985.0               1152.0
67048        Multi < 7              2202.0               1874.0
67049   Single Family                95.0                 11.0
67050        Multi < 7              2372.0               1787.0
```

```
       THERM MARCH 2010  THERM APRIL 2010  THERM MAY 2010  THERM JUNE 2010  \
0                1400.0             620.0           502.0            224.0
6                1151.0             373.0           124.0             26.0
11                632.0             448.0           254.0            113.0
13                481.0             307.0           194.0            145.0
19                244.0             129.0            73.0             46.0
…                   …                 …               …                …
67046            1858.0            1172.0           708.0            360.0
67047            1238.0             630.0           475.0            192.0
67048            1647.0             906.0           645.0            346.0
67049              47.0               9.0            45.0             18.0
67050            1449.0             718.0           572.0            286.0

       THERM JULY 2010  THERM AUGUST 2010  THERM SEPTEMBER 2010  \
0                222.0              187.0                 197.0
6                 29.0               25.0                  49.0
11                42.0               29.0                  33.0
13               119.0              102.0                  88.0
19                39.0               27.0                  39.0
…                   …                  …                     …
67046             72.0               67.0                  77.0
67047            141.0              162.0                 144.0
67048             84.0              150.0                 150.0
67049             22.0                9.0                  17.0
67050            155.0              134.0                 161.0

       THERM OCTOBER 2010  THERM NOVEMBER 2010  THERM DECEMBER 2010
0                   252.0                744.0               2112.0
6                   177.0                670.0               3895.0
11                   36.0                166.0                633.0
13                  109.0                141.0                249.0
19                   29.0                 50.0                144.0
…                     …                    …                   …
67046               185.0                623.0               1800.0
67047               210.0                653.0               1744.0
67048               260.0                694.0               1335.0
67049                11.0                 18.0                 13.0
67050               303.0                588.0               1469.0

[60736 rows x 17 columns]
```

[157]:
```
kmax = 8
k_values = (2, kmax)

scaler = preprocessing.StandardScaler()
norm_data = scaler.fit_transform(gas_building_age[dl.gas_cols])
```

```
km = KMeansCluster(norm_data)
km.cluster(k_values)

path = images_path / Path("ensayos/clustering_por_edad_edificio/gas")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("elbow.svg"))
```



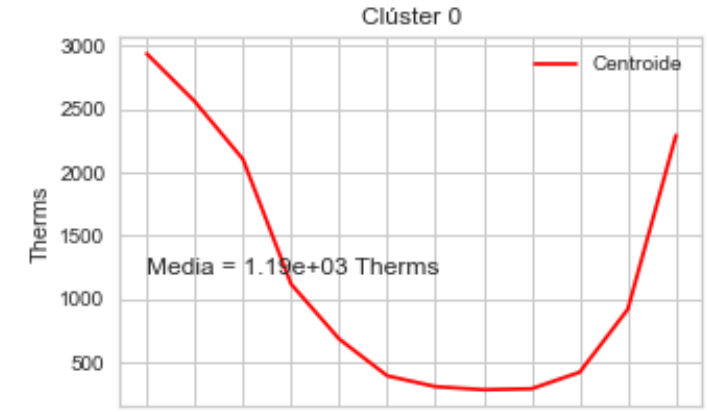
<Figure size 576x396 with 0 Axes>

[158]:
```
km_best = km.get_kmeans_of(4)
km_labels = km_best.labels_
km_centroids = scaler.inverse_transform(km_best.cluster_centers_)

np.unique(km_labels, return_counts=True)
```

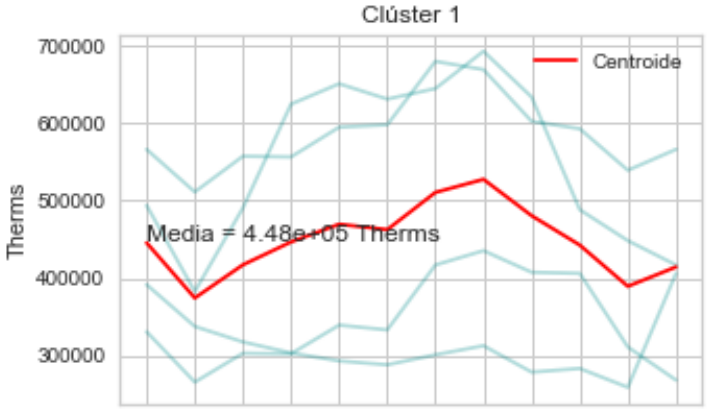
[158]: (array([0, 1, 2, 3]), array([60314,   4,   33,   385], dtype=int64))

[159]:
```
labels = pd.DataFrame(km_labels, columns=["label"])

k = km_centroids.shape[0]

f, axes = plt.subplots(k, 1, figsize=(5, 15), sharex=True)
```

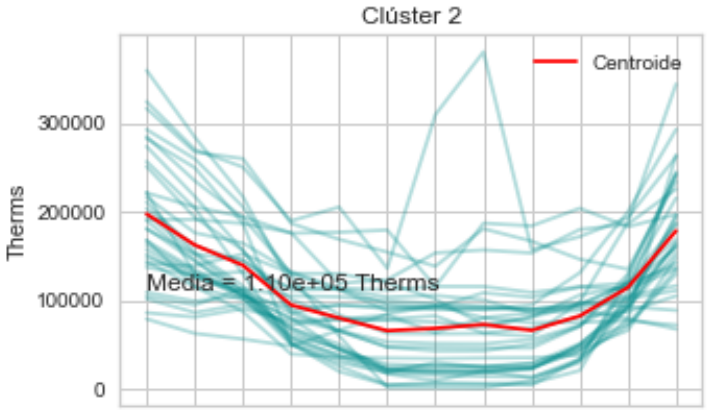
```
for ki in range(k):
    cluster = gas_building_age.iloc[labels.query(f'label == {ki}').index][dl.
 ↪gas_cols]

    palette={i:"darkcyan" for i in range(cluster.shape[0])}

    if len(cluster) < 10000:
        ax0 = sns.lineplot(data=cluster.values.T, ax=axes[ki], palette=palette,␣
 ↪alpha=0.3)
        _ = [line.set_linestyle("-") for line in ax0.lines]

    ax0c = sns.lineplot(data=km_centroids[ki, :].T, ax=axes[ki], color="red")
    _ = [line.set_linestyle("-") for line in ax0c.lines]

    ax0c.legend(ax0c.lines[-1:], ["Centroide"]);

    axes[ki].set_title(f"Clúster {ki}")
    Media = km_centroids[ki, :].mean()
    axes[ki].text(0, Media, f"{Media = :0.2e} Therms")

    axes[ki].set_xticks([e for e in range(0, 12)])
    axes[ki].set_xticklabels([f"{datetime.date(2008, i, 1).strftime('%B').
 ↪upper()}" for i in range(1, 13)], rotation=90)
    axes[ki].set(ylabel="Therms")
    #print(f"Gráfica {ki} realizada")

    cluster_age = energy_building_age.iloc[labels.query(f'label == {ki}').
 ↪index]["AVERAGE BUILDING AGE"]

    txt = f"Edad máxima: {cluster_age.max(): 0.2f} años\n" + \
          f"Edad mínima: {cluster_age.min(): 0.2f} años\n" + \
          f"Edad media: {cluster_age.mean(): 0.2f} años\n"
    axes[ki].text(12, Media, txt)


path = images_path / Path("ensayos/clustering_por_edad_edificio/gas")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("grafica.svg"))
```



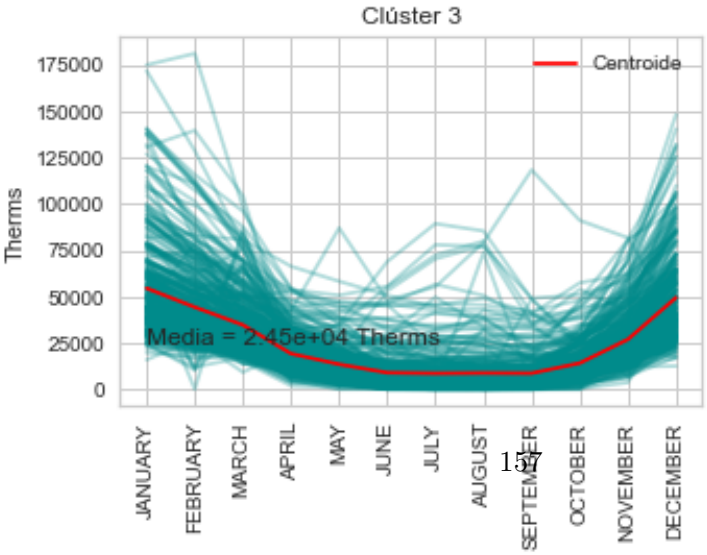
Edad máxima: 158.00 años
Edad mínima: 0.00 años
Edad media: 71.96 años



Edad máxima: 117.33 años
Edad mínima: 0.00 años
Edad media: 68.39 años



Edad máxima: 125.20 años
Edad mínima: 0.00 años
Edad media: 74.39 años



Edad máxima: 146.25 años
Edad mínima: 0.00 años
Edad media: 73.63 años

[ ]:
```
comunities_by_cluster = {}

for nlab in range(k):
    com_index = np.where(labels==nlab)[0]
    com_list = []
    for c in gas_building_age.values[com_index][:,1:5]:
        com_list.append(c)
    comunities_by_cluster[nlab] = com_list

for k in sorted(comunities_by_cluster, key=lambda k:␣
 ↪len(comunities_by_cluster[k])):
    print(f"Al clúster {k} pertenecen las comunidades:")
    for e in comunities_by_cluster[k]:
        print("\t", e)
```

1.3.14 Volver al inicio

1.3.15 Clustering por dimensión del hogar

En este caso vamos a buscar patrones según la dimensión del hogar (número de personas entre número de casas). En este caso, solo se seleccionarán las zonas residenciales.

[161]:
```
housesize = dl[['AVERAGE HOUSESIZE', 'BUILDING TYPE']]
housesize = housesize[housesize['BUILDING TYPE'] == "Residential"]
housesize
```

[161]:

	AVERAGE HOUSESIZE	BUILDING TYPE
0	3.87	Residential
1	1.81	Residential
8	2.93	Residential
9	3.82	Residential
10	0.00	Residential
…	…	…
67045	3.14	Residential
67046	3.14	Residential
67048	2.07	Residential
67049	0.00	Residential
67050	2.57	Residential

[49740 rows x 2 columns]

[162]:
```
minsize = housesize['AVERAGE HOUSESIZE'].min().item()
maxsize = housesize['AVERAGE HOUSESIZE'].max().item()
```

