Aluno: Marcelo Barros de Azevedo Vieira

In [10]:
```
!pip install yellowbrick
```

```
Collecting yellowbrick
  Using cached yellowbrick-1.5-py3-none-any.whl.metadata (7.7 kB)
Requirement already satisfied: matplotlib!=3.0.0,>=2.0.2 in /Users/marcelodeaze
vedo/miniconda3/lib/python3.12/site-packages (from yellowbrick) (3.9.2)
Requirement already satisfied: scipy>=1.0.0 in /Users/marcelodeazevedo/minicond
a3/lib/python3.12/site-packages (from yellowbrick) (1.14.1)
Requirement already satisfied: scikit-learn>=1.0.0 in /Users/marcelodeazevedo/m
iniconda3/lib/python3.12/site-packages (from yellowbrick) (1.5.1)
Requirement already satisfied: numpy>=1.16.0 in /Users/marcelodeazevedo/minicon
da3/lib/python3.12/site-packages (from yellowbrick) (2.1.3)
Requirement already satisfied: cycler>=0.10.0 in /Users/marcelodeazevedo/minico
nda3/lib/python3.12/site-packages (from yellowbrick) (0.12.1)
Requirement already satisfied: contourpy>=1.0.1 in /Users/marcelodeazevedo/mini
conda3/lib/python3.12/site-packages (from matplotlib!=3.0.0,>=2.0.2->yellowbric
k) (1.3.0)
Requirement already satisfied: fonttools>=4.22.0 in /Users/marcelodeazevedo/min
iconda3/lib/python3.12/site-packages (from matplotlib!=3.0.0,>=2.0.2->yellowbri
ck) (4.53.1)
Requirement already satisfied: kiwisolver>=1.3.1 in /Users/marcelodeazevedo/min
iconda3/lib/python3.12/site-packages (from matplotlib!=3.0.0,>=2.0.2->yellowbri
ck) (1.4.7)
Requirement already satisfied: packaging>=20.0 in /Users/marcelodeazevedo/minic
onda3/lib/python3.12/site-packages (from matplotlib!=3.0.0,>=2.0.2->yellowbric
k) (24.1)
Requirement already satisfied: pillow>=8 in /Users/marcelodeazevedo/miniconda3/
lib/python3.12/site-packages (from matplotlib!=3.0.0,>=2.0.2->yellowbrick) (1
1.0.0)
Requirement already satisfied: pyparsing>=2.3.1 in /Users/marcelodeazevedo/mini
conda3/lib/python3.12/site-packages (from matplotlib!=3.0.0,>=2.0.2->yellowbric
k) (3.1.4)
Requirement already satisfied: python-dateutil>=2.7 in /Users/marcelodeazevedo/
miniconda3/lib/python3.12/site-packages (from matplotlib!=3.0.0,>=2.0.2->yellow
brick) (2.9.0.post0)
Requirement already satisfied: joblib>=1.2.0 in /Users/marcelodeazevedo/minicon
da3/lib/python3.12/site-packages (from scikit-learn>=1.0.0->yellowbrick)
(1.4.2)
Requirement already satisfied: threadpoolctl>=3.1.0 in /Users/marcelodeazevedo/
miniconda3/lib/python3.12/site-packages (from scikit-learn>=1.0.0->yellowbrick)
(3.5.0)
Requirement already satisfied: six>=1.5 in /Users/marcelodeazevedo/miniconda3/l
ib/python3.12/site-packages (from python-dateutil>=2.7->matplotlib!=3.0.0,>=
2.0.2->yellowbrick) (1.16.0)
Using cached yellowbrick-1.5-py3-none-any.whl (282 kB)
Installing collected packages: yellowbrick
Successfully installed yellowbrick-1.5
```

In [11]:
```
import numpy as np
import pandas as pd
import sys
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans
from sklearn.decomposition import PCA
from sklearn.datasets import make_blobs, make_moons
```

```
from sklearn.preprocessing import StandardScaler
from yellowbrick.cluster import SilhouetteVisualizer
from sklearn.cluster import DBSCAN
from sklearn.preprocessing import StandardScaler
from yellowbrick.cluster import KElbowVisualizer
from sklearn.pipeline import Pipeline
from sklearn.impute import SimpleImputer
from sklearn import preprocessing
from sklearn.cluster import DBSCAN
from scipy.spatial.distance import euclidean
import scipy.cluster.hierarchy as sch
```

1. Versão do Python: 3.13.1

In [12]:
```
print(f"Versão do Python: {sys.version}")
```

```
Versão do Python: 3.12.4 | packaged by Anaconda, Inc. | (main, Jun 18 2024, 1
0:07:17) [Clang 14.0.6 ]
```

2. Demonstrando que esta sendo utilizado o ambiente anaconda

In [13]:
```
!conda info
```

```
/Users/marcelodeazevedo/miniconda3/lib/python3.12/site-packages/conda/base/cont
ext.py:201: FutureWarning: Adding 'defaults' to channel list implicitly is depr
ecated and will be removed in 25.3.

To remove this warning, please choose a default channel explicitly with conda's
regular configuration system, e.g. by adding 'defaults' to the list of channel
s:

  conda config --add channels defaults

For more information see https://docs.conda.io/projects/conda/en/stable/user-gu
ide/configuration/use-condarc.html

  deprecated.topic(

     active environment : base
    active env location : /Users/marcelodeazevedo/miniconda3
            shell level : 1
       user config file : /Users/marcelodeazevedo/.condarc
 populated config files :
          conda version : 24.11.1
    conda-build version : not installed
         python version : 3.12.4.final.0
                 solver : libmamba (default)
       virtual packages : __archspec=1=m1
                          __conda=24.11.1=0
                          __osx=15.2=0
                          __unix=0=0
       base environment : /Users/marcelodeazevedo/miniconda3  (writable)
      conda av data dir : /Users/marcelodeazevedo/miniconda3/etc/conda
  conda av metadata url : None
          channel URLs : https://repo.anaconda.com/pkgs/main/osx-arm64
                         https://repo.anaconda.com/pkgs/main/noarch
                         https://repo.anaconda.com/pkgs/r/osx-arm64
                         https://repo.anaconda.com/pkgs/r/noarch
         package cache : /Users/marcelodeazevedo/miniconda3/pkgs
                         /Users/marcelodeazevedo/.conda/pkgs
       envs directories : /Users/marcelodeazevedo/miniconda3/envs
                         /Users/marcelodeazevedo/.conda/envs
                platform : osx-arm64
              user-agent : conda/24.11.1 requests/2.32.3 CPython/3.12.4 Darwin/2
4.2.0 OSX/15.2 solver/libmamba conda-libmamba-solver/24.7.0 libmambapy/1.5.8 aa
u/0.4.4 c/CO3i7I_7soBQdtRRQNdUJA s/DbTnvuA_yleNyt5rZAW7ag e/pgSw9zK-V7_rgD_uva9
Zlw
                UID:GID : 501:20
             netrc file : None
           offline mode : False
```