```
f'El valor de dimensión del hogar oscila entre {minsize} y {maxsize}'
```

[162]:
```
'El valor de dimensión del hogar oscila entre 0.0 y 9.0'
```

Energia Aplicamos clustering a las muestras de consumo de electricidad.

[163]:
```
energy_housesize = dl[dl.energy_cols + ['AVERAGE HOUSESIZE', 'BUILDING TYPE',␣
↪'COMMUNITY AREA NAME']]
energy_housesize = energy_housesize[energy_housesize['BUILDING TYPE'] ==␣
↪"Residential"][dl.energy_cols + ['AVERAGE HOUSESIZE', 'COMMUNITY AREA NAME']]
energy_housesize
```

[163]:

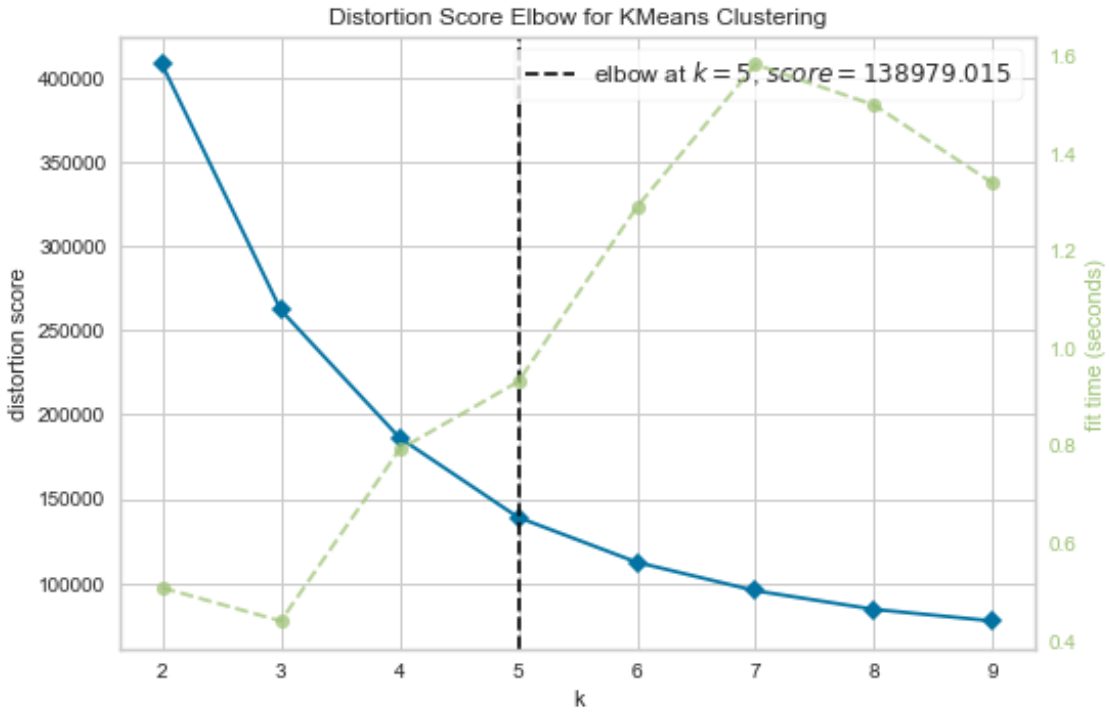
	KWH JANUARY 2010	KWH FEBRUARY 2010	KWH MARCH 2010	KWH APRIL 2010	\
1	7334.0	7741.0	4214.0	4284.0	
8	2461.0	4888.0	2893.0	2737.0	
9	0.0	0.0	0.0	0.0	
10	96.0	202.0	1837.0	1118.0	
29	7542.0	7912.0	5451.0	3173.0	
…	…	…	…	…	
67045	9572.0	9104.0	8525.0	7756.0	
67046	2705.0	1318.0	1582.0	1465.0	
67048	3567.0	3031.0	2582.0	2295.0	
67049	1208.0	1055.0	1008.0	1109.0	
67050	2717.0	3057.0	2695.0	3793.0	

	KWH MAY 2010	KWH JUNE 2010	KWH JULY 2010	KWH AUGUST 2010	\
1	2518.0	4273.0	4566.0	2787.0	
8	2350.0	3037.0	3874.0	4861.0	
9	0.0	511.0	904.0	1818.0	
10	669.0	889.0	812.0	1880.0	
29	3792.0	4120.0	3197.0	4574.0	
…	…	…	…	…	
67045	11256.0	11669.0	12099.0	13200.0	
67046	1494.0	2990.0	2449.0	2351.0	
67048	7902.0	4987.0	5773.0	3996.0	
67049	1591.0	1367.0	1569.0	1551.0	
67050	4237.0	5383.0	5544.0	6929.0	

	KWH SEPTEMBER 2010	KWH OCTOBER 2010	KWH NOVEMBER 2010	\
1	3357.0	5540.0	15774.0	
8	5180.0	2984.0	2635.0	
9	1968.0	738.0	450.0	
10	1743.0	696.0	684.0	
29	5509.0	6531.0	14402.0	
…	…	…	…	
67045	9694.0	8419.0	19077.0	

```
67046              1213.0            2174.0             2888.0
67048              3050.0            3103.0             3880.0
67049              1376.0            1236.0             2108.0
67050              5280.0            5971.0             6986.0

       KWH DECEMBER 2010  AVERAGE HOUSESIZE COMMUNITY AREA NAME
1                19676.0               1.81              Ashburn
8                 3597.0               2.93               Austin
9                 2207.0               3.82               Austin
10                 903.0               0.00               Austin
29               22510.0               2.40              Chatham
…                    …                  …                    …
67045            18869.0               3.14            Woodlawn
67046             5025.0               3.14            Woodlawn
67048             4684.0               2.07            Woodlawn
67049             2529.0               0.00            Woodlawn
67050             5144.0               2.57            Woodlawn

[49440 rows x 14 columns]
```

[164]:
```
kmax = 10
k_values = (2, kmax)

scaler = preprocessing.StandardScaler()
norm_data = scaler.fit_transform(energy_housesize[dl.energy_cols])

km = KMeansCluster(norm_data)
km.cluster(k_values)

path = images_path / Path("ensayos/clustering_por_dimension_hogar/energia")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("elbow.svg"))
```



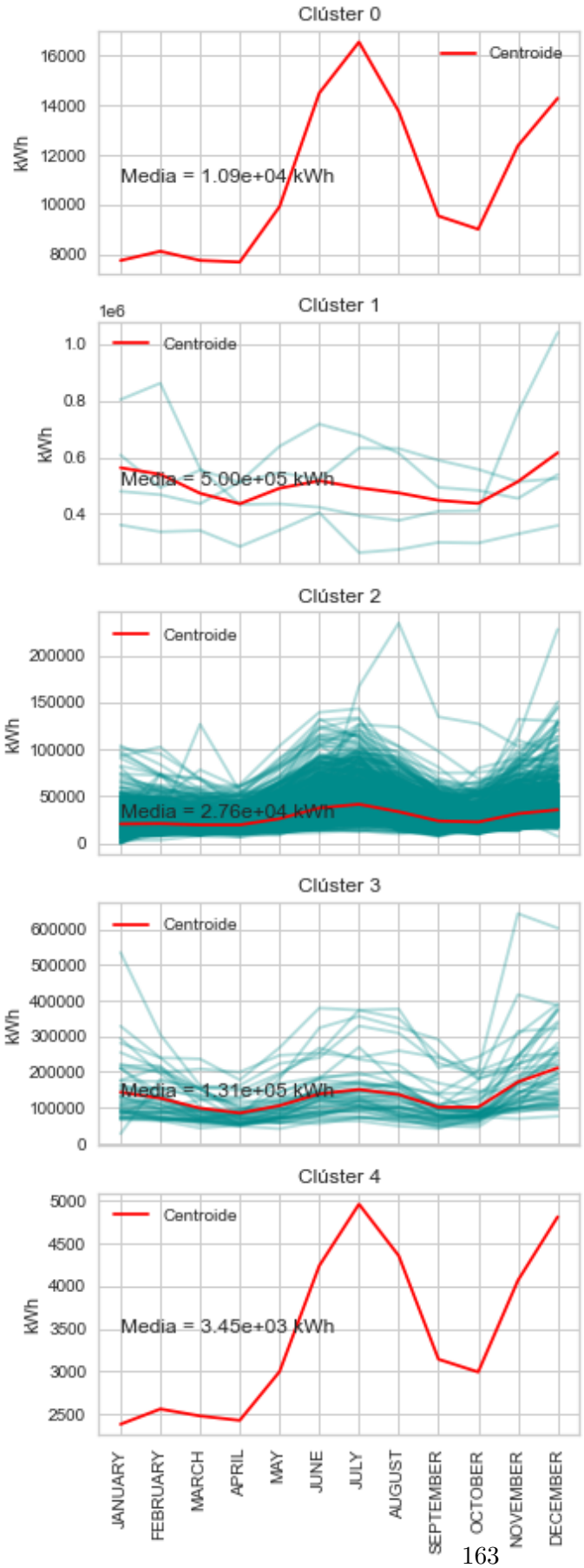
<Figure size 576x396 with 0 Axes>

[165]:
```
km_best = km.get_kmeans_of(5)
km_labels = km_best.labels_
km_centroids = scaler.inverse_transform(km_best.cluster_centers_)

np.unique(km_labels, return_counts=True)
```

[165]:
```
(array([0, 1, 2, 3, 4]),
 array([17752,     4,  1473,    46, 30165], dtype=int64))
```

[166]:
```
labels = pd.DataFrame(km_labels, columns=["label"])

k = km_centroids.shape[0]

f, axes = plt.subplots(k, 1, figsize=(5, 15), sharex=True)

for ki in range(k):
    cluster = energy_housesize.iloc[labels.query(f'label == {ki}').index][dl.
 ↪energy_cols]

    palette={i:"darkcyan" for i in range(cluster.shape[0])}

    if len(cluster) < 10000:
```

```
        ax0 = sns.lineplot(data=cluster.values.T, ax=axes[ki], palette=palette,␣
↪alpha=0.3)
        _ = [line.set_linestyle("-") for line in ax0.lines]

    ax0c = sns.lineplot(data=km_centroids[ki, :].T, ax=axes[ki], color="red")
    _ = [line.set_linestyle("-") for line in ax0c.lines]

    ax0c.legend(ax0c.lines[-1:], ["Centroide"]);

    axes[ki].set_title(f"Clúster {ki}")
    Media = km_centroids[ki, :].mean()
    axes[ki].text(0, Media, f"{Media = :0.2e} kWh")

    axes[ki].set_xticks([e for e in range(0, 12)])
    axes[ki].set_xticklabels([f"{datetime.date(2008, i, 1).strftime('%B').
↪upper()}" for i in range(1, 13)], rotation=90)
    axes[ki].set(ylabel="kWh")
    #print(f"Gráfica {ki} realizada")

    cluster_housesize = energy_housesize.iloc[labels.query(f'label == {ki}').
↪index]["AVERAGE HOUSESIZE"]

    txt = f"Max housesize: {cluster_housesize.max()}\n" + \
          f"Min housesize: {cluster_housesize.min()}\n" + \
          f"Mean housesize:{cluster_housesize.mean()}"

    axes[ki].text(12, Media, txt)

path = images_path / Path("ensayos/clustering_por_dimension_hogar/energia")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("grafica.svg"))
```



Max housesize: 9.0
Min housesize: 0.0
Mean housesize:2.9584751013970254

Max housesize: 3.16
Min housesize: 0.0
Mean housesize:2.0475000000000003

Max housesize: 6.05
Min housesize: 0.0
Mean housesize:2.6256143923964697

Max housesize: 3.73
Min housesize: 1.11
Mean housesize:1.8167391304347829

Max housesize: 9.0
Min housesize: 0.0
Mean housesize:2.8359827614785353

[ ]:
```
comunities_by_cluster = {}

for nlab in range(k):
    com_index = np.where(labels==nlab)[0]
    com_list = []
    for c in energy_housesize.values[com_index][:,-1]:
        com_list.append(c)
    comunities_by_cluster[nlab] = com_list

for k in sorted(comunities_by_cluster, key=lambda k:␣
↪len(comunities_by_cluster[k])):
    print(f"Al clúster {k} pertenecen las comunidades:")
    for e in np.unique(comunities_by_cluster[k]):
        print("\t", e)
```

Gas Aplicamos clustering a las muestras de consumo de gas.

[168]:
```
gas_housesize = dl[dl.gas_cols + ['AVERAGE HOUSESIZE', 'BUILDING TYPE',␣
↪'COMMUNITY AREA NAME']]
gas_housesize = gas_housesize[gas_housesize['BUILDING TYPE'] ==␣
↪"Residential"][dl.gas_cols + ['AVERAGE HOUSESIZE', 'COMMUNITY AREA NAME']]
gas_housesize
```

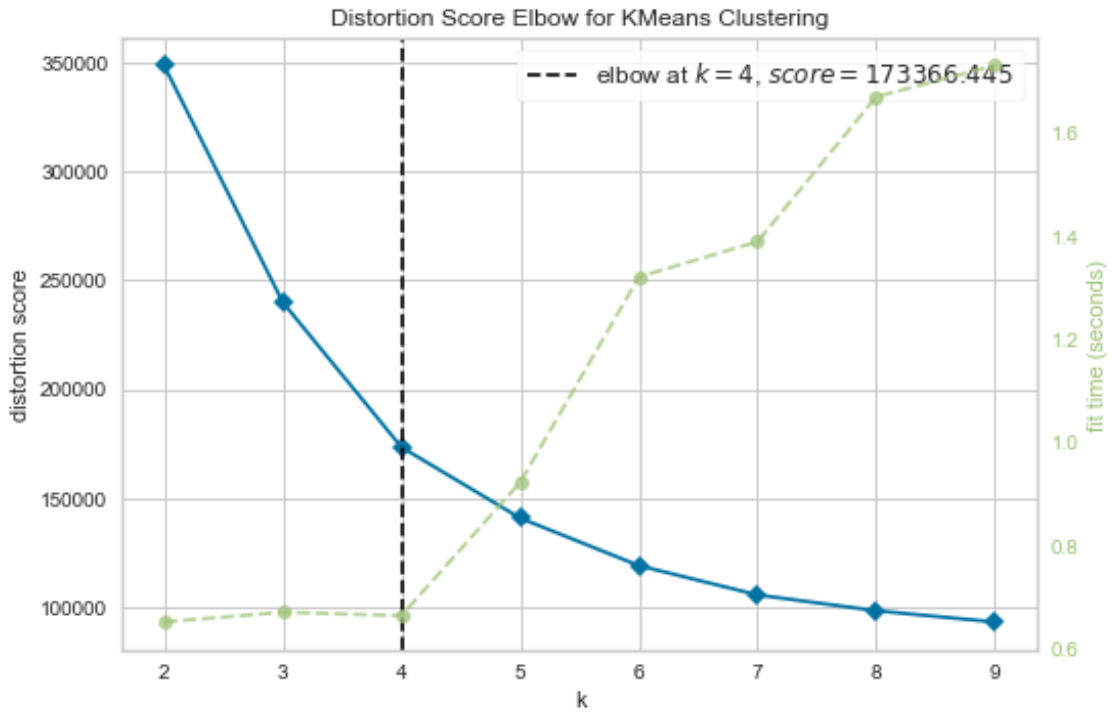
[168]:

	THERM JANUARY 2010	THERM FEBRUARY 2010	THERM MARCH 2010	\
0	2326.0	2131.0	1400.0	
19	285.0	285.0	244.0	
22	243.0	195.0	156.0	
39	555.0	440.0	378.0	
41	326.0	305.0	244.0	
…	…	…	…	
67045	6914.0	5433.0	5054.0	
67046	2166.0	1681.0	1858.0	
67048	2202.0	1874.0	1647.0	
67049	95.0	11.0	47.0	
67050	2372.0	1787.0	1449.0	

	THERM APRIL 2010	THERM MAY 2010	THERM JUNE 2010	THERM JULY 2010	\
0	620.0	502.0	224.0	222.0	
19	129.0	73.0	46.0	39.0	
22	46.0	6.0	7.0	6.0	
39	228.0	126.0	88.0	61.0	
41	137.0	96.0	76.0	44.0	
…	…	…	…	…	
67045	2967.0	2241.0	1107.0	770.0	

```
67046            1172.0          708.0           360.0            72.0
67048             906.0          645.0           346.0            84.0
67049               9.0           45.0            18.0            22.0
67050             718.0          572.0           286.0           155.0

       THERM AUGUST 2010  THERM SEPTEMBER 2010  THERM OCTOBER 2010  \
0                  187.0                 197.0               252.0
19                  27.0                  39.0                29.0
22                   6.0                   7.0                 5.0
39                  62.0                  63.0                69.0
41                  27.0                  31.0                48.0
…                    …                     …                   …
67045              674.0                 788.0               954.0
67046               67.0                  77.0               185.0
67048              150.0                 150.0               260.0
67049                9.0                  17.0                11.0
67050              134.0                 161.0               303.0

       THERM NOVEMBER 2010  THERM DECEMBER 2010  AVERAGE HOUSESIZE  \
0                    744.0                2112.0               3.87
19                    50.0                 144.0               2.20
22                    26.0                  81.0               2.00
39                   173.0                 304.0               2.61
41                    71.0                 205.0               3.19
…                      …                    …                   …
67045               2423.0                4619.0               3.14
67046                623.0                1800.0               3.14
67048                694.0                1335.0               2.07
67049                 18.0                  13.0               0.00
67050                588.0                1469.0               2.57

      COMMUNITY AREA NAME
0          Archer Heights
19               Avondale
22                Beverly
39               Clearing
41                Dunning
…                    …
67045            Woodlawn
67046            Woodlawn
67048            Woodlawn
67049            Woodlawn
67050            Woodlawn

[46195 rows x 14 columns]
```

[169]:
```
kmax = 10
k_values = (2, kmax)

scaler = preprocessing.StandardScaler()
norm_data = scaler.fit_transform(gas_housesize[dl.gas_cols])

km = KMeansCluster(norm_data)
km.cluster(k_values)

path = images_path / Path("ensayos/clustering_por_dimension_hogar/gas")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("elbow.svg"))
```

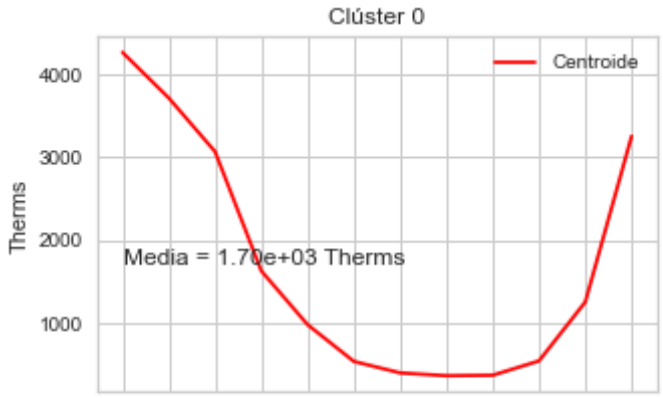


<Figure size 576x396 with 0 Axes>

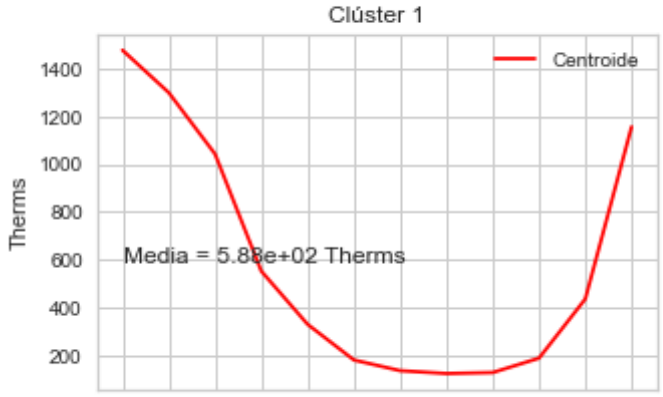
[170]:
```
km_best = km.get_kmeans_of(4)
km_labels = km_best.labels_
km_centroids = scaler.inverse_transform(km_best.cluster_centers_)

np.unique(km_labels, return_counts=True)
```

[170]: (array([0, 1, 2, 3]), array([17451, 26212,   68, 2464], dtype=int64))

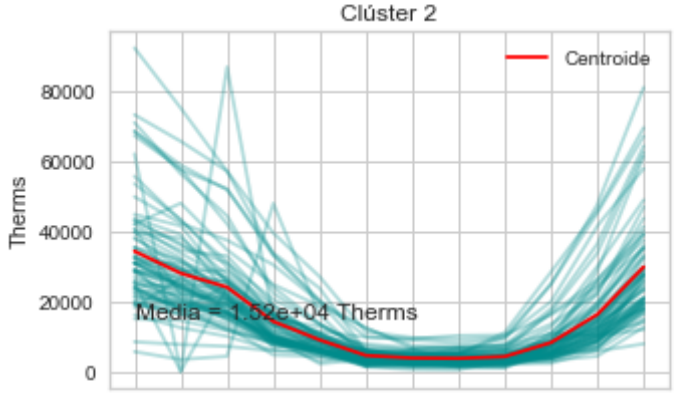
[171]:
```
labels = pd.DataFrame(km_labels, columns=["label"])

k = km_centroids.shape[0]

f, axes = plt.subplots(k, 1, figsize=(5, 15), sharex=True)

for ki in range(k):
    cluster = gas_housesize.iloc[labels.query(f'label == {ki}').index][dl.
 ↪gas_cols]

    palette={i:"darkcyan" for i in range(cluster.shape[0])}

    if len(cluster) < 10000:
        ax0 = sns.lineplot(data=cluster.values.T, ax=axes[ki], palette=palette,␣
 ↪alpha=0.3)
        _ = [line.set_linestyle("-") for line in ax0.lines]