3. Bibliotecas utilizadas no ambiente virtual anadonda:

In [14]:
```
!conda list
```

```
# packages in environment at /Users/marcelodeazevedo/miniconda3:
#
# Name                    Version                   Build  Channel
anaconda-anon-usage       0.4.4           py312hd6b623d_100
anaconda-cli-base         0.4.1           py312hca03da5_1
anaconda-client           1.13.0          py312hca03da5_0
anaconda-cloud-auth       0.7.2           py312hca03da5_0
anaconda-navigator        2.6.4           py312hca03da5_0
annotated-types           0.6.0           py312hca03da5_0
anyio                     4.6.0                    pypi_0    pypi
appnope                   0.1.4                    pypi_0    pypi
archspec                  0.2.3              pyhd3eb1b0_0
argon2-cffi               23.1.0                   pypi_0    pypi
argon2-cffi-bindings      21.2.0                   pypi_0    pypi
arrow                     1.3.0                    pypi_0    pypi
asttokens                 2.4.1                    pypi_0    pypi
async-lru                 2.0.4                    pypi_0    pypi
attrs                     24.2.0          py312hca03da5_0
babel                     2.16.0                   pypi_0    pypi
beautifulsoup4            4.12.3                   pypi_0    pypi
bleach                    6.1.0                    pypi_0    pypi
boltons                   23.0.0          py312hca03da5_0
brotli-python             1.0.9           py312h313beb8_8
bzip2                     1.0.8                h80987f9_6
c-ares                    1.19.1               h80987f9_0
ca-certificates           2024.11.26           hca03da5_0
certifi                   2024.8.30       py312hca03da5_0
cffi                      1.16.0          py312h80987f9_1
chardet                   4.0.0           py312hca03da5_1003
charset-normalizer        3.3.2              pyhd3eb1b0_0
click                     8.1.7           py312hca03da5_0
colorama                  0.4.6           py312hca03da5_0
comm                      0.2.2                    pypi_0    pypi
conda                     24.11.1         py312hca03da5_0
conda-content-trust       0.2.0           py312hca03da5_1
conda-libmamba-solver     24.7.0             pyhd3eb1b0_0
conda-package-handling    2.3.0           py312hca03da5_0
conda-package-streaming   0.10.0          py312hca03da5_0
conda-repo-cli            1.0.114         py312hca03da5_0
conda-token               0.5.0              pyhd3eb1b0_0
contourpy                 1.3.0                    pypi_0    pypi
cryptography              42.0.5          py312hd4332d6_1
cycler                    0.12.1                   pypi_0    pypi
cyrus-sasl                2.1.28               h9131b1a_1
debugpy                   1.8.5                    pypi_0    pypi
decorator                 5.1.1                    pypi_0    pypi
defusedxml                0.7.1              pyhd3eb1b0_0
distro                    1.9.0           py312hca03da5_0
dmglib                    0.9.5           py312hca03da5_0
executing                 2.1.0                    pypi_0    pypi
expat                     2.6.2                h313beb8_0
fmt                       9.1.0                h48ca7d4_1
fonttools                 4.53.1                   pypi_0    pypi
fqdn                      1.5.1                    pypi_0    pypi
freetype                  2.12.1               h1192e45_0
frozendict                2.4.2           py312hca03da5_0
gettext                   0.21.0               h13f89a0_1
glib                      2.78.4               h313beb8_0
glib-tools                2.78.4               h313beb8_0
gst-plugins-base          1.14.1               h313beb8_1
gstreamer                 1.14.1               h80987f9_1
h11                       0.14.0                   pypi_0    pypi
```

```
httpcore                  1.0.5                    pypi_0    pypi
httpx                     0.27.2                   pypi_0    pypi
icu                       73.1                 h313beb8_0
idna                      3.7              py312hca03da5_0
ipykernel                 6.29.5                   pypi_0    pypi
ipython                   8.27.0                   pypi_0    pypi
ipywidgets                8.1.5                    pypi_0    pypi
isoduration               20.11.0                  pypi_0    pypi
jaraco.classes            3.2.1               pyhd3eb1b0_0
jedi                      0.19.1                   pypi_0    pypi
jinja2                    3.1.4                    pypi_0    pypi
joblib                    1.4.2                    pypi_0    pypi
jpeg                      9e                   h80987f9_3
json5                     0.9.25                   pypi_0    pypi
jsonpatch                 1.33             py312hca03da5_1
jsonpointer               2.1                 pyhd3eb1b0_0
jsonschema                4.23.0           py312hca03da5_0
jsonschema-specifications 2023.12.1                pypi_0    pypi
jupyter                   1.1.1                    pypi_0    pypi
jupyter-client            8.6.2                    pypi_0    pypi
jupyter-console           6.6.3                    pypi_0    pypi
jupyter-events            0.10.0                   pypi_0    pypi
jupyter-lsp               2.2.5                    pypi_0    pypi
jupyter-server            2.14.2                   pypi_0    pypi
jupyter-server-terminals  0.5.3                    pypi_0    pypi
jupyter_core              5.7.2            py312hca03da5_0
jupyterlab                4.2.5                    pypi_0    pypi
jupyterlab-pygments       0.3.0                    pypi_0    pypi
jupyterlab-server         2.27.3                   pypi_0    pypi
jupyterlab-widgets        3.0.13                   pypi_0    pypi
keyring                   24.3.1           py312hca03da5_0
kiwisolver                1.4.7                    pypi_0    pypi
krb5                      1.20.1               hf3e1bf2_1
lcms2                     2.12                 hba8e193_0
lerc                      3.0                  hc377ac9_0
libabseil                 20240116.2      cxx17_h313beb8_0
libarchive                3.6.2                h62fee54_3
libclang                  14.0.6          default_h1b80db6_1
libclang13                14.0.6          default_h24352ff_1
libcurl                   8.7.1                h3e2b118_0
libcxx                    14.0.6               h848a8c0_0
libdeflate                1.17                 h80987f9_1
libedit                   3.1.20230828         h80987f9_0
libev                     4.33                 h1a28f6b_1
libffi                    3.4.4                hca03da5_1
libglib                   2.78.4               h0a96307_0
libiconv                  1.16                 h80987f9_3
libllvm14                 14.0.6               h19fdd8a_4
libmamba                  1.5.8                haeffa04_2
libmambapy                1.5.8            py312h1c5506f_2
libnghttp2                1.57.0               h62f6fdd_0
libpng                    1.6.39               h80987f9_0
libpq                     17.2                 h02f6b3c_0
libprotobuf               4.25.3               h514c7bf_0
libsolv                   0.7.24               h514c7bf_1
libssh2                   1.11.0               h3e2b118_0
libtiff                   4.5.1                h313beb8_0
libwebp-base              1.3.2                h80987f9_1
libxml2                   2.10.4               h0b34f26_2
llvm-openmp               14.0.6               hc6e5704_0
lz4-c                     1.9.4                h313beb8_1
markdown-it-py            2.2.0            py312hca03da5_1
```