    ax0c = sns.lineplot(data=km_centroids[ki, :].T, ax=axes[ki], color="red")
    _ = [line.set_linestyle("-") for line in ax0c.lines]

    ax0c.legend(ax0c.lines[-1:], ["Centroide"]);

    axes[ki].set_title(f"Clúster {ki}")
    Media = km_centroids[ki, :].mean()
    axes[ki].text(0, Media, f"{Media = :0.2e} Therms")

    axes[ki].set_xticks([e for e in range(0, 12)])
    axes[ki].set_xticklabels([f"{datetime.date(2008, i, 1).strftime('%B').
 ↪upper()}" for i in range(1, 13)], rotation=90)
    axes[ki].set(ylabel="Therms")
    #print(f"Gráfica {ki} realizada")

    cluster_housesize = gas_housesize.iloc[labels.query(f'label == {ki}').
 ↪index]["AVERAGE HOUSESIZE"]

    txt = f"Max housesize: {cluster_housesize.max()}\n" + \
          f"Min housesize: {cluster_housesize.min()}\n" + \
          f"Mean housesize:{cluster_housesize.mean()}"

    axes[ki].text(12, Media, txt)

path = images_path / Path("ensayos/clustering_por_dimension_hogar/gas")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("grafica.svg"))
```



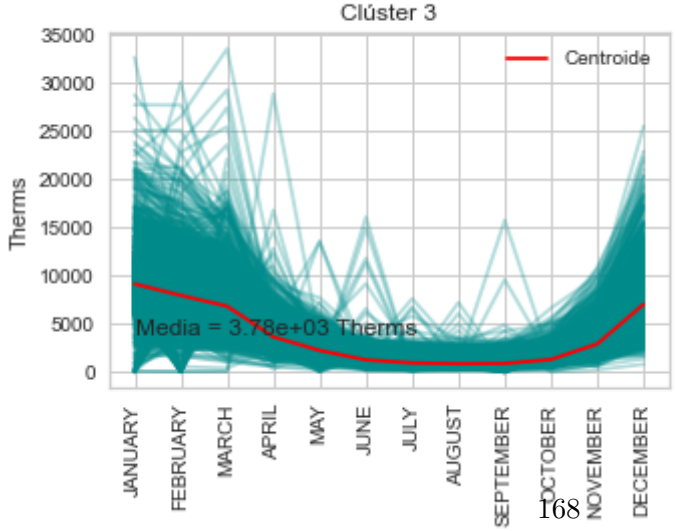
Max housesize: 9.0
Min housesize: 0.0
Mean housesize:2.998601799323821



Max housesize: 9.0
Min housesize: 0.0
Mean housesize:2.8323390050358617



Max housesize: 6.33
Min housesize: 0.0
Mean housesize:2.0235294117647062



Max housesize: 5.59
Min housesize: 0.0
Mean housesize:2.6699959415584416

[ ]:
```
comunities_by_cluster = {}

for nlab in range(k):
    com_index = np.where(labels==nlab)[0]
    com_list = []
    for c in gas_housesize.values[com_index][:, -1]:
        com_list.append(c)
    comunities_by_cluster[nlab] = com_list

for k in sorted(comunities_by_cluster, key=lambda k:␣
↪len(comunities_by_cluster[k])):
    print(f"Al clúster {k} pertenecen las comunidades:")
    for e in np.unique(comunities_by_cluster[k]):
        print("\t", e)
```

1.3.16 Volver al inicio

1.3.17 Clustering por ocupación total

En este capo se buscarán patrones según el porcentaje de ocupación para las zonas residenciales.

Energía Seleccionamos las muestras de consumo de electricidad para las zonas residenciales y aplicamos clustering.

[6]:
```
energy_occupied = dl[dl.energy_cols + ['OCCUPIED UNITS PERCENTAGE', 'BUILDING␣
↪TYPE', 'COMMUNITY AREA NAME']]
energy_occupied = energy_occupied[energy_occupied['BUILDING TYPE'] ==␣
↪"Residential"][dl.energy_cols + ['OCCUPIED UNITS PERCENTAGE', 'COMMUNITY␣
↪AREA NAME']]
energy_occupied
```

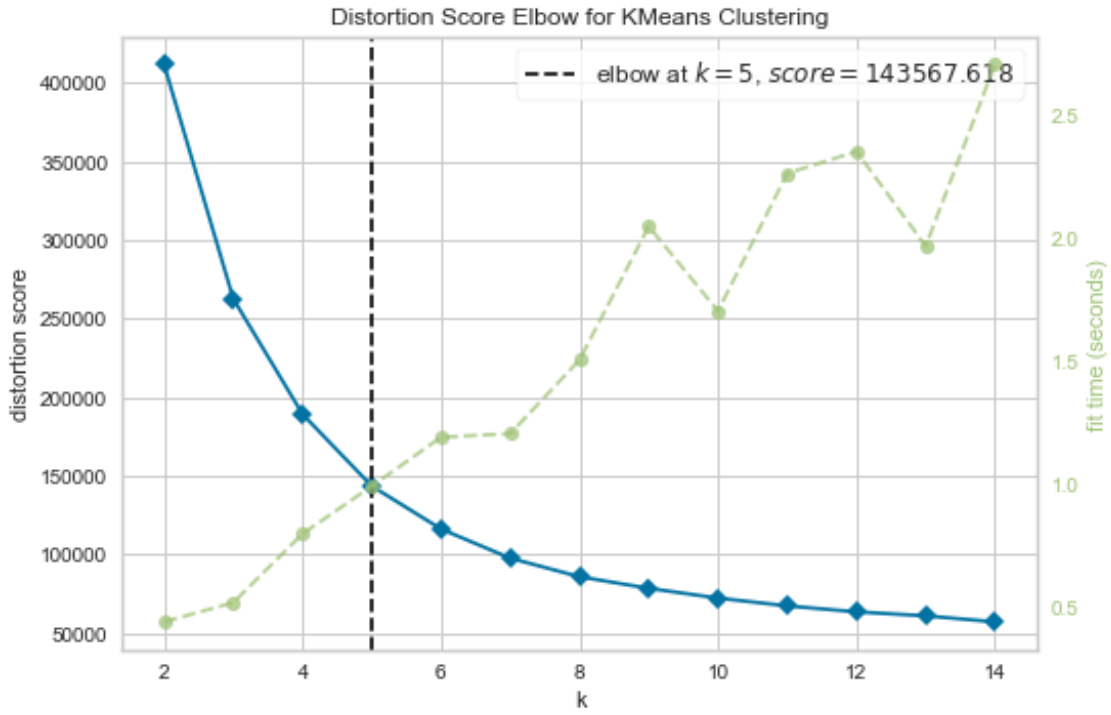
[6]:

	KWH JANUARY 2010	KWH FEBRUARY 2010	KWH MARCH 2010	KWH APRIL 2010	\
1	7334.0	7741.0	4214.0	4284.0	
8	2461.0	4888.0	2893.0	2737.0	
9	0.0	0.0	0.0	0.0	
29	7542.0	7912.0	5451.0	3173.0	
30	0.0	0.0	115.0	695.0	
…	…	…	…	…	
67043	653.0	989.0	1281.0	1110.0	
67045	9572.0	9104.0	8525.0	7756.0	
67046	2705.0	1318.0	1582.0	1465.0	
67048	3567.0	3031.0	2582.0	2295.0	
67050	2717.0	3057.0	2695.0	3793.0	

```
       KWH MAY 2010  KWH JUNE 2010  KWH JULY 2010  KWH AUGUST 2010  \
1            2518.0         4273.0         4566.0           2787.0
8            2350.0         3037.0         3874.0           4861.0
9               0.0          511.0          904.0           1818.0
29           3792.0         4120.0         3197.0           4574.0
30           3527.0         8412.0         4175.0           3900.0
…               …              …              …                …
67043        2077.0         2265.0         3694.0           3588.0
67045       11256.0        11669.0        12099.0          13200.0
67046        1494.0         2990.0         2449.0           2351.0
67048        7902.0         4987.0         5773.0           3996.0
67050        4237.0         5383.0         5544.0           6929.0

       KWH SEPTEMBER 2010  KWH OCTOBER 2010  KWH NOVEMBER 2010  \
1                  3357.0            5540.0            15774.0
8                  5180.0            2984.0             2635.0
9                  1968.0             738.0              450.0
29                 5509.0            6531.0            14402.0
30                 3257.0            3305.0             2813.0
…                     …                 …                 …
67043              3233.0            4107.0             5930.0
67045              9694.0            8419.0            19077.0
67046              1213.0            2174.0             2888.0
67048              3050.0            3103.0             3880.0
67050              5280.0            5971.0             6986.0

       KWH DECEMBER 2010  OCCUPIED UNITS PERCENTAGE COMMUNITY AREA NAME
1                19676.0                     0.9254             Ashburn
8                 3597.0                     0.8710              Austin
9                 2207.0                     0.6667              Austin
29               22510.0                     0.8955             Chatham
30                3344.0                     0.2000             Chatham
…                    …                          …                  …
67043             6640.0                     0.9000            Woodlawn
67045            18869.0                     0.6727            Woodlawn
67046             5025.0                     0.6727            Woodlawn
67048             4684.0                     0.6250            Woodlawn
67050             5144.0                     0.6122            Woodlawn

[48661 rows x 14 columns]
```

[7]:
```
kmax = 15
k_values = (2, kmax)

scaler = preprocessing.StandardScaler()
norm_data = scaler.fit_transform(energy_occupied[dl.energy_cols])
```

```
km = KMeansCluster(norm_data)
km.cluster(k_values)

path = images_path / Path("ensayos/clustering_por_ocupacion_total/energia")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("elbow.svg"))
```



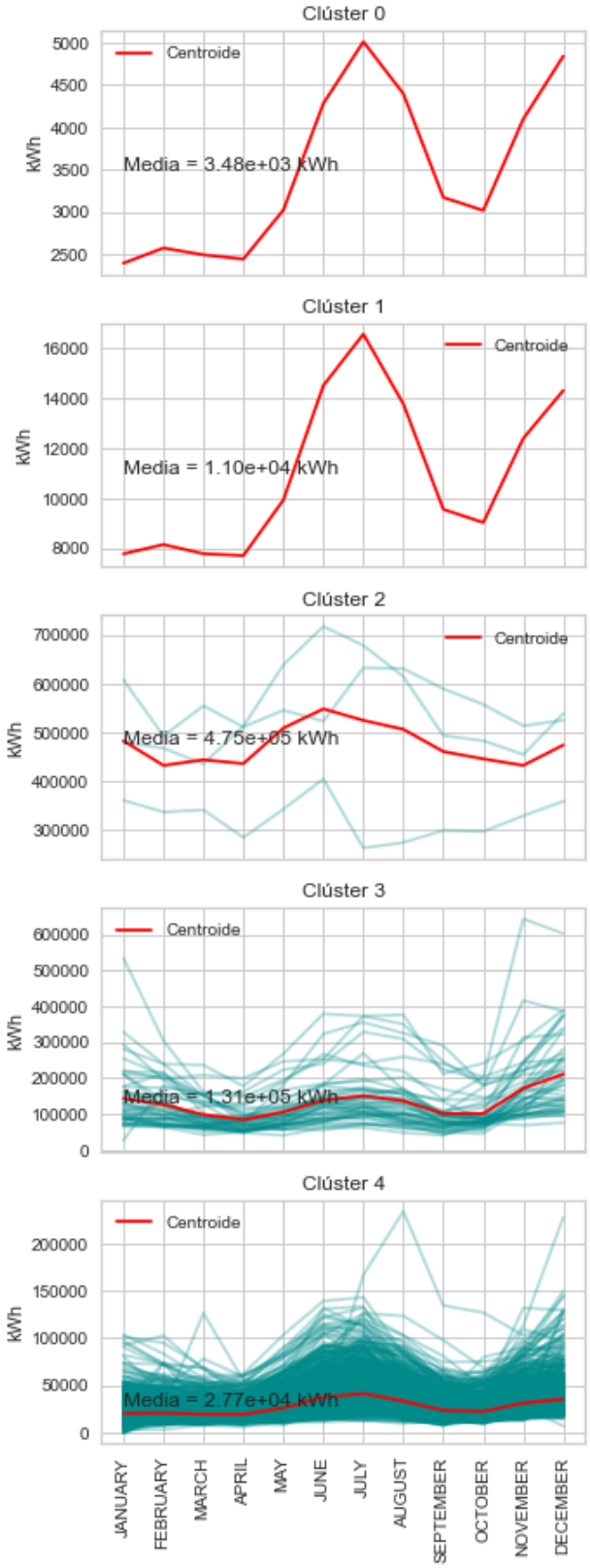

```
<Figure size 576x396 with 0 Axes>
```

[8]:
```
km_best = km.get_kmeans_of(5)
km_labels = km_best.labels_
km_centroids = scaler.inverse_transform(km_best.cluster_centers_)

np.unique(km_labels, return_counts=True)
```

[8]:
```
(array([0, 1, 2, 3, 4]),
 array([29584, 17577,     3,    46,  1451], dtype=int64))
```

[9]:
```
labels = pd.DataFrame(km_labels, columns=["label"])

k = km_centroids.shape[0]

f, axes = plt.subplots(k, 1, figsize=(5, 15), sharex=True)
```

```
for ki in range(k):
    cluster = energy_occupied.iloc[labels.query(f'label == {ki}').index][dl.
 ↪energy_cols]

    palette={i:"darkcyan" for i in range(cluster.shape[0])}

    if len(cluster) < 10000:
        ax0 = sns.lineplot(data=cluster.values.T, ax=axes[ki], palette=palette,␣
 ↪alpha=0.3)
        _ = [line.set_linestyle("-") for line in ax0.lines]

    ax0c = sns.lineplot(data=km_centroids[ki, :].T, ax=axes[ki], color="red")
    _ = [line.set_linestyle("-") for line in ax0c.lines]

    ax0c.legend(ax0c.lines[-1:], ["Centroide"]);

    axes[ki].set_title(f"Clúster {ki}")
    Media = km_centroids[ki, :].mean()
    axes[ki].text(0, Media, f"{Media = :0.2e} kWh")

    axes[ki].set_xticks([e for e in range(0, 12)])
    axes[ki].set_xticklabels([f"{datetime.date(2008, i, 1).strftime('%B').
 ↪upper()}" for i in range(1, 13)], rotation=90)
    axes[ki].set(ylabel="kWh")
    #print(f"Gráfica {ki} realizada")

    cluster_occupied = energy_occupied.iloc[labels.query(f'label == {ki}').
 ↪index]["OCCUPIED UNITS PERCENTAGE"]

    txt = f"Porcentaje ocupación máxima: {cluster_occupied.max()}\n" + \
          f"Porcentaje ocupación minimo: {cluster_occupied.min()}\n" + \
          f"Porcentaje ocupación medio:{cluster_occupied.mean()}"

    axes[ki].text(12, Media, txt)


path = images_path / Path("ensayos/clustering_por_ocupacion_total/energia")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("grafica.svg"))
```



Porcentaje ocupación máxima: 1.0
Porcentaje ocupación minimo: 0.0
Porcentaje ocupación medio:0.8651949026500811

Porcentaje ocupación máxima: 1.0
Porcentaje ocupación minimo: 0.0
Porcentaje ocupación medio:0.9162872958980486

Porcentaje ocupación máxima: 1.0
Porcentaje ocupación minimo: 0.8238
Porcentaje ocupación medio:0.9270333333333333

Porcentaje ocupación máxima: 1.0
Porcentaje ocupación minimo: 0.2669
Porcentaje ocupación medio:0.8948326086956521

Porcentaje ocupación máxima: 1.0
Porcentaje ocupación minimo: 0.0
Porcentaje ocupación medio:0.9320445210199862

[ ]:
```
comunities_by_cluster = {}

for nlab in range(k):
    com_index = np.where(labels==nlab)[0]
    com_list = []
    for c in energy_occupied.values[com_index][:, -1]:
        com_list.append(c)
    comunities_by_cluster[nlab] = com_list

for k in sorted(comunities_by_cluster, key=lambda k:␣
 ↪len(comunities_by_cluster[k])):
    print(f"Al clúster {k} pertenecen las comunidades:")
    for e in np.unique(comunities_by_cluster[k]):
        print("\t", e)
```

Gas Seleccionamos las muestras de consumo de gas para las zonas residenciales y aplicamos clustering.

[11]:
```
gas_occupied = dl[dl.gas_cols + ['OCCUPIED UNITS PERCENTAGE', 'BUILDING TYPE',␣
 ↪'COMMUNITY AREA NAME']]
gas_occupied = gas_occupied[gas_occupied['BUILDING TYPE'] == "Residential"][dl.
 ↪gas_cols + ['OCCUPIED UNITS PERCENTAGE', 'COMMUNITY AREA NAME']]
gas_occupied
```

[11]:

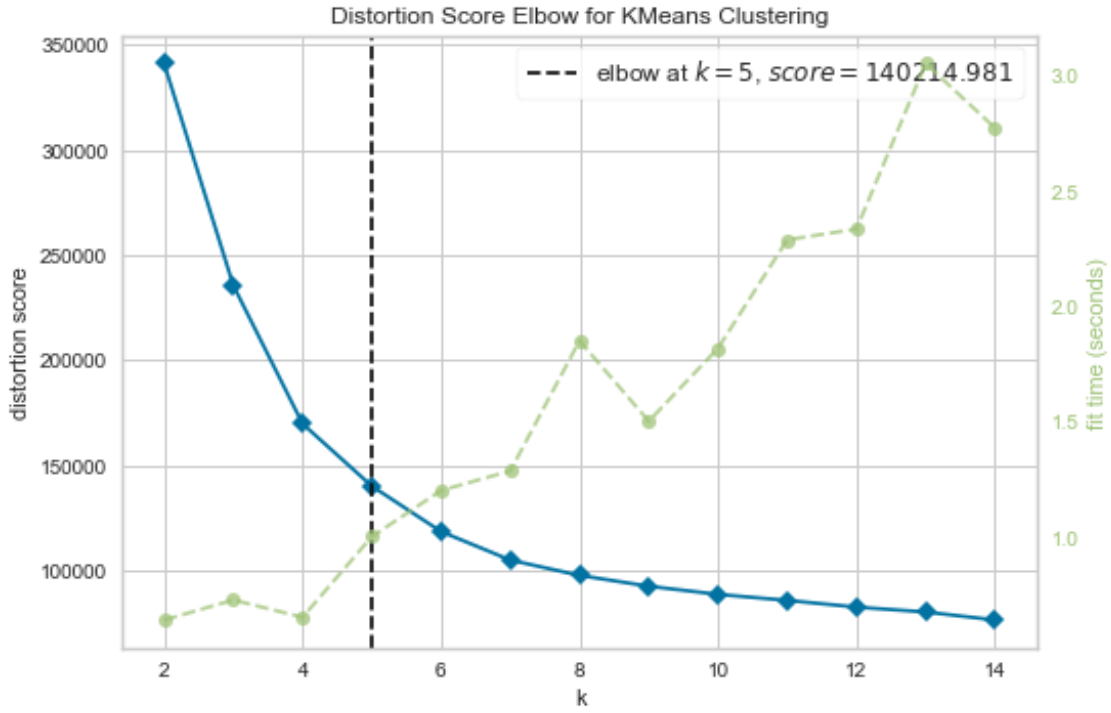
	THERM JANUARY 2010	THERM FEBRUARY 2010	THERM MARCH 2010	\
0	2326.0	2131.0	1400.0	
19	285.0	285.0	244.0	
22	243.0	195.0	156.0	
39	555.0	440.0	378.0	
41	326.0	305.0	244.0	
…	…	…	…	
67043	3299.0	1884.0	1944.0	
67045	6914.0	5433.0	5054.0	
67046	2166.0	1681.0	1858.0	
67048	2202.0	1874.0	1647.0	
67050	2372.0	1787.0	1449.0	

	THERM APRIL 2010	THERM MAY 2010	THERM JUNE 2010	THERM JULY 2010	\
0	620.0	502.0	224.0	222.0	
19	129.0	73.0	46.0	39.0	
22	46.0	6.0	7.0	6.0	
39	228.0	126.0	88.0	61.0	
41	137.0	96.0	76.0	44.0	
…	…	…	…	…	
67043	876.0	712.0	372.0	271.0	
67045	2967.0	2241.0	1107.0	770.0	
67046	1172.0	708.0	360.0	72.0	

```
67048             906.0           645.0            346.0             84.0
67050             718.0           572.0            286.0            155.0

       THERM AUGUST 2010  THERM SEPTEMBER 2010  THERM OCTOBER 2010  \
0                  187.0                 197.0               252.0
19                  27.0                  39.0                29.0
22                   6.0                   7.0                 5.0
39                  62.0                  63.0                69.0
41                  27.0                  31.0                48.0
…                    …                     …                   …
67043              222.0                 184.0               233.0
67045              674.0                 788.0               954.0
67046               67.0                  77.0               185.0
67048              150.0                 150.0               260.0
67050              134.0                 161.0               303.0

       THERM NOVEMBER 2010  THERM DECEMBER 2010  OCCUPIED UNITS PERCENTAGE  \
0                    744.0                2112.0                     0.9582
19                    50.0                 144.0                     0.9375
22                    26.0                  81.0                     1.0000
39                   173.0                 304.0                     0.9200
41                    71.0                 205.0                     1.0000
…                      …                    …                         …
67043                524.0                1135.0                     0.9000
67045               2423.0                4619.0                     0.6727
67046                623.0                1800.0                     0.6727
67048                694.0                1335.0                     0.6250
67050                588.0                1469.0                     0.6122