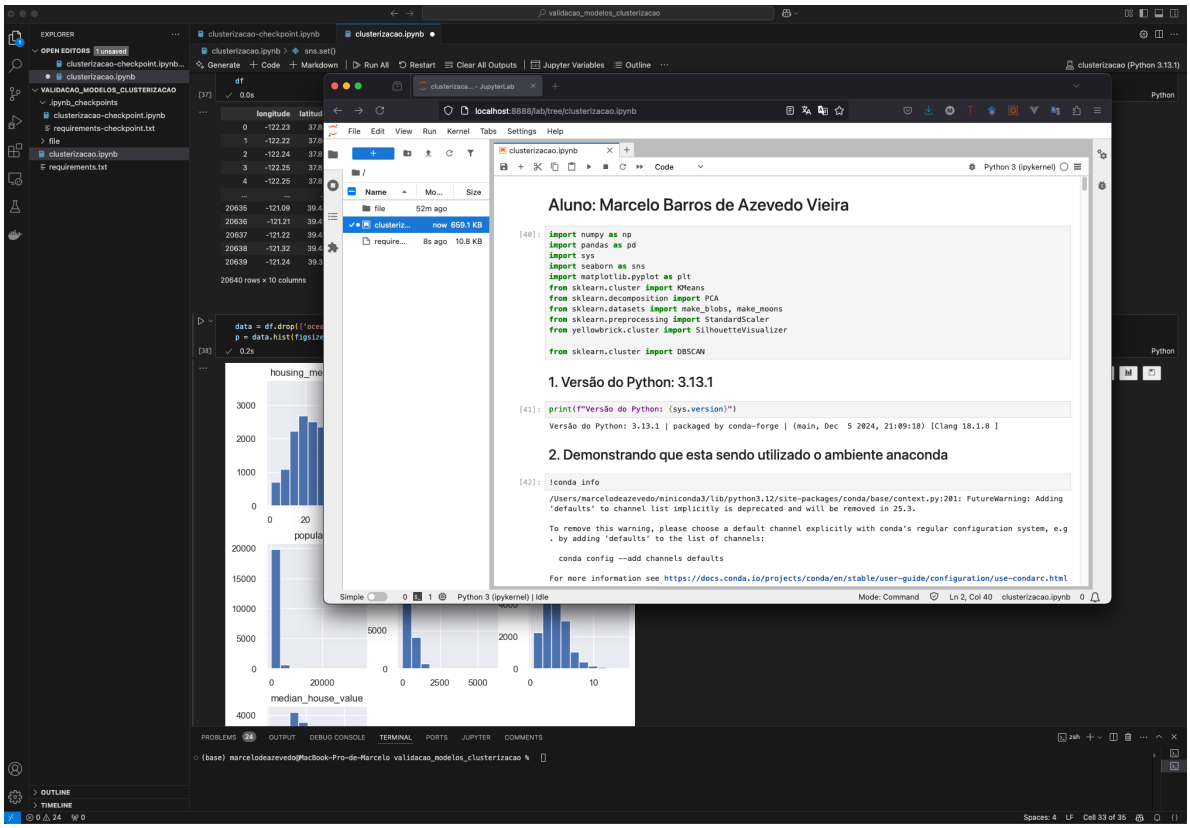
```
markupsafe                2.1.5                    pypi_0    pypi
matplotlib                3.9.2                    pypi_0    pypi
matplotlib-inline         0.1.7                    pypi_0    pypi
mdurl                     0.1.0           py312hca03da5_0
menuinst                  2.1.2           py312hca03da5_0
mistune                   3.0.2                    pypi_0    pypi
more-itertools            10.3.0          py312hca03da5_0
mysql                     8.4.0                h3a6587f_1
navigator-updater         0.5.1           py312hca03da5_0
nbclient                  0.10.0                   pypi_0    pypi
nbconvert                 7.16.4                   pypi_0    pypi
nbformat                  5.10.4          py312hca03da5_0
ncurses                   6.4                  h313beb8_0
nest-asyncio              1.6.0                    pypi_0    pypi
notebook                  7.2.2                    pypi_0    pypi
notebook-shim             0.2.4                    pypi_0    pypi
numpy                     2.1.3                    pypi_0    pypi
openjpeg                  2.5.2                h54b8e55_0
openldap                  2.6.4                he7ef289_0
openssl                   3.0.15               h80987f9_0
overrides                 7.7.0                    pypi_0    pypi
packaging                 24.1            py312hca03da5_0
pandas                    2.2.3                    pypi_0    pypi
pandocfilters             1.5.1                    pypi_0    pypi
parso                     0.8.4                    pypi_0    pypi
pcre2                     10.42                hb066dcc_1
pexpect                   4.9.0                    pypi_0    pypi
pillow                    10.4.0                   pypi_0    pypi
pip                       24.2            py312hca03da5_0
pkce                      1.0.3           py312hca03da5_0
platformdirs              3.10.0          py312hca03da5_0
pluggy                    1.0.0           py312hca03da5_1
ply                       3.11            py312hca03da5_1
prometheus-client         0.21.0                   pypi_0    pypi
prompt-toolkit            3.0.47                   pypi_0    pypi
psutil                    6.0.0                    pypi_0    pypi
ptyprocess                0.7.0                    pypi_0    pypi
pure-eval                 0.2.3                    pypi_0    pypi
pybind11-abi              5                    hd3eb1b0_0
pycosat                   0.6.6           py312h80987f9_1
pycparser                 2.21               pyhd3eb1b0_0
pydantic                  2.8.2           py312hca03da5_0
pydantic-core             2.20.1          py312hf0e4da2_0
pydantic-settings         2.6.1           py312hca03da5_0
pygments                  2.18.0                   pypi_0    pypi
pyjwt                     2.9.0           py312hca03da5_0
pyparsing                 3.1.4                    pypi_0    pypi
pyqt                      5.15.10         py312h313beb8_0
pyqt5-sip                 12.13.0         py312h80987f9_0
pyqtwebengine             5.15.10         py312h313beb8_0
pysocks                   1.7.1           py312hca03da5_0
python                    3.12.4               h99e199e_1
python-dateutil           2.9.0post0      py312hca03da5_2
python-dotenv             0.21.0          py312hca03da5_0
python-fastjsonschema     2.20.0          py312hca03da5_0
python-json-logger        2.0.7                    pypi_0    pypi
python.app                3               py312h80987f9_0
pytz                      2024.2                   pypi_0    pypi
pyyaml                    6.0.2           py312h80987f9_0
pyzmq                     26.2.0                   pypi_0    pypi
qt-main                   5.15.2              h0917680_11
qt-webengine              5.15.9               h2903aaf_7
```

```
qtpy                      2.4.1           py312hca03da5_0
readchar                  4.0.5           py312hca03da5_0
readline                  8.2                  h1a28f6b_0
referencing               0.35.1                   pypi_0    pypi
reproc                    14.2.4               h313beb8_2
reproc-cpp                14.2.4               h313beb8_2
requests                  2.32.3          py312hca03da5_0
requests-toolbelt         1.0.0           py312hca03da5_0
rfc3339-validator         0.1.4                    pypi_0    pypi
rfc3986-validator         0.1.1                    pypi_0    pypi
rich                      13.9.4          py312hca03da5_0
rpds-py                   0.20.0                   pypi_0    pypi
ruamel.yaml               0.17.21         py312h80987f9_0
scikit-learn              1.5.1                    pypi_0    pypi
scipy                     1.14.1                   pypi_0    pypi
seaborn                   0.13.2                   pypi_0    pypi
semver                    3.0.2           py312hca03da5_0
send2trash                1.8.3                    pypi_0    pypi
setuptools                72.1.0          py312hca03da5_0
shellingham               1.5.0           py312hca03da5_0
sip                       6.7.12          py312h313beb8_0
six                       1.16.0             pyhd3eb1b0_1
sniffio                   1.3.1                    pypi_0    pypi
soupsieve                 2.6                      pypi_0    pypi
sqlite                    3.45.3               h80987f9_0
stack-data                0.6.3                    pypi_0    pypi
tabulate                  0.9.0           py312hca03da5_0
terminado                 0.18.1                   pypi_0    pypi
threadpoolctl             3.5.0                    pypi_0    pypi
tinycss2                  1.3.0                    pypi_0    pypi
tk                        8.6.14               h6ba3021_0
tornado                   6.4.1                    pypi_0    pypi
tqdm                      4.66.4          py312h989b03a_0
traitlets                 5.14.3          py312hca03da5_0
truststore                0.8.0           py312hca03da5_0
typer                     0.9.0           py312hca03da5_0
types-python-dateutil     2.9.0.20240906           pypi_0    pypi
typing-extensions         4.11.0          py312hca03da5_0
typing_extensions         4.11.0          py312hca03da5_0
tzdata                    2024.2                   pypi_0    pypi
ujson                     5.10.0          py312h313beb8_0
uri-template              1.3.0                    pypi_0    pypi
urllib3                   2.2.2           py312hca03da5_0
wcwidth                   0.2.13                   pypi_0    pypi
webcolors                 24.8.0                   pypi_0    pypi
webencodings              0.5.1                    pypi_0    pypi
websocket-client          1.8.0                    pypi_0    pypi
wheel                     0.43.0          py312hca03da5_0
widgetsnbextension        4.0.13                   pypi_0    pypi
xz                        5.4.6                h80987f9_1
yaml                      0.2.5                h1a28f6b_0
yaml-cpp                  0.8.0                h313beb8_1
yellowbrick               1.5                      pypi_0    pypi
zlib                      1.2.13               h18a0788_1
zstandard                 0.22.0          py312h1a4646a_0
zstd                      1.5.5                hd90d995_2
```

4. Arquivo com as bibliotecas instaladas

In [15]:
```
!conda list > requirements.txt
```

5. Printscreen do ambiente onde o projeto está sendo executado



6. GitHub do Projeto

https://github.com/marcelobazevedo/validacao_modelos_clusterizacao

1. Escolha da Base de Dados

A Base de dados escolhida foi a California Housing Prices, disponível em https://www.kaggle.com/datasets/camnugent/california-housing-prices/data

2. Justificativa para a escolha da base de dados

O dataset "California Housing Prices" de 1990 é amplamente utilizado em projetos de aprendizado de máquina e análise de dados devido à sua relevância educacional, metodológica e histórica, mesmo com dados antigos. Ele é ideal para introduzir conceitos fundamentais como regressão, engenharia de features e aprendizado supervisionado, graças à sua estrutura clara e documentação acessível. Além disso, muitos padrões subjacentes aos preços de imóveis, como localização, renda e densidade populacional, permanecem válidos e generalizáveis para diferentes contextos.

Seu uso é também justificado pela possibilidade de estudos históricos e comparativos, como a análise das condições do mercado imobiliário em 1990 frente a dados atuais, permitindo explorar mudanças urbanas e socioeconômicas ao longo do tempo. Por ser simplificado, o dataset oferece um ambiente controlado para aprendizado e prática antes de lidar com datasets mais complexos, consolidando habilidades analíticas essenciais.

Portanto, sua utilização é valiosa não apenas como exercício de modelagem teórica e prática, mas também para desenvolver insights generalizáveis e transferíveis para problemas modernos.

In [16]:
```
df = pd.read_csv('file/housing.csv')
df.reset_index(inplace=True, drop=True)
```

In [17]:
```
df
```

Out[17]:

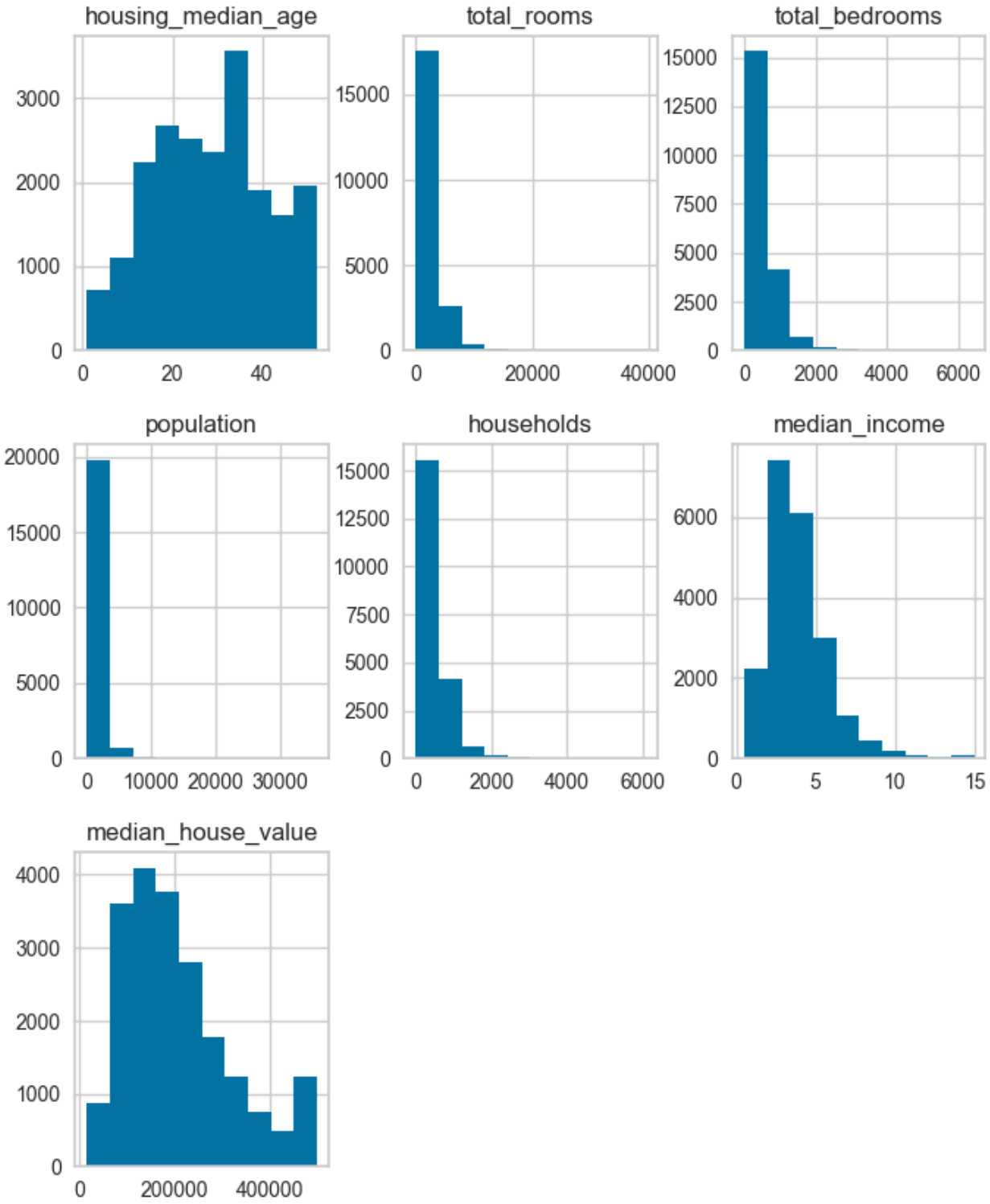
	longitude	latitude	housing_median_age	total_rooms	total_bedrooms	populatic
0	-122.23	37.88	41.0	880.0	129.0	322
1	-122.22	37.86	21.0	7099.0	1106.0	2401
2	-122.24	37.85	52.0	1467.0	190.0	496
3	-122.25	37.85	52.0	1274.0	235.0	558
4	-122.25	37.85	52.0	1627.0	280.0	565
...	...	...	...	...	...	
20635	-121.09	39.48	25.0	1665.0	374.0	845
20636	-121.21	39.49	18.0	697.0	150.0	356
20637	-121.22	39.43	17.0	2254.0	485.0	1007
20638	-121.32	39.43	18.0	1860.0	409.0	741
20639	-121.24	39.37	16.0	2785.0	616.0	1387

20640 rows × 10 columns

3. Gráfico de Faixa dinâmica e o que deve ser feito com os dados antes da clusterização

Normalizar os dados, transformar números do tipo float para inteiros e buscar e corrigir valores nulos

In [18]:
```
data = df.drop(['ocean_proximity','longitude','latitude'],axis=1)
p = data.hist(figsize = (8,10))
```


housing_median_age
total_rooms
total_bedrooms
population
households
median_income
median_house_value

In [19]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20640 entries, 0 to 20639
Data columns (total 10 columns):
 #   Column              Non-Null Count  Dtype  
---  ------              --------------  -----  
 0   longitude           20640 non-null  float64
 1   latitude            20640 non-null  float64
 2   housing_median_age  20640 non-null  float64
 3   total_rooms         20640 non-null  float64
 4   total_bedrooms      20433 non-null  float64
 5   population          20640 non-null  float64
 6   households          20640 non-null  float64
 7   median_income       20640 non-null  float64
 8   median_house_value  20640 non-null  float64
 9   ocean_proximity     20640 non-null  object 
dtypes: float64(9), object(1)
memory usage: 1.6+ MB
```

In [20]:
```
df.describe().T
```

Out[20]:

	count	mean	std	min	25%
longitude	20640.0	-119.569704	2.003532	-124.3500	-121.8000
latitude	20640.0	35.631861	2.135952	32.5400	33.9300
housing_median_age	20640.0	28.639486	12.585558	1.0000	18.0000
total_rooms	20640.0	2635.763081	2181.615252	2.0000	1447.7500
total_bedrooms	20433.0	537.870553	421.385070	1.0000	296.0000
population	20640.0	1425.476744	1132.462122	3.0000	787.0000
households	20640.0	499.539680	382.329753	1.0000	280.0000
median_income	20640.0	3.870671	1.899822	0.4999	2.5634
median_house_value	20640.0	206855.816909	115395.615874	14999.0000	119600.0000

In [21]:
```
data.shape
```

Out[21]:
```
(20640, 7)
```

Identificação de valores nulos

In [22]:
```
print(data.isnull().sum())
```

```
housing_median_age      0
total_rooms             0
total_bedrooms        207
population              0
households              0
median_income           0
median_house_value      0
dtype: int64
```

Correção dos valos nulos

In [23]:
```
data.dropna(inplace=True)
```

In [24]:
```
print(data.isnull().sum())
```

```
housing_median_age    0
total_rooms           0
total_bedrooms        0
population            0
households            0
median_income         0
median_house_value    0
dtype: int64
```

Tranformando dados do tipo float em int

In [25]:
```
data['housing_median_age'] = data['housing_median_age'].astype('int')
data['total_rooms'] = data['total_rooms'].astype('int')
data['total_bedrooms'] = data['total_bedrooms'].astype('int')
data['population'] = data['population'].astype('int')
data['households'] = data['households'].astype('int')
data['median_income'] = data['median_income'].astype('int')
data['median_house_value'] = data['median_house_value'].astype('int')
```

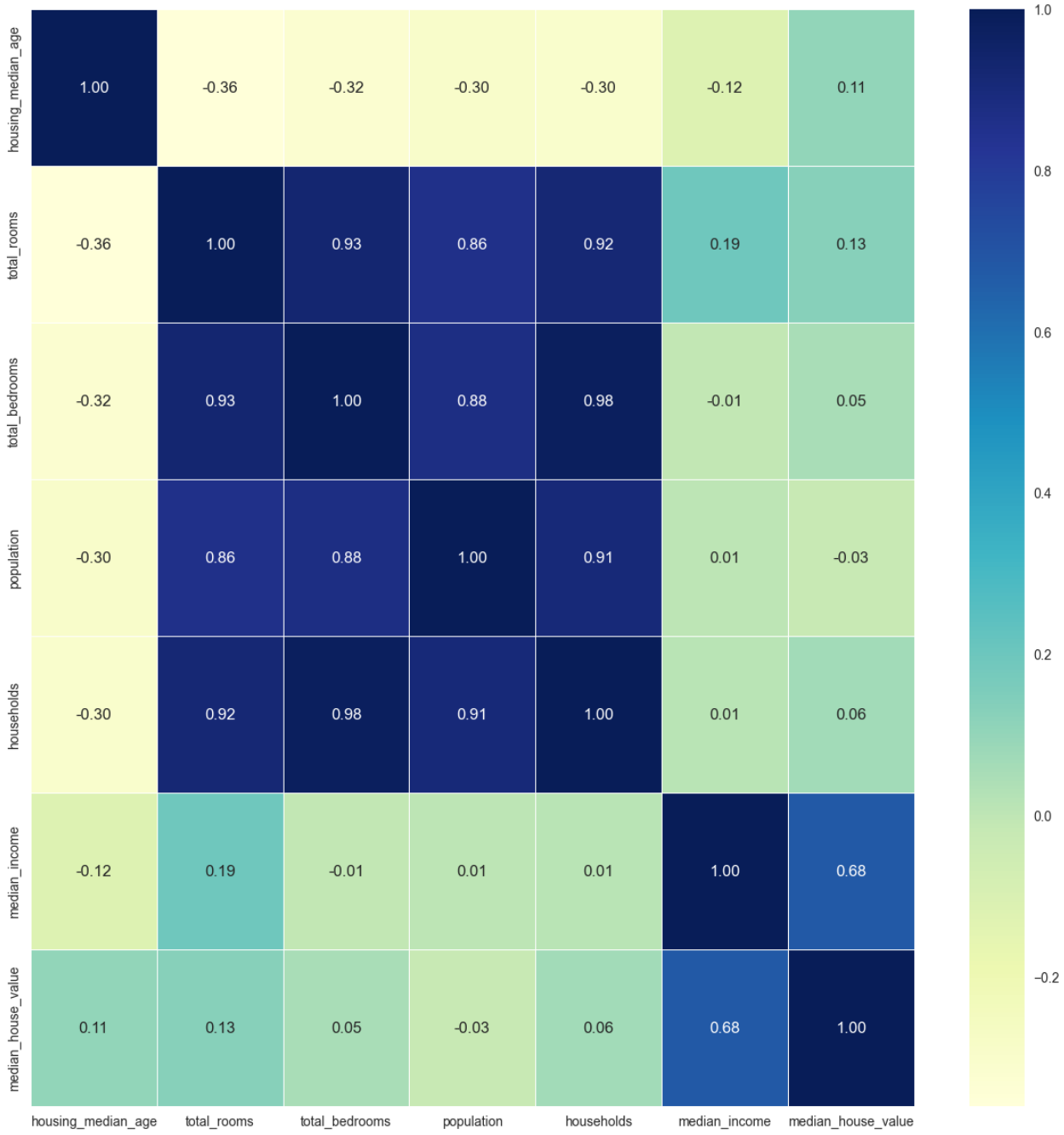
In [26]:
```
data
```

Out[26]:

	housing_median_age	total_rooms	total_bedrooms	population	households	mec
0	41	880	129	322	126	
1	21	7099	1106	2401	1138	
2	52	1467	190	496	177	
3	52	1274	235	558	219	
4	52	1627	280	565	259	
...	...	...	...	...	...	...
20635	25	1665	374	845	330	
20636	18	697	150	356	114	
20637	17	2254	485	1007	433	
20638	18	1860	409	741	349	
20639	16	2785	616	1387	530	

20433 rows × 7 columns

In [27]:
```
corr_matrix = data.corr()
fig, ax = plt.subplots(figsize=(15, 15))
ax = sns.heatmap(corr_matrix,
                 annot=True,
                 linewidths=0.5,
                 fmt=".2f",
                 cmap="YlGnBu");
bottom, top = ax.get_ylim()
```



Normalização dos dados

In [28]:
```
x = data.values
x
```

Out[28]:
```
array([[    41,    880,    129, ...,    126,      8, 452600],
       [    21,   7099,   1106, ...,   1138,      8, 358500],
       [    52,   1467,    190, ...,    177,      7, 352100],
       ...,
       [    17,   2254,    485, ...,    433,      1,  92300],
       [    18,   1860,    409, ...,    349,      1,  84700],
       [    16,   2785,    616, ...,    530,      2,  89400]])
```

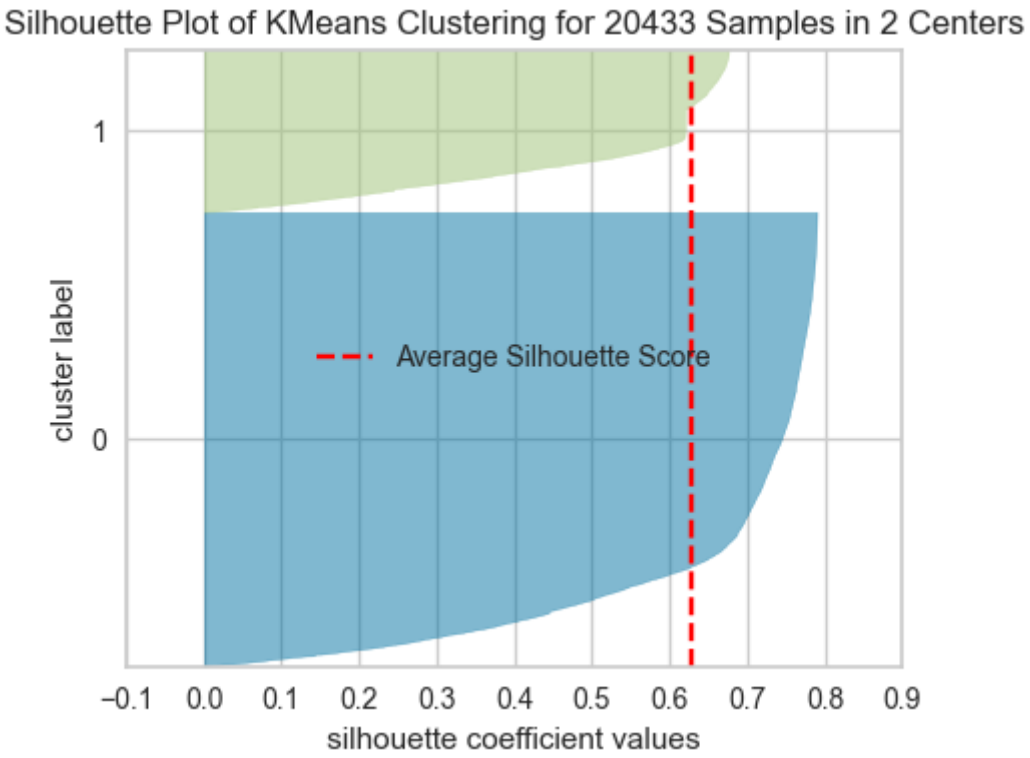
In [29]:
```
scaler = StandardScaler()
X = scaler.fit_transform(x)
X
```

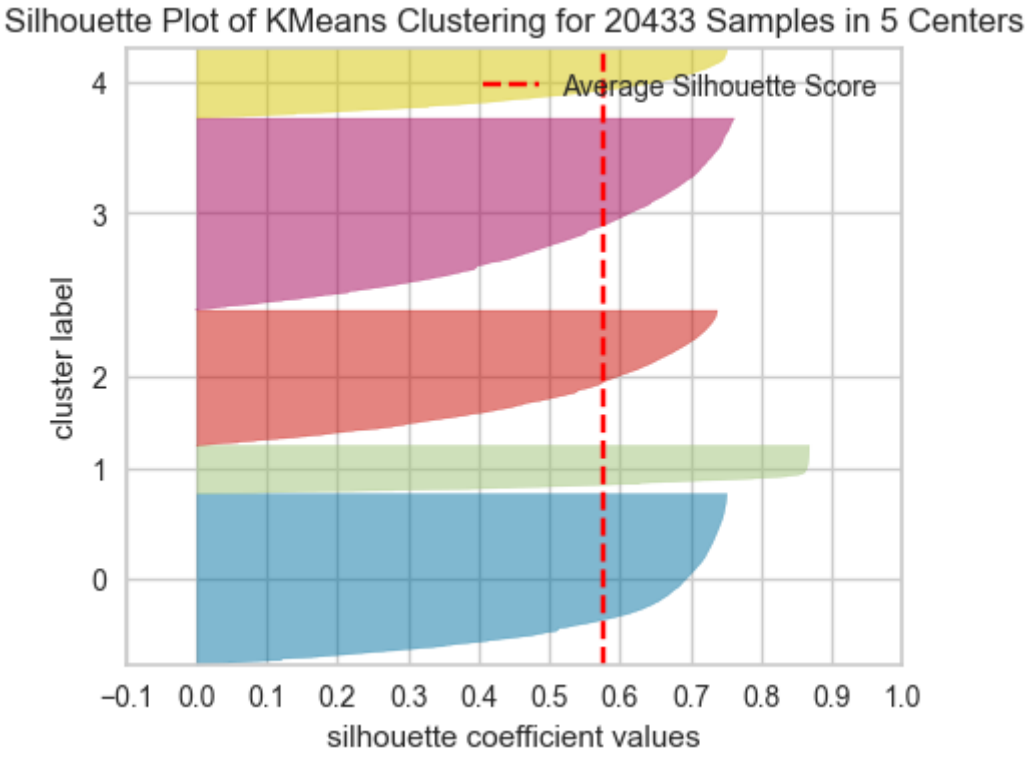
Out[29]:
```
array([[ 0.98216331, -0.8038126 , -0.97032521, ..., -0.97683327,
         2.38510785,  2.12881864],
       [-0.60621017,  2.0421302 ,  1.34827594, ...,  1.67037262,
         2.38510785,  1.31362603],
       [ 1.85576873, -0.53518928, -0.82556097, ..., -0.84342665,
         1.86735089,  1.25818254],
       ...,
       [-0.92388486, -0.17504183, -0.12547157, ..., -0.17377773,
        -1.23919082, -0.99247676],
       [-0.84446619, -0.35534437, -0.30583358, ..., -0.39350628,
        -1.23919082, -1.05831591],
       [-1.00330353,  0.06795473,  0.18541559, ...,  0.07995643,
        -0.72143387, -1.01759959]])
```

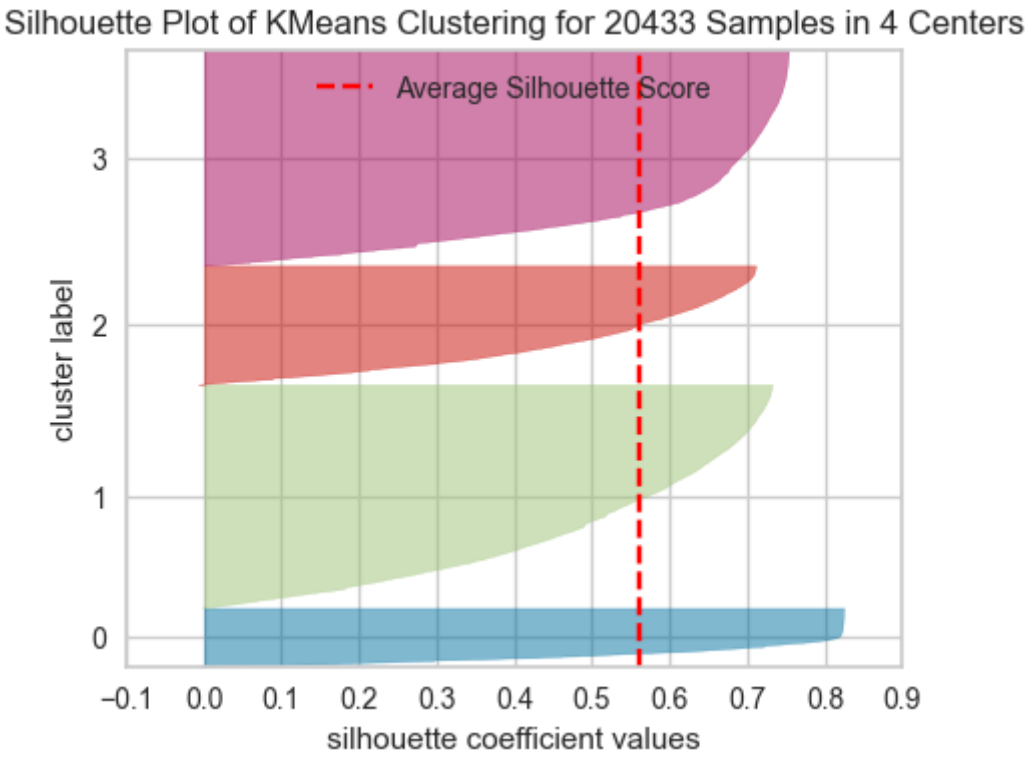
Clusterizacão

k-means

In [30]:
```
#indice de silhueta
km = KMeans(n_clusters=2).fit(x)