      COMMUNITY AREA NAME
0          Archer Heights
19               Avondale
22                Beverly
39               Clearing
41                Dunning
…                    …
67043            Woodlawn
67045            Woodlawn
67046            Woodlawn
67048            Woodlawn
67050            Woodlawn

[45514 rows x 14 columns]
```

[12]:
```
kmax = 15
k_values = (2, kmax)
```

```
scaler = preprocessing.StandardScaler()
norm_data = scaler.fit_transform(gas_occupied[dl.gas_cols])

km = KMeansCluster(norm_data)
km.cluster(k_values)

path = images_path / Path("ensayos/clustering_por_ocupacion_total/gas")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("elbow.svg"))
```



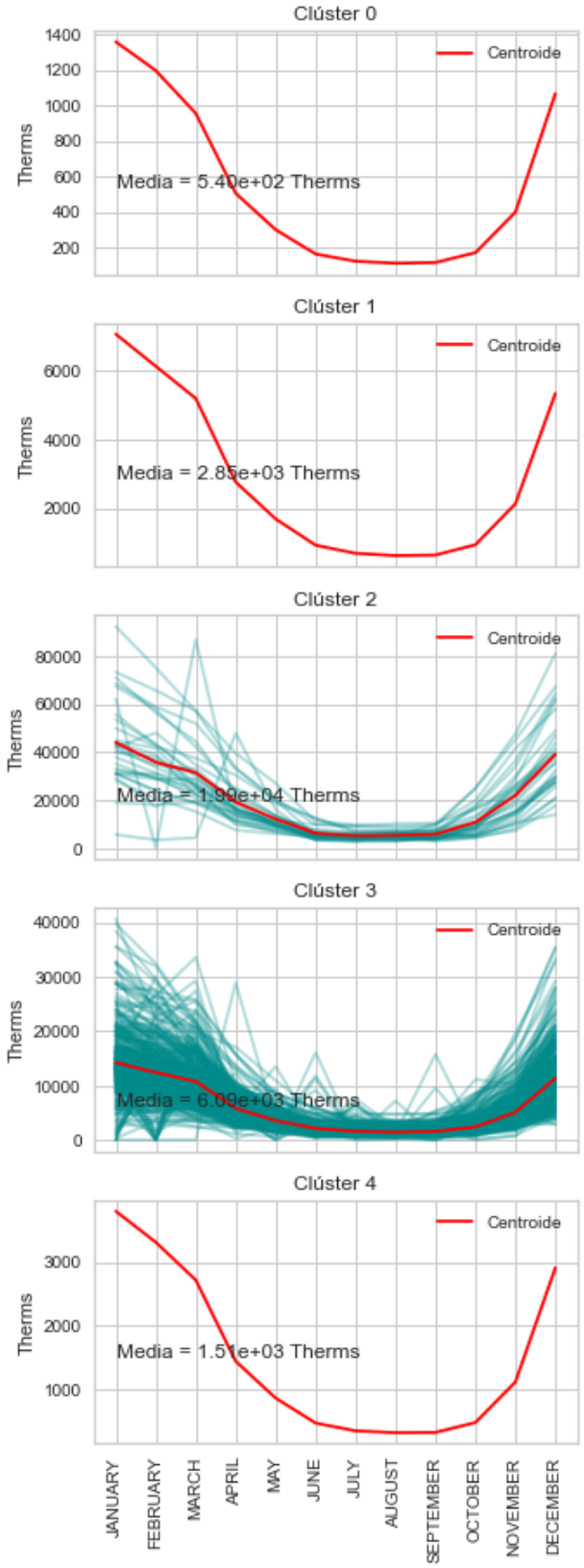

```
<Figure size 576x396 with 0 Axes>
```

[13]:
```
km_best = km.get_kmeans_of(5)
km_labels = km_best.labels_
km_centroids = scaler.inverse_transform(km_best.cluster_centers_)

np.unique(km_labels, return_counts=True)
```

[13]:
```
(array([0, 1, 2, 3, 4]),
 array([23267,  4440,    29,   483, 17295], dtype=int64))
```

[14]:
```
labels = pd.DataFrame(km_labels, columns=["label"])

k = km_centroids.shape[0]
```

```
f, axes = plt.subplots(k, 1, figsize=(5, 15), sharex=True)

for ki in range(k):
    cluster = gas_occupied.iloc[labels.query(f'label == {ki}').index][dl.
 ↪gas_cols]

    palette={i:"darkcyan" for i in range(cluster.shape[0])}

    if len(cluster) < 1000:
        ax0 = sns.lineplot(data=cluster.values.T, ax=axes[ki], palette=palette,␣
 ↪alpha=0.3)
        _ = [line.set_linestyle("-") for line in ax0.lines]

    ax0c = sns.lineplot(data=km_centroids[ki, :].T, ax=axes[ki], color="red")
    _ = [line.set_linestyle("-") for line in ax0c.lines]

    ax0c.legend(ax0c.lines[-1:], ["Centroide"]);

    axes[ki].set_title(f"Clúster {ki}")
    Media = km_centroids[ki, :].mean()
    axes[ki].text(0, Media, f"{Media = :0.2e} Therms")

    axes[ki].set_xticks([e for e in range(0, 12)])
    axes[ki].set_xticklabels([f"{datetime.date(2008, i, 1).strftime('%B').
 ↪upper()}" for i in range(1, 13)], rotation=90)
    axes[ki].set(ylabel="Therms")
    #print(f"Gráfica {ki} realizada")

    cluster_occupied = gas_occupied.iloc[labels.query(f'label == {ki}').
 ↪index]["OCCUPIED UNITS PERCENTAGE"]

    txt = f"Porcentaje ocupación máxima: {cluster_occupied.max()}\n" + \
          f"Porcentaje ocupación minimo: {cluster_occupied.min()}\n" + \
          f"Porcentaje ocupación medio:{cluster_occupied.mean()}"

    axes[ki].text(12, Media, txt)

path = images_path / Path("ensayos/clustering_por_ocupacion_total/gas")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("grafica.svg"))
```



Porcentaje ocupación máxima: 1.0
Porcentaje ocupación minimo: 0.0
Porcentaje ocupación medio:0.8688239093995789

Porcentaje ocupación máxima: 1.0
Porcentaje ocupación minimo: 0.0
Porcentaje ocupación medio:0.8994772747747747

Porcentaje ocupación máxima: 1.0
Porcentaje ocupación minimo: 0.2669
Porcentaje ocupación medio:0.8995620689655172

Porcentaje ocupación máxima: 1.0
Porcentaje ocupación minimo: 0.4719
Porcentaje ocupación medio:0.9083900621118012

Porcentaje ocupación máxima: 1.0
Porcentaje ocupación minimo: 0.0
Porcentaje ocupación medio:0.9088747210176351

[ ]:
```
comunities_by_cluster = {}

for nlab in range(k):
    com_index = np.where(labels==nlab)[0]
    com_list = []
    for c in gas_occupied.values[com_index][:, -1]:
        com_list.append(c)
    comunities_by_cluster[nlab] = com_list

for k in sorted(comunities_by_cluster, key=lambda k:␣
 ↪len(comunities_by_cluster[k])):
    print(f"Al clúster {k} pertenecen las comunidades:")
    for e in np.unique(comunities_by_cluster[k]):
        print("\t", e)
```

1.3.18 Volver al inicio

1.3.19 Clustering por porcentaje de casas en renta

En este caso, se seleccionará el porcentaje de casas en alquiler criterio de comparación.

Energía Seleccionamos las muestras de consumo de electricidad para las zonas residenciales y aplicamos clustering.

[16]:
```
energy_occupied = dl[dl.energy_cols + ['RENTER-OCCUPIED HOUSING PERCENTAGE',␣
 ↪'BUILDING TYPE', 'COMMUNITY AREA NAME']]
energy_occupied = energy_occupied[energy_occupied['BUILDING TYPE'] ==␣
 ↪"Residential"][dl.energy_cols + ['RENTER-OCCUPIED HOUSING PERCENTAGE',␣
 ↪'COMMUNITY AREA NAME']]
energy_occupied
```

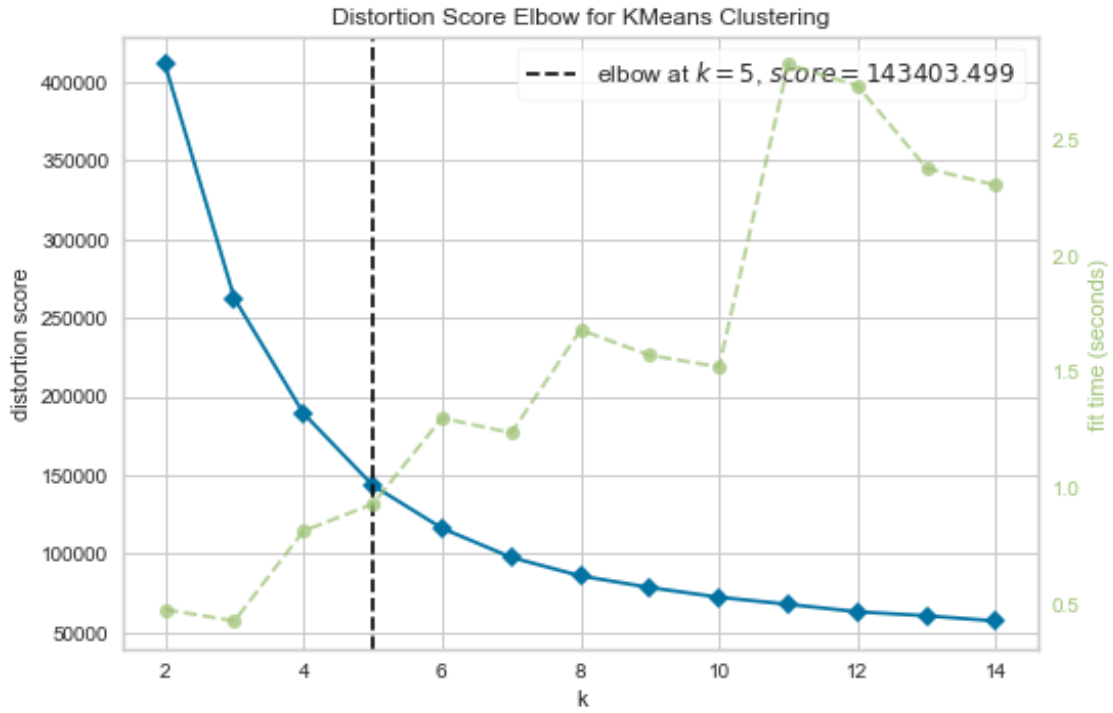
[16]:

	KWH JANUARY 2010	KWH FEBRUARY 2010	KWH MARCH 2010	KWH APRIL 2010	\
1	7334.0	7741.0	4214.0	4284.0	
8	2461.0	4888.0	2893.0	2737.0	
9	0.0	0.0	0.0	0.0	
29	7542.0	7912.0	5451.0	3173.0	
30	0.0	0.0	115.0	695.0	
…	…	…	…	…	
67043	653.0	989.0	1281.0	1110.0	
67045	9572.0	9104.0	8525.0	7756.0	
67046	2705.0	1318.0	1582.0	1465.0	
67048	3567.0	3031.0	2582.0	2295.0	
67050	2717.0	3057.0	2695.0	3793.0	

```
       KWH MAY 2010  KWH JUNE 2010  KWH JULY 2010  KWH AUGUST 2010  \
1            2518.0         4273.0         4566.0           2787.0
8            2350.0         3037.0         3874.0           4861.0
9               0.0          511.0          904.0           1818.0
29           3792.0         4120.0         3197.0           4574.0
30           3527.0         8412.0         4175.0           3900.0
…               …              …              …                …
67043        2077.0         2265.0         3694.0           3588.0
67045       11256.0        11669.0        12099.0          13200.0
67046        1494.0         2990.0         2449.0           2351.0
67048        7902.0         4987.0         5773.0           3996.0
67050        4237.0         5383.0         5544.0           6929.0

       KWH SEPTEMBER 2010  KWH OCTOBER 2010  KWH NOVEMBER 2010  \
1                  3357.0            5540.0            15774.0
8                  5180.0            2984.0             2635.0
9                  1968.0             738.0              450.0
29                 5509.0            6531.0            14402.0
30                 3257.0            3305.0             2813.0
…                     …                 …                 …
67043              3233.0            4107.0             5930.0
67045              9694.0            8419.0            19077.0
67046              1213.0            2174.0             2888.0
67048              3050.0            3103.0             3880.0
67050              5280.0            5971.0             6986.0

       KWH DECEMBER 2010  RENTER-OCCUPIED HOUSING PERCENTAGE  \
1                19676.0                                0.8059
8                 3597.0                                1.0000
9                 2207.0                                0.7270
29               22510.0                                0.8670
30                3344.0                                0.5000
…                    …                                     …
67043             6640.0                                0.7220
67045            18869.0                                0.7030
67046             5025.0                                0.7030
67048             4684.0                                0.8670
67050             5144.0                                0.9329

      COMMUNITY AREA NAME
1                 Ashburn
8                  Austin
9                  Austin
29                Chatham
30                Chatham
…                    …
67043            Woodlawn
```

```
67045            Woodlawn
67046            Woodlawn
67048            Woodlawn
67050            Woodlawn

[48601 rows x 14 columns]
```

[17]:
```
kmax = 15
k_values = (2, kmax)

scaler = preprocessing.StandardScaler()
norm_data = scaler.fit_transform(energy_occupied[dl.energy_cols])

km = KMeansCluster(norm_data)
km.cluster(k_values)

path = images_path / Path("ensayos/clustering_por_ocupacion_en_renta/energia")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("elbow.svg"))
```




```
<Figure size 576x396 with 0 Axes>
```

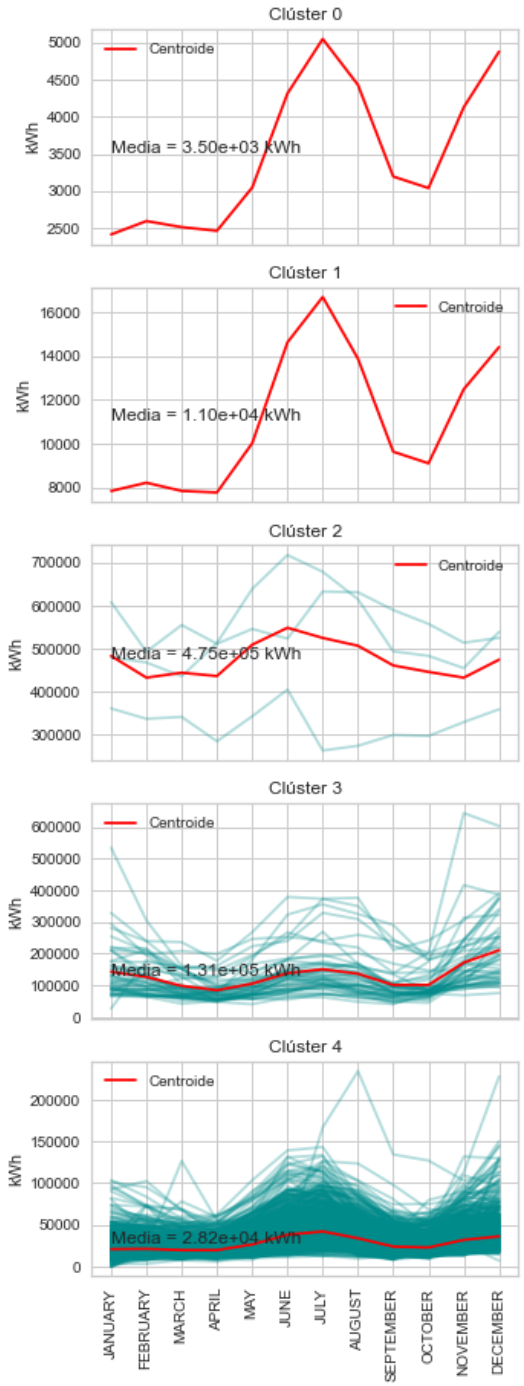
[18]:
```
km_best = km.get_kmeans_of(5)
km_labels = km_best.labels_
```

```
km_centroids = scaler.inverse_transform(km_best.cluster_centers_)

np.unique(km_labels, return_counts=True)
```

[18]:
```
(array([0, 1, 2, 3, 4]),
 array([29677, 17501,     3,    46,  1374], dtype=int64))
```

[19]:
```
labels = pd.DataFrame(km_labels, columns=["label"])

k = km_centroids.shape[0]

f, axes = plt.subplots(k, 1, figsize=(5, 15), sharex=True)

for ki in range(k):
    cluster = energy_occupied.iloc[labels.query(f'label == {ki}').index][dl.
↪energy_cols]

    palette={i:"darkcyan" for i in range(cluster.shape[0])}

    if len(cluster) < 10000:
        ax0 = sns.lineplot(data=cluster.values.T, ax=axes[ki], palette=palette,␣
↪alpha=0.3)
        _ = [line.set_linestyle("-") for line in ax0.lines]

    ax0c = sns.lineplot(data=km_centroids[ki, :].T, ax=axes[ki], color="red")
    _ = [line.set_linestyle("-") for line in ax0c.lines]

    ax0c.legend(ax0c.lines[-1:], ["Centroide"]);

    axes[ki].set_title(f"Clúster {ki}")
    Media = km_centroids[ki, :].mean()
    axes[ki].text(0, Media, f"{Media = :0.2e} kWh")

    axes[ki].set_xticks([e for e in range(0, 12)])
    axes[ki].set_xticklabels([f"{datetime.date(2008, i, 1).strftime('%B').
↪upper()}" for i in range(1, 13)], rotation=90)
    axes[ki].set(ylabel="kWh")
    #print(f"Gráfica {ki} realizada")

    cluster_occupied = energy_occupied.iloc[labels.query(f'label == {ki}').
↪index]["RENTER-OCCUPIED HOUSING PERCENTAGE"]

    txt = f"Porcentaje ocupación de casas en renta máxima: {cluster_occupied.
↪max()}\n" + \
          f"Porcentaje ocupación de casas en renta minimo: {cluster_occupied.
↪min()}\n" + \
```

```
        f"Porcentaje ocupación de casas en renta medio:{cluster_occupied.
↪mean()}"

    axes[ki].text(12, Media, txt)

path = images_path / Path("ensayos/clustering_por_ocupacion_en_renta/energia")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("grafica.svg"))
```



Porcentaje ocupación de casas en renta máxima: 1.0
Porcentaje ocupación de casas en renta minimo: 0.0
Porcentaje ocupación de casas en renta medio:0.5325645348249487

Porcentaje ocupación de casas en renta máxima: 1.0
Porcentaje ocupación de casas en renta minimo: 0.0
Porcentaje ocupación de casas en renta medio:0.3775499057196732

Porcentaje ocupación de casas en renta máxima: 1.0
Porcentaje ocupación de casas en renta minimo: 0.1
Porcentaje ocupación de casas en renta medio:0.511

Porcentaje ocupación de casas en renta máxima: 1.0
Porcentaje ocupación de casas en renta minimo: 0.038
Porcentaje ocupación de casas en renta medio:0.5935391304347827

Porcentaje ocupación de casas en renta máxima: 1.0
Porcentaje ocupación de casas en renta minimo: 0.0
Porcentaje ocupación de casas en renta medio:0.36725509461426487

[ ]:
```
comunities_by_cluster = {}

for nlab in range(k):
    com_index = np.where(labels==nlab)[0]
    com_list = []
    for c in energy_occupied.values[com_index][:, -1]:
        com_list.append(c)
    comunities_by_cluster[nlab] = com_list

for k in sorted(comunities_by_cluster, key=lambda k:␣
 ↪len(comunities_by_cluster[k])):
    print(f"Al clúster {k} pertenecen las comunidades:")
    for e in np.unique(comunities_by_cluster[k]):
        print("\t", e)
```

Gas Seleccionamos las muestras de consumo de gas para las zonas residenciales y aplicamos clustering.

[21]:
```
gas_occupied = dl[dl.gas_cols + ['RENTER-OCCUPIED HOUSING PERCENTAGE',␣
 ↪'BUILDING TYPE', 'COMMUNITY AREA NAME']]
gas_occupied = gas_occupied[gas_occupied['BUILDING TYPE'] == "Residential"][dl.
 ↪gas_cols + ['RENTER-OCCUPIED HOUSING PERCENTAGE', 'COMMUNITY AREA NAME']]
gas_occupied
```

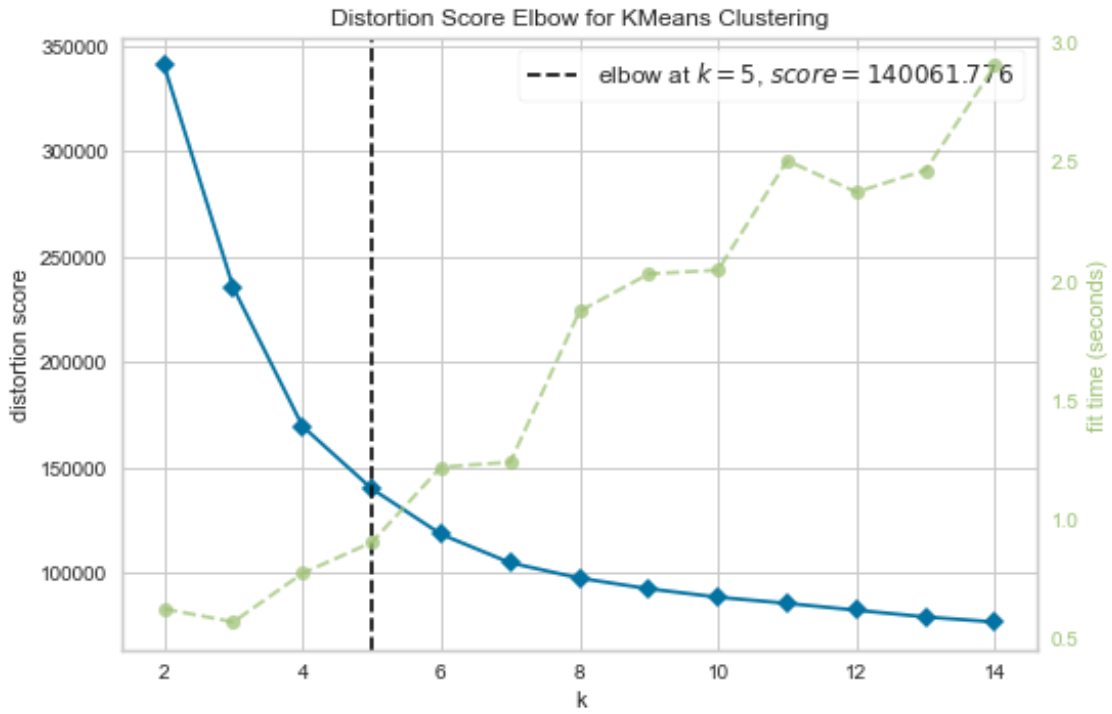
[21]:

	THERM JANUARY 2010	THERM FEBRUARY 2010	THERM MARCH 2010	\
0	2326.0	2131.0	1400.0	
19	285.0	285.0	244.0	
22	243.0	195.0	156.0	
39	555.0	440.0	378.0	
41	326.0	305.0	244.0	
…	…	…	…	
67043	3299.0	1884.0	1944.0	
67045	6914.0	5433.0	5054.0	
67046	2166.0	1681.0	1858.0	
67048	2202.0	1874.0	1647.0	
67050	2372.0	1787.0	1449.0	

	THERM APRIL 2010	THERM MAY 2010	THERM JUNE 2010	THERM JULY 2010	\
0	620.0	502.0	224.0	222.0	
19	129.0	73.0	46.0	39.0	
22	46.0	6.0	7.0	6.0	
39	228.0	126.0	88.0	61.0	
41	137.0	96.0	76.0	44.0	
…	…	…	…	…	

```
67043            876.0          712.0           372.0           271.0
67045           2967.0         2241.0          1107.0           770.0
67046           1172.0          708.0           360.0            72.0
67048            906.0          645.0           346.0            84.0
67050            718.0          572.0           286.0           155.0

       THERM AUGUST 2010  THERM SEPTEMBER 2010  THERM OCTOBER 2010  \
0                  187.0                 197.0               252.0
19                  27.0                  39.0                29.0
22                   6.0                   7.0                 5.0
39                  62.0                  63.0                69.0
41                  27.0                  31.0                48.0
…                    …                     …                   …
67043              222.0                 184.0               233.0
67045              674.0                 788.0               954.0
67046               67.0                  77.0               185.0
67048              150.0                 150.0               260.0
67050              134.0                 161.0               303.0

       THERM NOVEMBER 2010  THERM DECEMBER 2010  \
0                    744.0               2112.0
19                    50.0                144.0
22                    26.0                 81.0
39                   173.0                304.0
41                    71.0                205.0
…                      …                   …
67043                524.0               1135.0
67045               2423.0               4619.0
67046                623.0               1800.0
67048                694.0               1335.0
67050                588.0               1469.0

       RENTER-OCCUPIED HOUSING PERCENTAGE COMMUNITY AREA NAME
0                                  0.3910      Archer Heights
19                                 0.1330            Avondale
22                                 1.0000             Beverly
39                                 0.2170            Clearing
41                                 0.1250             Dunning
…                                   …                  …
67043                              0.7220            Woodlawn
67045                              0.7030            Woodlawn
67046                              0.7030            Woodlawn
67048                              0.8670            Woodlawn
67050                              0.9329            Woodlawn

[45463 rows x 14 columns]
```

[22]:
```
kmax = 15
k_values = (2, kmax)

scaler = preprocessing.StandardScaler()
norm_data = scaler.fit_transform(gas_occupied[dl.gas_cols])

km = KMeansCluster(norm_data)
km.cluster(k_values)

path = images_path / Path("ensayos/clustering_por_ocupacion_en_renta/gas")
path.mkdir(parents=True, exist_ok=True)
plt.savefig(path / Path("elbow.svg"))
```



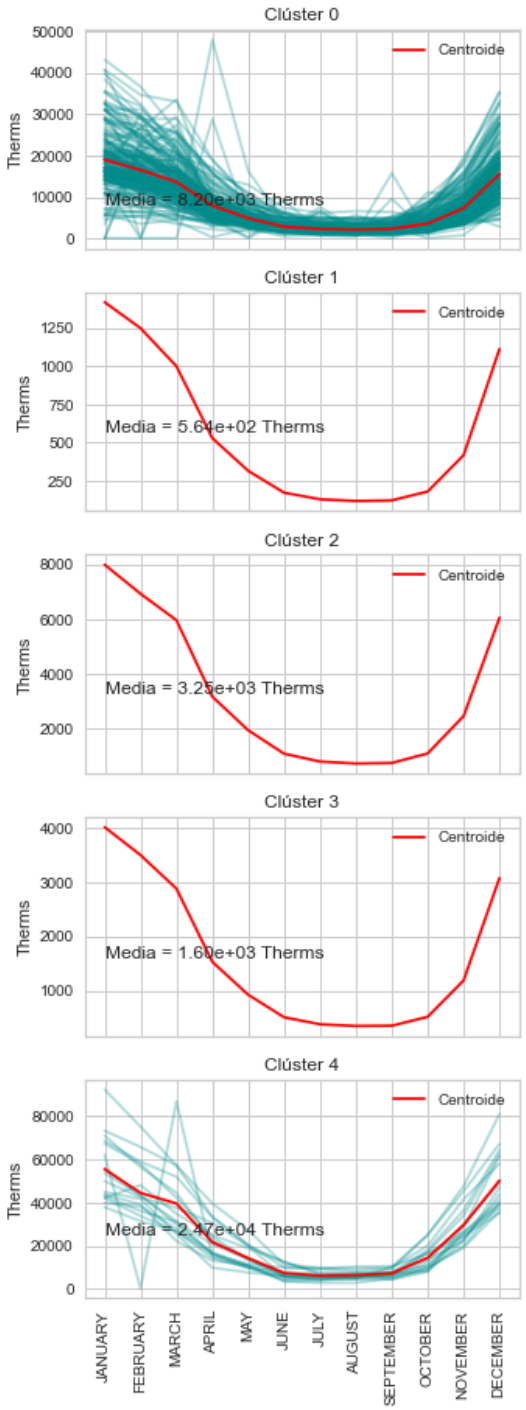

```
<Figure size 576x396 with 0 Axes>
```

[23]:
```
km_best = km.get_kmeans_of(5)
km_labels = km_best.labels_
km_centroids = scaler.inverse_transform(km_best.cluster_centers_)

np.unique(km_labels, return_counts=True)
```

[23]:
```
(array([0, 1, 2, 3, 4]),
 array([  200, 24350,  3402, 17495,    16], dtype=int64))
```

[24]:
```
labels = pd.DataFrame(km_labels, columns=["label"])

k = km_centroids.shape[0]

f, axes = plt.subplots(k, 1, figsize=(5, 15), sharex=True)

for ki in range(k):
    cluster = gas_occupied.iloc[labels.query(f'label == {ki}').index][dl.
 ↪gas_cols]

    palette={i:"darkcyan" for i in range(cluster.shape[0])}

    if len(cluster) < 1000:
        ax0 = sns.lineplot(data=cluster.values.T, ax=axes[ki], palette=palette,␣
 ↪alpha=0.3)
        _ = [line.set_linestyle("-") for line in ax0.lines]

    ax0c = sns.lineplot(data=km_centroids[ki, :].T, ax=axes[ki], color="red")
    _ = [line.set_linestyle("-") for line in ax0c.lines]

    ax0c.legend(ax0c.lines[-1:], ["Centroide"]);

    axes[ki].set_title(f"Clúster {ki}")
    Media = km_centroids[ki, :].mean()
    axes[ki].text(0, Media, f"{Media = :0.2e} Therms")

    axes[ki].set_xticks([e for e in range(0, 12)])
    axes[ki].set_xticklabels([f"{datetime.date(2008, i, 1).strftime('%B').
 ↪upper()}" for i in range(1, 13)], rotation=90)
    axes[ki].set(ylabel="Therms")
    #print(f"Gráfica {ki} realizada")

    cluster_occupied = gas_occupied.iloc[labels.query(f'label == {ki}').
 ↪index]["RENTER-OCCUPIED HOUSING PERCENTAGE"]

    txt = f"Porcentaje ocupación de casas en renta máxima: {cluster_occupied.
 ↪max()}\n" + \
          f"Porcentaje ocupación de casas en renta minimo: {cluster_occupied.
 ↪min()}\n" + \
          f"Porcentaje ocupación de casas en renta medio:{cluster_occupied.
 ↪mean()}"

    axes[ki].text(12, Media, txt)

path = images_path / Path("ensayos/clustering_por_ocupacion_en_renta/gas")
path.mkdir(parents=True, exist_ok=True)
```

```
plt.savefig(path / Path("grafica.svg"))
```



Porcentaje ocupación de casas en renta máxima: 1.0
Porcentaje ocupación de casas en renta minimo: 0.0
Porcentaje ocupación de casas en renta medio:0.579664

Porcentaje ocupación de casas en renta máxima: 1.0
Porcentaje ocupación de casas en renta minimo: 0.0
Porcentaje ocupación de casas en renta medio:0.5091561314168379

Porcentaje ocupación de casas en renta máxima: 1.0
Porcentaje ocupación de casas en renta minimo: 0.0
Porcentaje ocupación de casas en renta medio:0.539351469723692

Porcentaje ocupación de casas en renta máxima: 1.0
Porcentaje ocupación de casas en renta minimo: 0.0
Porcentaje ocupación de casas en renta medio:0.3866614289797085

Porcentaje ocupación de casas en renta máxima: 1.0
Porcentaje ocupación de casas en renta minimo: 0.0
Porcentaje ocupación de casas en renta medio:0.61493125

[ ]:
```
comunities_by_cluster = {}
```

```
for nlab in range(k):
    com_index = np.where(labels==nlab)[0]
    com_list = []
    for c in gas_occupied.values[com_index][:, -1]:
        com_list.append(c)
    comunities_by_cluster[nlab] = com_list

for k in sorted(comunities_by_cluster, key=lambda k:␣
 ↪len(comunities_by_cluster[k])):
    print(f"Al clúster {k} pertenecen las comunidades:")
    for e in np.unique(comunities_by_cluster[k]):
        print("\t", e)
```

1.3.20 Volver al inicio