fig, ax = plt.subplots(1, 1, figsize=(5, 4))

visualizer = SilhouetteVisualizer(km, ax=ax, colors='yellowbrick')
visualizer.fit(x)
visualizer.show();



km = KMeans(n_clusters=5).fit(x)

fig, ax = plt.subplots(1, 1, figsize=(5, 4))

visualizer = SilhouetteVisualizer(km, ax=ax, colors='yellowbrick')
visualizer.fit(x)
visualizer.show();


km = KMeans(n_clusters=4).fit(x)

fig, ax = plt.subplots(1, 1, figsize=(5, 4))

visualizer = SilhouetteVisualizer(km, ax=ax, colors='yellowbrick')
visualizer.fit(x)
visualizer.show();


km = KMeans(n_clusters=3).fit(x)

fig, ax = plt.subplots(1, 1, figsize=(5, 4))

visualizer = SilhouetteVisualizer(km, ax=ax, colors='yellowbrick')
visualizer.fit(x)
visualizer.show();
```


Silhouette Plot of KMeans Clustering for 20433 Samples in 2 Centers
Average Silhouette Score
cluster label
1
0
−0.1
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
silhouette coefficient values


Silhouette Plot of KMeans Clustering for 20433 Samples in 5 Centers
Average Silhouette Score
cluster label
4
3
2
1
0
−0.1
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0
silhouette coefficient values



1 e 2. Justificar o número de clusters

A diferença mais marcante entre os gráficos está no tamanho das silhuetas de cada um. Observa-se que, com 2 clusters, há uma maior discrepância nos tamanhos das silhuetas, indicando uma menor consistência na formação dos grupos. Por outro lado, com 4 clusters, os tamanhos das silhuetas são mais uniformes, o que sugere que esta configuração proporciona uma divisão mais equilibrada e coesa dos dados, sendo, portanto, a melhor escolha para a quantidade de clusters.

1. K-means

In [46]:
```
kmeans=KMeans(n_clusters=4,random_state=10) #init='k-means++',
y=kmeans.fit_predict(X)
y
```

Out[46]:
```
array([0, 3, 0, ..., 2, 2, 2], dtype=int32)
```

In [47]:
```
data['Cluster']=y
data.groupby('Cluster').mean()
```

Out[47]:

Cluster	housing_median_age	total_rooms	total_bedrooms	population	households
0	31.282156	2390.799592	407.264946	1029.524683	389.401042
1	13.816872	11907.697531	2310.693416	5927.767490	2099.370370
2	31.514947	1676.651601	366.838434	1020.807384	341.135854
3	20.036122	4353.580750	919.498019	2380.515031	846.112328

In [48]:
```
### Clusters 0,1,2 e 3
df_0=data[data['Cluster']==0]
df_1=data[data['Cluster']==1]
df_2=data[data['Cluster']==2]
df_3=data[data['Cluster']==3]
```

In [49]:
```
df_0.head(3)
```

Out[49]:

	housing_median_age	total_rooms	total_bedrooms	population	households	median_i
0	41	880	129	322	126	
2	52	1467	190	496	177	
3	52	1274	235	558	219	

In [50]:
```
df_1.head(3)
```

Out[50]:

	housing_median_age	total_rooms	total_bedrooms	population	households	media
95	36	5329	2477	3469	2323	
283	22	12842	2048	4985	1967	
508	14	7355	2408	3100	2051	

In [51]:
```
import numpy as np
from sklearn.metrics import pairwise_distances

def DBCV(X, labels, metric='euclidean'):
    """
    Density-Based Clustering Validation (DBCV).
    Calcula a métrica de validação baseada em densidade para clusters gerados
    """
    def core_distance(point, neighbors, metric):
        distances = pairwise_distances([point], neighbors, metric=metric)
```

```
        return np.min(distances[distances > 0])

    def reachability_distance(p, o, neighbors, metric):
        return max(core_distance(p, neighbors, metric), np.linalg.norm(p - o))

    def cluster_density(X_cluster, metric):
        n = len(X_cluster)
        distances = pairwise_distances(X_cluster, metric=metric)
        return np.sum(distances) / (n * (n - 1))

    clusters = np.unique(labels)
    total_density = 0
    for cluster in clusters:
        if cluster == -1:  # Ignore noise
            continue
        cluster_points = X[labels == cluster]
        density = cluster_density(cluster_points, metric)
        total_density += density

    return total_density / len(clusters)
```

2. DBScan

In [52]:
```
num_pipeline = Pipeline(
    [
      ("imputer", SimpleImputer(strategy="median")),
    ]
)
df_prepared = num_pipeline.fit_transform(x)
df_prepared = preprocessing.normalize(df_prepared)
```

In [53]:
```
from sklearn.decomposition import PCA

pca = PCA(n_components = 2)
df_principal = pca.fit_transform(df_prepared)
df_principal = pd.DataFrame(df_principal)
df_principal.columns = ['P1', 'P2']
df_principal
```

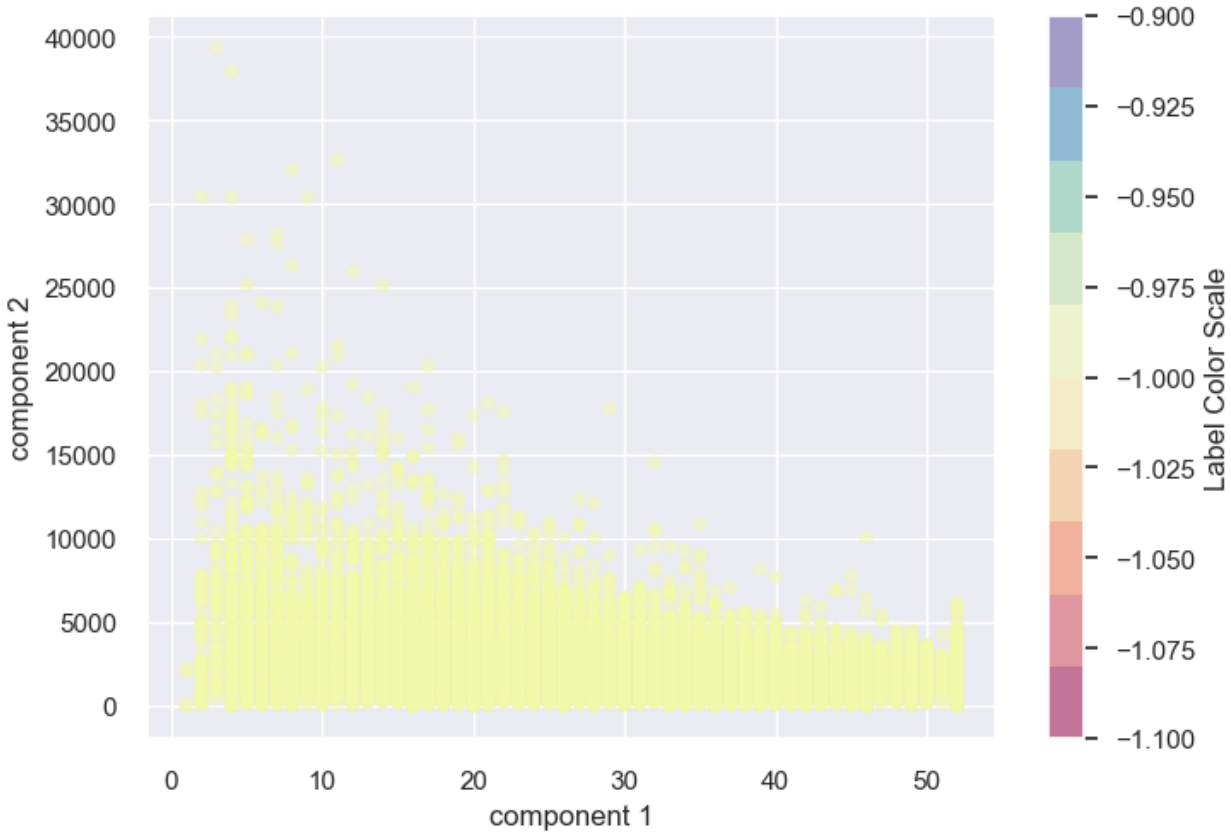
Out[53]:

	P1	P2
0	-0.017062	-0.000235
1	0.001782	-0.004042
2	-0.014778	-0.000756
3	-0.014986	-0.000337
4	-0.014079	-0.000837
...	...	...
20428	0.005517	-0.001245
20429	-0.008732	-0.000447
20430	0.008321	-0.002719
20431	0.005041	-0.003352
20432	0.016689	-0.002132

20433 rows × 2 columns

In [54]:
```
db=DBSCAN(eps=0.5,min_samples=10).fit(data)
labels=db.labels_
```

In [55]:
```
from matplotlib import colormaps
cmap=colormaps.get_cmap('Spectral').resampled(10)

plt.scatter(x[:, 0], x[:, 1],
            c= labels.astype(float), edgecolor='none', alpha=0.5,
            cmap=cmap)
plt.xlabel('component 1')
plt.ylabel('component 2')
plt.colorbar(label="Label Color Scale")
plt.show()
```



3. Compare os dois resultados, aponte as semelhanças e diferenças e interprete

K-means recuperou 4 grupos distintos já o dbscan não conseguiu recuperar

K-means

- facil de ser implementado e interpretado
- é mais escalavel mais eficiente
- requer que o usuario diga inicialmente o n° de clusters *sensivel a outliers

DBScan

- simples e facil de ser implementado
- não requer que o usuario diga p n° de cluster *não é sensivel a outlires

O algoritmo K-means é um método de aprendizado de máquina supervisionado que determina o número de centróides kk, atribuindo cada ponto de dados ao cluster mais próximo e buscando minimizar a distância total dos pontos aos centróides. Em contraste, o DBSCAN (Density-Based Spatial Clustering of Applications with Noise) é baseado na densidade de pontos, identificando clusters como conjuntos densos de pontos conectados. Ele é capaz de dividir regiões densas em clusters, encontrando agrupamentos com formas arbitrárias, mesmo em bases de dados com ruído.

4. Além do índice de silhueta, outras duas métricas de

validação foram utilizadas para comparar os resultados:

Dendrograma - O dendrograma é uma representação gráfica em forma de árvore que demonstra os agrupamentos formados a cada etapa do processo hierárquico e seus níveis de similaridade. Analisando o dendrograma, é possível identificar que os dados foram agrupados em quatro grandes grupos, representando bem as divisões naturais do conjunto de dados.

KElbowVisualizer - O método do cotovelo, representado pelo KElbowVisualizer, avalia a proximidade dos pontos dentro de cada cluster. Observando o gráfico gerado, o ponto ideal para kk é indicado em 3 ou 4 clusters, com k=4k=4 apresentando o melhor desempenho em termos de tempo de resposta e consistência do algoritmo.

5. Comparação com o Índice de Silhueta

- O índice de silhueta mede a qualidade dos clusters considerando a distância entre os centróides e os pontos que os cercam. Ao analisar os gráficos, k=4k=4 apresentou silhuetas mais uniformes e bem distribuídas, indicando uma melhor configuração de agrupamento.

Validação para DBSCAN

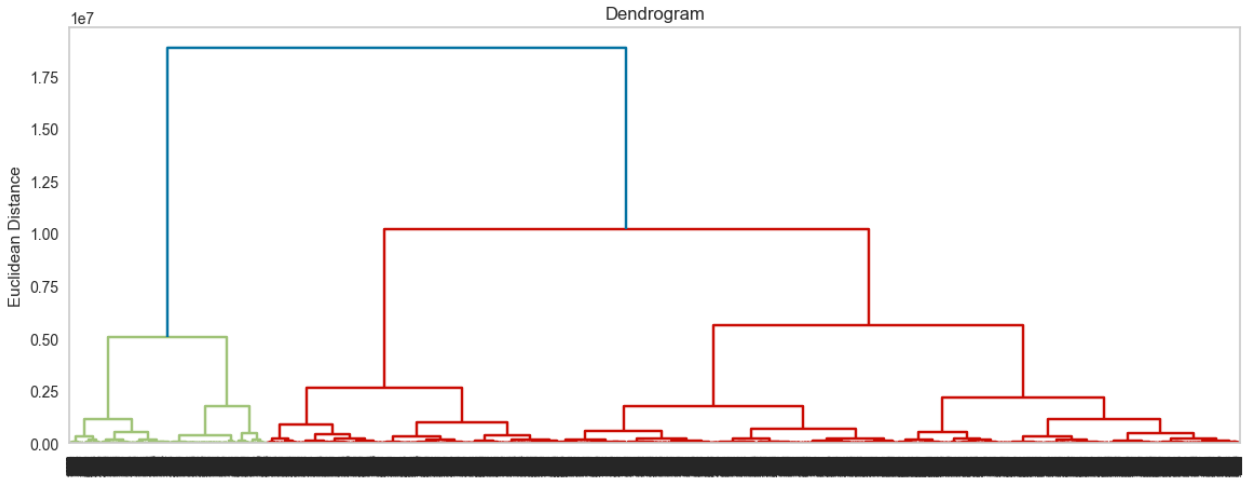
Para o algoritmo DBSCAN, uma métrica mais adequada é o DBCV (Density-Based Cluster Validation), que avalia a qualidade dos clusters com base na densidade dos pontos e não apenas na distância. Essa métrica é particularmente útil para bases de dados com ruído, pois captura a forma dos agrupamentos e considera variações de densidade.

Com base nos resultados das métricas de validação analisadas (índice de silhueta, dendrograma e KElbowVisualizer), o valor ideal para k nos agrupamentos é 4. Essa escolha oferece a melhor combinação de consistência entre os clusters e eficiência na resposta do algoritmo.

In [41]:
```
df=data.drop(['Cluster'],axis=1)
```

In [42]:
```
plt.figure(figsize=(14, 5))
plt.grid(False)
dendrogram = sch.dendrogram(sch.linkage(df, method='ward')) #, labels=df.index
plt.title('Dendrogram')
plt.ylabel('Euclidean Distance')
```

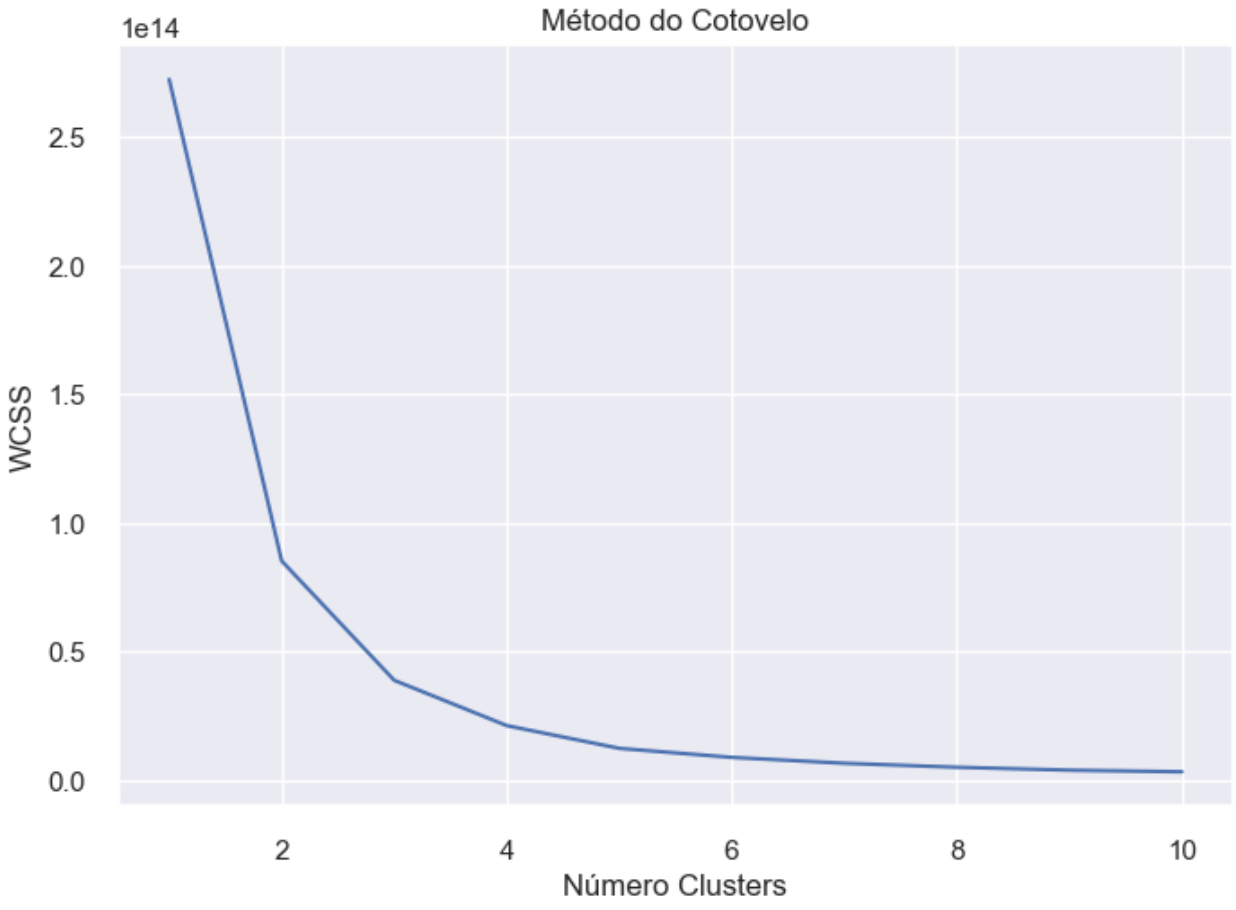
Out[42]:
```
Text(0, 0.5, 'Euclidean Distance')
```



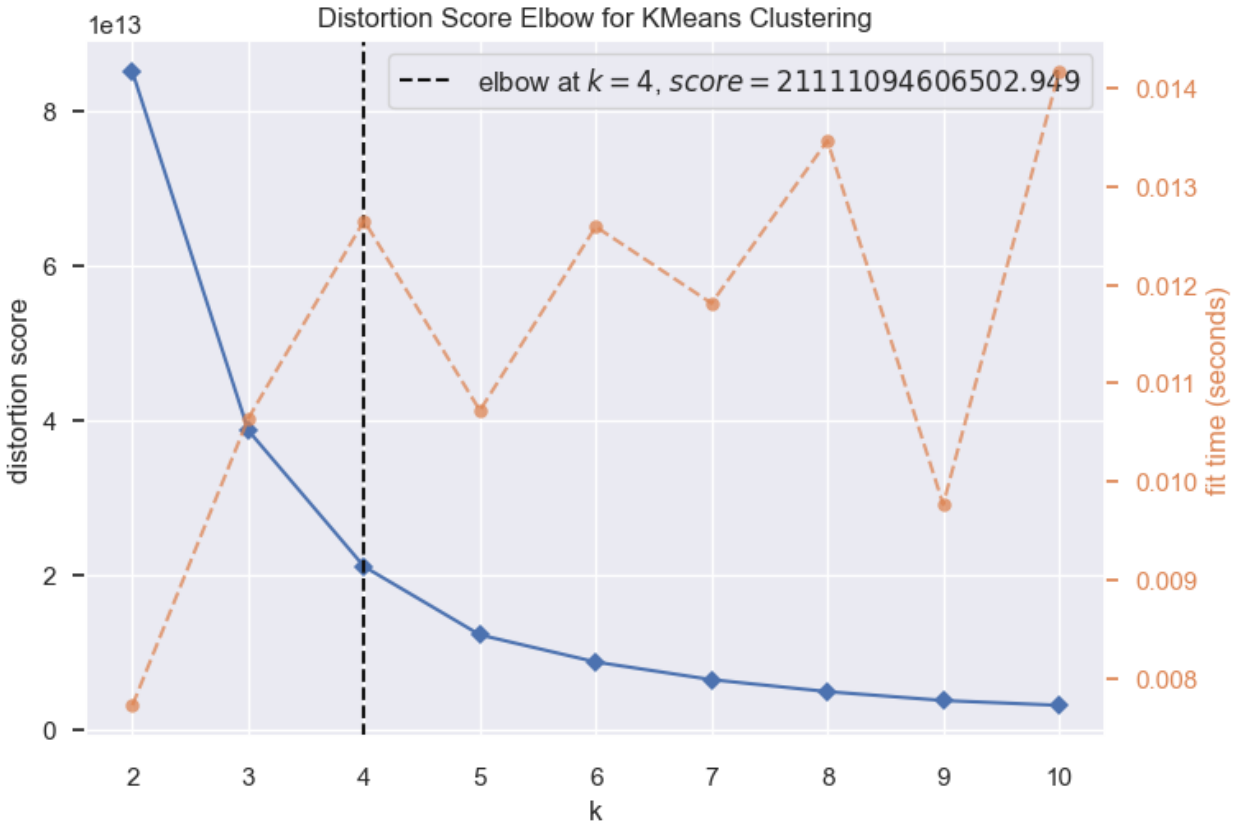
In [43]:
```
Wcss=[]
for i in range(1,11):
    kmeans=KMeans(n_clusters=i,init='k-means++',random_state=9)
    kmeans.fit(x)
    Wcss.append(kmeans.inertia_)
print(Wcss)
```

```
[272394861195557.38, 85139069740659.94, 38736688028295.414, 21111318378341.402,
12269251877454.535, 8758499004866.464, 6471603906594.478, 4923763650377.529, 37
85972457034.503, 3163284787015.993]
```

In [44]:
```
sns.set()
plt.plot(range(1,11),Wcss)
plt.title('Método do Cotovelo')
plt.xlabel('Número Clusters')
plt.ylabel('WCSS')
plt.show()
```



In [45]:
```
from yellowbrick.cluster import KElbowVisualizer
el = KElbowVisualizer(KMeans(), k=10)
el.fit(df)
el.show();
```



5. O Índice de Silhueta é adequado para escolher o número de Clusters no DBSCAN?

Resposta: Quando não existem rótulos disponíveis, é comum recorrer a métricas objetivas, como o Silhouette Score, para avaliar e decidir sobre o resultado final de um agrupamento. O Silhouette Score é uma métrica que mede a coesão e separação dos clusters, com valores variando entre -1 e 1. No entanto, ele não considera o ruído no cálculo e baseia-se exclusivamente em distâncias.

Como o DBSCAN é um algoritmo baseado em densidade, a dependência de distâncias viola um pressuposto fundamental desse método. Ignorar o ruído no cálculo da métrica compromete a avaliação da qualidade dos clusters em técnicas baseadas em densidade.

Portanto, métricas como o Silhouette Score não são **adequadas para medir a qualidade dos agrupamentos** gerados pelo DBSCAN.

Medidas de Similaridade

1. Definição do Problema

Um problema apresenta 10 séries temporais distintas, que precisam ser agrupadas em 3 grupos com base no critério de similaridade, utilizando o valor máximo da correlação

cruzada entre elas. Passos para calcular a similaridade:

- Etapa 1: Para cada par de séries temporais, aplicar um deslocamento (lag) em unidades de tempo.
- Etapa 2: A cada deslocamento, calcular a correlação de Pearson entre as duas séries.
- Etapa 3: Repetir o processo de deslocamento e cálculo da correlação até obter uma curva de correlação cruzada para cada par de séries.
- Etapa 4: Identificar o ponto de maior correlação na curva, que representará o valor máximo de correlação cruzada entre as séries.
- Etapa 5: Usar os valores máximos de correlação como métrica de similaridade entre as séries temporais.

2. Algoritmo de Clusterização

Algoritmo sugerido: KNN (K-Nearest Neighbors)

- O KNN é adequado para estimar densidades, verificando regiões de alta e baixa densidade. Ele fornece um índice de similaridade baseado em distância com valores que variam de -1 a 1.
- Justificativa: É eficaz para dados onde a proximidade entre os valores de similaridade determina o agrupamento.

Algoritmo sugerido: DTWclust (Dynamic Time Warping Clustering)

- O DTWclust utiliza técnicas relacionadas à distância dinâmica e oferece implementações de agrupamentos particionais e hierárquicos.
- Justificativa: Ele pode ser facilmente personalizado com métricas de distância específicas e definições de centróides, sendo uma escolha robusta para séries temporais.

3. Caso de Uso

- Um exemplo de aplicação seria agrupar séries temporais relacionadas ao clima, como padrões anuais de temperatura, ou ciclos de compra e venda em diferentes períodos de tempo. Esses dados podem ser usados para identificar tendências sazonais ou comportamentais.

4. Sugestão de outra Estratégia para Medir Similaridade

1. Definição da Estratégia Medir a similaridade entre séries temporais com base no comportamento de subida e descida em relação ao tempo.

Passos para implementar a estratégia:

- Etapa 1: Identificar o movimento das séries temporais (variações positivas ou

negativas ao longo do tempo).

- Etapa 2: Normalizar as séries temporais para reduzir o impacto de valores extremos.
- Etapa 3: Agrupar os movimentos das séries com base na sincronia entre elas, comparando os padrões de subida e descida ao longo do tempo.
- Etapa 4: Utilizar a correlação de Pearson como métrica de similaridade para quantificar o alinhamento dos movimentos.
- Etapa 5: Utilizar o valor máximo de correlação (entre -1 e 1) para identificar o grau de similaridade entre as séries.

Essa abordagem considera a sincronia do comportamento das séries temporais, independentemente do ruído ou deslocamentos, permitindo uma análise mais contextual das similaridades.

In [ ]: