Código para descargar y consolidar datos de Quién es quién en los precios de Profeco

La información es de 2015 a 2022

In [ ]:
```
#Librerías
import pandas as pd
import requests
import os
import patoolib
import datetime
from matplotlib import pyplot as plt
from matplotlib import dates as mdates
import matplotlib.ticker as ticker
import locale

# Locale para Mex
locale.setlocale(locale.LC_ALL, 'es_MX')
```

Out[ ]:
```
'es_MX'
```

In [ ]:
```
#Cambiar directorio de trabajo
os.chdir("C:/Users/claudio.pacheco/Documents/")
#os.chdir("D:/")
#Crear carpeta. Si ya existe, no la crea
os.makedirs("profeco", exist_ok=True)
```

In [ ]:
```
#Lista de urls
urls=["https://datos.profeco.gob.mx/datos_abiertos/file.php?t=4ecfa981c01e742a5461bf543a7b4108",
      "https://datos.profeco.gob.mx/datos_abiertos/file.php?t=c388a30cb3f4b4c4fa29302618ef5557",
      "https://datos.profeco.gob.mx/datos_abiertos/file.php?t=059e79ffa462f6f51ed3aa1dbfa83a70",
      "https://datos.profeco.gob.mx/datos_abiertos/file.php?t=01fafa951fb6c82e6e4bb491af8f1688",
      "https://datos.profeco.gob.mx/datos_abiertos/file.php?t=09939d92d2afcde64dbc06e057877e16",
      "https://datos.profeco.gob.mx/datos_abiertos/file.php?t=4df382eefa26f1f0d28d3a11aaf41add",
      "https://datos.profeco.gob.mx/datos_abiertos/file.php?t=493b83b886f0266909d783fc8f776b11",
      "https://datos.profeco.gob.mx/datos_abiertos/file.php?t=af88f42c5cb82c6c35dd962b1ae69051"
      ]
```

In [ ]:
```
# Define the destination folder for the downloaded files
folder_path = "profeco"

for url in urls:
    # Define the file name and path
    file_name = "temp.rar"
    file_path = os.path.join(folder_path, file_name)

    # Download the file
    response = requests.get(url)

    # Save the file to the destination folder
    with open(file_path, 'wb') as f:
        f.write(response.content)
```

```
    # Extract the files from the RAR archive
    patoolib.extract_archive(file_path, outdir=folder_path)

    # Remove the RAR file
    os.remove(file_path)
```

```
patool: Extracting profeco\temp.rar ...
patool: running "C:\Program Files\WinRAR\rar.EXE" x -- D:\profeco\temp.rar
patool:     with cwd=profeco
```

In [ ]:
```
#Concatenar archivos de las carpetas en un solo dataframe
#Lista de carpetas
carpetas = os.listdir("profeco")
archivos = [os.listdir("profeco/"+carpeta) for carpeta in carpetas]
```

In [ ]:
```
carpetas
```

Out[ ]:
```
['2015', '2016', '2017', '2018', '2019', '2020', '2021', '2022']
```

In [ ]:
```
archivos
```

Out[ ]:
```
[['012015.csv',
  '022015.csv',
  '032015.csv',
  '042015.csv',
  '052015.csv',
  '062015.csv',
  '072015.csv',
  '082015.csv',
  '092015.csv',
  '102015.csv',
  '112015.csv',
  '122015.csv',
  '132015.csv',
  '142015.csv',
  '152015.csv',
  '162015.csv',
  '172015.csv',
  '182015.csv',
  '192015.csv',
  '202015.csv',
  '212015.csv',
  '222015.csv',
  '232015.csv',
  '242015.csv',
  '252015.csv',
  '262015.csv',
  '272015.csv',
  '282015.csv',
  '292015.csv',
  '302015.csv',
  '312015.csv',
  '322015.csv',
  '332015.csv',
  '342015.csv',
  '352015.csv',
  '362015.csv',
  '372015.csv',
  '382015.csv',
  '392015.csv',
  '402015.csv',
  '412015.csv',
  '422015.csv',
  '432015.csv',
  '442015.csv',
  '452015.csv',
  '462015.csv',
  '472015.csv',
  '482015.csv',
  '492015.csv',
  '502015.csv',
  '512015.csv',
  '522015.csv',
  '532015.csv'],
 ['022016.csv',
  '032016.csv',
```

```
 '042016.csv',
 '052016.csv',
 '062016.csv',
 '072016.csv',
 '082016.csv',
 '092016.csv',
 '102016.csv',
 '112016.csv',
 '122016.csv',
 '132016.csv',
 '142016.csv',
 '152016.csv',
 '162016.csv',
 '172016.csv',
 '182016.csv',
 '192016.csv',
 '202016.csv',
 '212016.csv',
 '222016.csv',
 '232016.csv',
 '242016.csv',
 '252016.csv',
 '262016.csv',
 '272016.csv',
 '282016.csv',
 '292016.csv',
 '302016.csv',
 '312016.csv',
 '322016.csv',
 '332016.csv',
 '342016.csv',
 '352016.csv',
 '362016.csv',
 '372016.csv',
 '382016.csv',
 '392016.csv',
 '402016.csv',
 '412016.csv',
 '422016.csv',
 '432016.csv',
 '442016.csv',
 '452016.csv',
 '462016.csv',
 '472016.csv',
 '482016.csv',
 '492016.csv',
 '502016.csv',
 '512016.csv',
 '522016.csv',
 '532016.csv'],
['012017.csv',
 '022017.csv',
 '032017.csv',
 '042017.csv',
 '052017.csv',
```

```
 '062017.csv',
 '072017.csv',
 '082017.csv',
 '092017.csv',
 '102017.csv',
 '112017.csv',
 '122017.csv',
 '132017.csv',
 '142017.csv',
 '152017.csv',
 '162017.csv',
 '172017.csv',
 '182017.csv',
 '192017.csv',
 '202017.csv',
 '212017.csv',
 '222017.csv',
 '232017.csv',
 '242017.csv',
 '252017.csv',
 '262017.csv',
 '272017.csv',
 '282017.csv',
 '292017.csv',
 '302017.csv',
 '312017.csv',
 '322017.csv',
 '332017.csv',
 '342017.csv',
 '352017.csv',
 '362017.csv',
 '372017.csv',
 '382017.csv',
 '392017.csv',
 '402017.csv',
 '412017.csv',
 '422017.csv',
 '432017.csv',
 '442017.csv',
 '452017.csv',
 '462017.csv',
 '472017.csv',
 '482017.csv',
 '492017.csv',
 '502017.csv',
 '512017.csv',
 '522017.csv'],
['012018.csv',
 '022018.csv',
 '032018.csv',
 '042018.csv',
 '052018.csv',
 '062018.csv',
 '072018.csv',
 '082018.csv',
```

```
 '092018.csv',
 '102018.csv',
 '112018.csv',
 '122018.csv',
 '132018.csv',
 '142018.csv',
 '152018.csv',
 '162018.csv',
 '172018.csv',
 '182018.csv',
 '192018.csv',
 '202018.csv',
 '212018.csv',
 '222018.csv',
 '232018.csv',
 '242018.csv',
 '252018.csv',
 '262018.csv',
 '272018.csv',
 '282018.csv',
 '292018.csv',
 '302018.csv',
 '312018.csv',
 '322018.csv',
 '332018.csv',
 '342018.csv',
 '352018.csv',
 '362018.csv',
 '372018.csv',
 '382018.csv',
 '392018.csv',
 '402018.csv',
 '412018.csv',
 '422018.csv',
 '432018.csv',
 '442018.csv',
 '452018.csv',
 '462018.csv',
 '472018.csv',
 '482018.csv',
 '492018.csv',
 '502018.csv',
 '512018.csv',
 '522018.csv',
 '532018.csv'],
['012019.csv',
 '022019.csv',
 '032019.csv',
 '042019.csv',
 '052019.csv',
 '062019.csv',
 '072019.csv',
 '082019.csv',
 '092019.csv',
 '102019.csv',
```

```
 '112019.csv',
 '122019.csv',
 '132019.csv',
 '142019.csv',
 '152019.csv',
 '162019.csv',
 '172019.csv',
 '182019.csv',
 '192019.csv',
 '202019.csv',
 '212019.csv',
 '222019.csv',
 '232019.csv',
 '242019.csv',
 '252019.csv',
 '262019.csv',
 '272019.csv',
 '282019.csv',
 '292019.csv',
 '302019.csv',
 '312019.csv',
 '322019.csv',
 '332019.csv',
 '342019.csv',
 '352019.csv',
 '362019.csv',
 '372019.csv',
 '382019.csv',
 '392019.csv',
 '402019.csv',
 '412019.csv',
 '422019.csv',
 '432019.csv',
 '442019.csv',
 '452019.csv',
 '462019.csv',
 '472019.csv',
 '482019.csv',
 '492019.csv',
 '502019.csv',
 '512019.csv',
 '522019.csv',
 '532019.csv'],
['012020.csv',
 '022020.csv',
 '032020.csv',
 '042020.csv',
 '052020.csv',
 '062020.csv',
 '072020.csv',
 '082020.csv',
 '092020.csv',
 '102020.csv',
 '112020.csv',
 '122020.csv',
```

```
 '132020.csv',
 '142020.csv',
 '152020.csv',
 '162020.csv',
 '172020.csv',
 '182020.csv',
 '192020.csv',
 '202020.csv',
 '212020.csv',
 '222020.csv',
 '232020.csv',
 '242020.csv',
 '252020.csv',
 '262020.csv',
 '272020.csv',
 '282020.csv',
 '292020.csv',
 '302020.csv',
 '312020.csv',
 '322020.csv',
 '332020.csv',
 '342020.csv',
 '352020.csv',
 '362020.csv',
 '372020.csv',
 '382020.csv',
 '392020.csv',
 '402020.csv',
 '412020.csv',
 '422020.csv',
 '432020.csv',
 '442020.csv',
 '452020.csv',
 '462020.csv',
 '472020.csv',
 '482020.csv',
 '492020.csv',
 '502020.csv',
 '512020.csv',
 '522020.csv',
 '532020.csv'],
['022021.csv',
 '032021.csv',
 '042021.csv',
 '052021.csv',
 '062021.csv',
 '072021.csv',
 '082021.csv',
 '092021.csv',
 '102021.csv',
 '112021.csv',
 '122021.csv',
 '132021.csv',
 '142021.csv',
 '152021.csv',
```

```
 '162021.csv',
 '172021.csv',
 '182021.csv',
 '192021.csv',
 '202021.csv',
 '212021.csv',
 '222021.csv',
 '232021.csv',
 '242021.csv',
 '252021.csv',
 '262021.csv',
 '272021.csv',
 '282021.csv',
 '292021.csv',
 '302021.csv',
 '312021.csv',
 '322021.csv',
 '332021.csv',
 '342021.csv',
 '352021.csv',
 '362021.csv',
 '372021.csv',
 '382021.csv',
 '392021.csv',
 '402021.csv',
 '412021.csv',
 '422021.csv',
 '432021.csv',
 '442021.csv',
 '452021.csv',
 '462021.csv',
 '472021.csv',
 '482021.csv',
 '492021.csv',
 '502021.csv',
 '512021.csv',
 '522021.csv',
 '532021.csv'],
['022022.csv',
 '032022.csv',
 '042022.csv',
 '052022.csv',
 '062022.csv',
 '072022.csv',
 '082022.csv',
 '092022.csv',
 '102022.csv',
 '112022.csv',
 '122022.csv',
 '132022.csv',
 '142022.csv',
 '152022.csv',
 '162022.csv',
 '172022.csv',
 '182022.csv',
```

```
 '192022.csv',
 '202022.csv',
 '212022.csv',
 '222022.csv',
 '232022.csv',
 '242022.csv',
 '252022.csv',
 '262022.csv',
 '272022.csv',
 '282022.csv',
 '292022.csv',
 '302022.csv',
 '312022.csv',
 '322022.csv',
 '332022.csv',
 '342022.csv',
 '352022.csv',
 '362022.csv',
 '372022.csv',
 '382022.csv',
 '392022.csv',
 '402022.csv',
 '412022.csv',
 '422022.csv',
 '432022.csv',
 '442022.csv',
 '452022.csv',
 '462022.csv',
 '472022.csv',
 '482022.csv',
 '492022.csv',
 '502022.csv',
 '512022.csv',
 '522022.csv',
 '532022.csv']]
```

In [ ]:

```
periodos = ["2015-11-13_2015-11-16",
            "2016-11-18_2016-11-21",
            "2017-11-17_2017-11-20",
            "2018-11-16_2018-11-19",
            "2019-11-15_2019-11-18",
            "2020-11-09_2020-11-20",
            "2021-11-10_2021-11-16",
            "2022-11-18_2022-11-21"]

week_numbers_bf = {}

for periodo in periodos:
    inicio, fin = periodo.split("_")
    inicio_date = datetime.datetime.strptime(inicio, "%Y-%m-%d")
    fin_date = datetime.datetime.strptime(fin, "%Y-%m-%d")
    inicio_year = inicio_date.year
    fin_year = fin_date.year
    week_number_inicio = inicio_date.isocalendar()[1]
    week_number_fin = fin_date.isocalendar()[1]
```

```
    if week_number_fin < week_number_inicio:
        week_number_fin = datetime.date(int(fin[:4]), 12, 31).isocalendar()[1]
    if inicio_year == fin_year:
        if inicio_year in week_numbers_bf:
            week_numbers_bf[inicio_year].append(range(week_number_inicio, week_number_fin+1))
        else:
            week_numbers_bf[inicio_year] = [range(week_number_inicio, week_number_fin+1)]
    else:
        if inicio_year in week_numbers_bf:
            week_numbers_bf[inicio_year].append(range(week_number_inicio, 53))
        else:
            week_numbers_bf[inicio_year] = [range(week_number_inicio, 53)]
        for year in range(inicio_year+1, fin_year):
            if year in week_numbers:
                week_numbers_bf[year].append(range(1, 53))
            else:
                week_numbers[year] = [range(1, 53)]
        if fin_year in week_numbers_bf:
            week_numbers_bf[fin_year].append(range(1, week_number_fin+1))
        else:
            week_numbers_bf[fin_year] = [range(1, week_number_fin+1)]
```

In [ ]:
```
week_numbers_bf
```

Out[ ]:
```
{2015: [range(46, 48)],
 2016: [range(46, 48)],
 2017: [range(46, 48)],
 2018: [range(46, 48)],
 2019: [range(46, 48)],
 2020: [range(46, 48)],
 2021: [range(45, 47)],
 2022: [range(46, 48)]}
```

In [ ]:
```
# Periods with 4 weeks before and after

periods = ["2015-11-13_2015-11-16",
           "2016-11-18_2016-11-21",
           "2017-11-17_2017-11-20",
           "2018-11-16_2018-11-19",
           "2019-11-15_2019-11-18",
           "2020-11-09_2020-11-20",
           "2021-11-10_2021-11-16",
           "2022-11-18_2022-11-21"]
# Generate list of week numbers for each period
week_numbers = {}
for period in periods:
    start_date, end_date = period.split("_")
    start_week = datetime.datetime.strptime(start_date, "%Y-%m-%d").isocalendar()[1]
    end_week = datetime.datetime.strptime(end_date, "%Y-%m-%d").isocalendar()[1]
    year = int(start_date[:4])
    week_nums = []
    for week in range(start_week-4, end_week+5):
        if week < 1:
            year -= 1
            week_nums.append(52 + week)
```

```
        elif week > 52:
            year += 1
            week_nums.append(week - 52)
        else:
            week_nums.append(week)
    if year not in week_numbers:
        week_numbers[year] = []
    week_numbers[year].append(week_nums)
```

In [ ]:
```
week_numbers
```

Out[ ]:
```
{2015: [[42, 43, 44, 45, 46, 47, 48, 49, 50, 51]],
 2016: [[42, 43, 44, 45, 46, 47, 48, 49, 50, 51]],
 2017: [[42, 43, 44, 45, 46, 47, 48, 49, 50, 51]],
 2018: [[42, 43, 44, 45, 46, 47, 48, 49, 50, 51]],
 2019: [[42, 43, 44, 45, 46, 47, 48, 49, 50, 51]],
 2020: [[42, 43, 44, 45, 46, 47, 48, 49, 50, 51]],
 2021: [[41, 42, 43, 44, 45, 46, 47, 48, 49, 50]],
 2022: [[42, 43, 44, 45, 46, 47, 48, 49, 50, 51]]}
```

In [ ]:
```
archivos_buenfin = []
for i in range(len(archivos)):
    for j in range(len(archivos[i])):
        year = archivos[i][j][2:6]
        week_number = archivos[i][j][0:2]
        # Filtrar por semana con base en el diccionario week_numbers
        if int(year) in week_numbers:
            for week in week_numbers[int(year)]:
                if int(week_number) in week:
                    archivos_buenfin.append(archivos[i][j])
                    break
```

In [ ]:
```
archivos_buenfin
```

Out[ ]:
```
['422015.csv',
 '432015.csv',
 '442015.csv',
 '452015.csv',
 '462015.csv',
 '472015.csv',
 '482015.csv',
 '492015.csv',
 '502015.csv',
 '512015.csv',
 '422016.csv',
 '432016.csv',
 '442016.csv',
 '452016.csv',
 '462016.csv',
 '472016.csv',
 '482016.csv',
 '492016.csv',
 '502016.csv',
 '512016.csv',
 '422017.csv',
 '432017.csv',
 '442017.csv',
 '452017.csv',
 '462017.csv',
 '472017.csv',
 '482017.csv',
 '492017.csv',
 '502017.csv',
 '512017.csv',
 '422018.csv',
 '432018.csv',
 '442018.csv',
 '452018.csv',
 '462018.csv',
 '472018.csv',
 '482018.csv',
 '492018.csv',
 '502018.csv',
 '512018.csv',
 '422019.csv',
 '432019.csv',
 '442019.csv',
 '452019.csv',
 '462019.csv',
 '472019.csv',
 '482019.csv',
 '492019.csv',
 '502019.csv',
 '512019.csv',
 '422020.csv',
 '432020.csv',
 '442020.csv',
 '452020.csv',
 '462020.csv',
```

```
 '472020.csv',
 '482020.csv',
 '492020.csv',
 '502020.csv',
 '512020.csv',
 '412021.csv',
 '422021.csv',
 '432021.csv',
 '442021.csv',
 '452021.csv',
 '462021.csv',
 '472021.csv',
 '482021.csv',
 '492021.csv',
 '502021.csv',
 '422022.csv',
 '432022.csv',
 '442022.csv',
 '452022.csv',
 '462022.csv',
 '472022.csv',
 '482022.csv',
 '492022.csv',
 '502022.csv',
 '512022.csv']
```

In [ ]:
```python
#Leer los archivos filtrados
df = pd.DataFrame()
for archivo in archivos_buenfin:
    df_temp = pd.read_csv("profeco/"+archivo[2:6]+"/"+archivo, encoding="utf-8",names=["producto",
                                                                                       "presentacion",
                                                                                       "marca",
                                                                                       "categoria",
                                                                                       "catalogo",
                                                                                       "precio",
                                                                                       "fecharegistro",
                                                                                       "cadenacomercial",
                                                                                         "giro",
                                                                                         "nombrecomercial",
                                                                                         "direccion",
                                                                                         "estado",
                                                                                         "municipio",
                                                                                         "latitud",
                                                                                         "longitud",
                                                                                       ])
    df=pd.concat([df,df_temp],ignore_index=True)
    print("Archivo "+archivo+" leído")
```

```
Archivo 422015.csv leído
Archivo 432015.csv leído
Archivo 442015.csv leído
Archivo 452015.csv leído
Archivo 462015.csv leído
Archivo 472015.csv leído
Archivo 482015.csv leído
Archivo 492015.csv leído
Archivo 502015.csv leído
Archivo 512015.csv leído
Archivo 422016.csv leído
Archivo 432016.csv leído
Archivo 442016.csv leído
Archivo 452016.csv leído
Archivo 462016.csv leído
Archivo 472016.csv leído
Archivo 482016.csv leído
Archivo 492016.csv leído
Archivo 502016.csv leído
Archivo 512016.csv leído
Archivo 422017.csv leído
Archivo 432017.csv leído
Archivo 442017.csv leído
Archivo 452017.csv leído
Archivo 462017.csv leído
Archivo 472017.csv leído
Archivo 482017.csv leído
Archivo 492017.csv leído
Archivo 502017.csv leído
Archivo 512017.csv leído
Archivo 422018.csv leído
Archivo 432018.csv leído
Archivo 442018.csv leído
Archivo 452018.csv leído
Archivo 462018.csv leído
Archivo 472018.csv leído
Archivo 482018.csv leído
Archivo 492018.csv leído
Archivo 502018.csv leído
Archivo 512018.csv leído
Archivo 422019.csv leído
Archivo 432019.csv leído
Archivo 442019.csv leído
Archivo 452019.csv leído
Archivo 462019.csv leído
Archivo 472019.csv leído
Archivo 482019.csv leído
Archivo 492019.csv leído
Archivo 502019.csv leído
Archivo 512019.csv leído
Archivo 422020.csv leído
Archivo 432020.csv leído
Archivo 442020.csv leído
Archivo 452020.csv leído
Archivo 462020.csv leído
```

```
Archivo 472020.csv leído
Archivo 482020.csv leído
Archivo 492020.csv leído
Archivo 502020.csv leído
Archivo 512020.csv leído
Archivo 412021.csv leído
Archivo 422021.csv leído
Archivo 432021.csv leído
Archivo 442021.csv leído
Archivo 452021.csv leído
Archivo 462021.csv leído
Archivo 472021.csv leído
Archivo 482021.csv leído
Archivo 492021.csv leído
Archivo 502021.csv leído
Archivo 422022.csv leído
Archivo 432022.csv leído
Archivo 442022.csv leído
Archivo 452022.csv leído
Archivo 462022.csv leído
Archivo 472022.csv leído
Archivo 482022.csv leído
Archivo 492022.csv leído
Archivo 502022.csv leído
Archivo 512022.csv leído
```

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

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 21658296 entries, 0 to 21658295
Data columns (total 15 columns):
 #   Column           Dtype  
---  ------           -----  
 0   producto         object 
 1   presentacion     object 
 2   marca            object 
 3   categoria        object 
 4   catalogo         object 
 5   precio           float64
 6   fecharegistro    object 
 7   cadenacomercial  object 
 8   giro             object 
 9   nombrecomercial  object 
 10  direccion        object 
 11  estado           object 
 12  municipio        object 
 13  latitud          float64
 14  longitud         float64
dtypes: float64(3), object(12)
memory usage: 2.4+ GB
```

In [ ]:
```
#Filtrar electrodomésticos
electrodomesticos = df[df["catalogo"]=="ELECTRODOMESTICOS"]
electrodomesticos["fecharegistro"] = pd.to_datetime(electrodomesticos["fecharegistro"], format="%Y-%m-%d")
```

```
C:\Users\claudio.pacheco\AppData\Local\Temp\ipykernel_12172\1934804542.py:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  electrodomesticos["fecharegistro"] = pd.to_datetime(electrodomesticos["fecharegistro"], format="%Y-%m-%d")
```

In [ ]:
```
#Print precio medio y mediano para cada año
for year in range(2015,2023):
    print("Precio medio y mediano para el año "+str(year))
    print(electrodomesticos[electrodomesticos["fecharegistro"].dt.year==year]["precio"].agg(["mean","median"]))
    print("")
```

```
Precio medio y mediano para el año 2015
mean      4315.537062
median    2399.000000
Name: precio, dtype: float64

Precio medio y mediano para el año 2016
mean      4814.949772
median    3098.000000
Name: precio, dtype: float64

Precio medio y mediano para el año 2017
mean      5725.673986
median    3789.000000
Name: precio, dtype: float64

Precio medio y mediano para el año 2018
mean      5585.716984
median    3599.000000
Name: precio, dtype: float64

Precio medio y mediano para el año 2019
mean      5982.475925
median    4299.000000
Name: precio, dtype: float64

Precio medio y mediano para el año 2020
mean      6210.030425
median    4754.500000
Name: precio, dtype: float64

Precio medio y mediano para el año 2021
mean      6846.266575
median    4599.000000
Name: precio, dtype: float64

Precio medio y mediano para el año 2022
mean      6741.326178
median    4997.000000
Name: precio, dtype: float64
```

In [ ]:
```
#Crear semana de registro
electrodomesticos["semana_registro"] = electrodomesticos["fecharegistro"].dt.isocalendar().week
```

```
C:\Users\claudio.pacheco\AppData\Local\Temp\ipykernel_12172\1117968008.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  electrodomesticos["semana_registro"] = electrodomesticos["fecharegistro"].dt.isocalendar().week
```

In [ ]:
```
electrodomesticos["fecharegistro"].min()
```

Out[ ]:
```
Timestamp('2015-10-12 00:00:00')
```

In [ ]:
```
productos=electrodomesticos["producto"].unique()
productos
```

Out[ ]:
```
array(['BATIDORAS', 'CAFETERAS', 'CAMARAS DIGITALES',
       'COMPONENTES DE AUDIO', 'EXTRACTORES DE JUGOS Y EXPRIMIDORES',
       'PANTALLAS', 'PLANCHAS', 'RADIOGRABADORAS', 'DVD / BLU RAY',
       'ESTUFAS', 'HORNO DE MICROONDAS', 'LAVADORAS', 'LICUADORAS',
       'REFRIGERADORES', 'SISTEMAS DE TEATRO EN CASA HOME THEATER',
       'TOSTADORES DE PAN', 'VENTILADORES', 'SANDWICHERAS',
       'ELECTRONICOS DE VIDEO', 'OLLA DE PRESION EXPRESS',
       'HORNO ELECTRICO', 'AIRES ACONDICIONADOS', 'VIDEOCAMARAS',
       'BOCINAS PORTÁTILES', 'CELULARES', 'SARTÉN', 'SECADORAS',
       'CENTRO DE LAVADO', 'COMPUTADORAS PORTÁTILES', 'TABLET',
       'BATERÍA DE COCINA', 'BARRA DE SONIDO', 'REPRODUCTORES STREAMING',
       'HORNO ELÉCTRICO', 'ASPIRADORAS', 'FREIDORAS DE AIRE',
       'ASISTENTES DE VOZ', 'MINIBOCINAS PORTÁTILES BLUETOOTH',
       'PLANCHAS DE VAPOR VERTICAL'], dtype=object)
```

In [ ]:
```
#Convertir lista de productos en dataframe
df_productos = pd.DataFrame(productos,columns=["producto"])
df_productos
#Salvar en excel
pd.DataFrame.to_excel(df_productos,"profeco/productos.xlsx",index=False)
```

In [ ]:
```
electrodomesticos["cadenacomercial"].unique()
```

Out[ ]:
```
array(['BODEGA AURRERA', 'COPPEL', 'SORIANA', 'SEARS ROEBUCK DE MEXICO',
       'WAL-MART', 'BODEGA COMERCIAL MEXICANA', 'CHEDRAUI', 'ELEKTRA',
       'COMERCIAL MEXICANA', 'FAMSA', 'LIVERPOOL', 'MERCADO SORIANA',
       'I.M.S.S.', 'HIPERMERCADO SORIANA', 'BECERRIL AIR',
       'LEY (AUTOSERVICIO)', 'SALINAS Y ROCHA', 'SORIANA SUPER',
       'CASA LEY', 'MEGA COMERCIAL', 'LEY EXPRESS', 'MUEBLERIAS',
       'MEGA COMERCIAL MEXICANA', 'COLCHONES Y MUEBLES DE CAMPECHE',
       'MEGA SORIANA', 'ULTRA HOGAR', 'MUEBLERIA CREDILAND',
       'MUEBLERIA FABRICAS DE FRANCIA', 'TIENDAS CONTINO',
       'MUEBLERIAS PORTILLO', 'S MART', 'ALSUPER STORE',
       'GALA DISEÑO EN MUEBLES, S.A. DE C.V.', 'CIMACO', 'LA COMER',
       'U.N.A.M.', 'PALACIO DE HIERRO', 'BEST BUY', 'SUPER GURIERREZ',
       'H.E.B.', 'OPERADORA MERCO', 'MUEBLERIAS EL GALLO',
       'SORIANA EXPRESS', 'SUPERAMA', 'RYSE', 'EKAR DE GAS',
       'MUEBLES AMERICA', 'MUEBLERIA EL GRAN SALTO', 'MUEBLERIA ELIZONDO',
       'MUEBLERIA MAYA', 'TELEBODEGA', 'SUPER AKI XTRA', 'MEGA ELEKTRA',
       'SUPER AKI', 'SUPER LEY', 'SUPERMERCADOS SANTA FE',
       'NOVEDADES CLARA', 'DEPARTAMENTAL DEL SOL',
       'MUEBLERIA GUADALUPANA', 'MUEBLERIA TELESERVI', 'ELECTROMUEBLES',
       'ISSSTEZAC', 'I.S.S.S.T.E.', 'PROHOGAR',
       'ELECTRODOMESTICOS CHAPUR', 'DEL SOL', 'RAC', 'CHEDRAUI SELECTO',
       'JUGUETRON', 'MI TIENDA DEL AHORRO', 'GAMERS', 'GAME PLANET',
       'SUPERMERCADOS LEY', 'MERCADO SORIANA EXPRESS', 'EL BODEGON',
       'LA MARINA', 'MUEBLERIA CENTRAL', 'MUEBLERIA FOTO CONTINO',
       'TORTILLERIAS TRADICIONALES', 'SUPERMERCADO GONZALEZ', 'ALSUPER',
       'SANBORN S HNOS.', 'RADIOSHACK', 'FARMACIAS DE SIMILARES',
       'MI BODEGA AURRERA', 'MIXUP', 'OFFICE DEPOT', 'SUPER CHEDRAUI',
       'OFFICE MAX'], dtype=object)
```

In [ ]:
```
#Contar registros por cadena comercial
registros_cadenas = electrodomesticos["cadenacomercial"].value_counts()
```

In [ ]:
```
#Hacerlo como dataframe
df_registros_cadenas = pd.DataFrame(registros_cadenas)
#Crear columna con porcentaje
df_registros_cadenas["porcentaje"] = df_registros_cadenas["cadenacomercial"]/df_registros_cadenas["cadenacomercial"].sum()*100
```

In [ ]:
```
#Mostrar todas las filas
#pd.set_option('display.max_rows', None)
df_registros_cadenas
```

Out[ ]:

	cadenacomercial	porcentaje
COPPEL	552038	20.595213
LIVERPOOL	276152	10.302568
WAL-MART	252691	9.427295
ELEKTRA	222712	8.308850
HIPERMERCADO SORIANA	195203	7.282555
...	...	...
SANBORN S HNOS.	13	0.000485
MERCADO SORIANA EXPRESS	7	0.000261
SUPERMERCADOS SANTA FE	6	0.000224
JUGUETRON	5	0.000187
ALSUPER	2	0.000075

89 rows × 2 columns

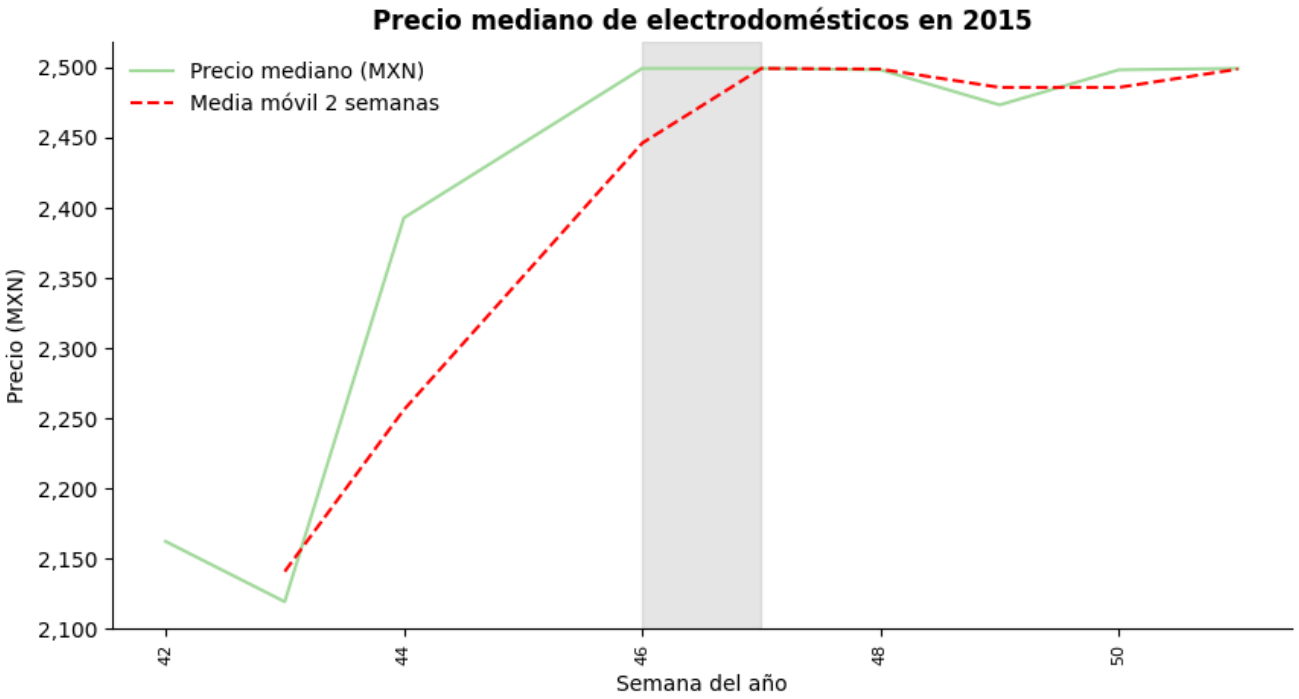
In [ ]:
```
for period in periods:
        start_date, end_date = period.split("_")
        start_week = datetime.datetime.strptime(start_date, "%Y-%m-%d").isocalendar()[1]
        end_week = datetime.datetime.strptime(end_date, "%Y-%m-%d").isocalendar()[1]
        print("Periodo "+str(start_week)+"-"+str(end_week))
```

```
Periodo 46-47
Periodo 46-47
Periodo 46-47
Periodo 46-47
Periodo 46-47
Periodo 46-47
Periodo 45-46
Periodo 46-47
```

In [ ]:
```
#Graficar precio medio de electrodomésticos para cada año. Quiero el registro de cada semana. Utilizar la función elementos_grafica()
for year in electrodomesticos["fecharegistro"].dt.year.unique():

    electrodomesticos_year = electrodomesticos[electrodomesticos["fecharegistro"].dt.year==year]
    electrodomesticos_year = electrodomesticos_year.groupby("semana_registro").median()
    electrodomesticos_max = electrodomesticos_year["precio"].max()
    electrodomesticos_min = electrodomesticos_year["precio"].min()
    electrodomesticos_year.plot(y="precio",figsize=(10,5),color="#a1d99b")
    #Gráficar max y min
    #plt.axhline(electrodomesticos_max, color="r", linestyle="--")
    #plt.axhline(electrodomesticos_min, color="r", linestyle="--")
    #media movil de 4 semanas
    electrodomesticos_year["precio"].rolling(2).median().plot(figsize=(10,5),style="r--")
    #Precio máximo
    #Incluir área de Buen Fin con base en la lista periods. Poner el área en gris con transparencia 0.2 y etiquetarla con "Buen Fin"
```

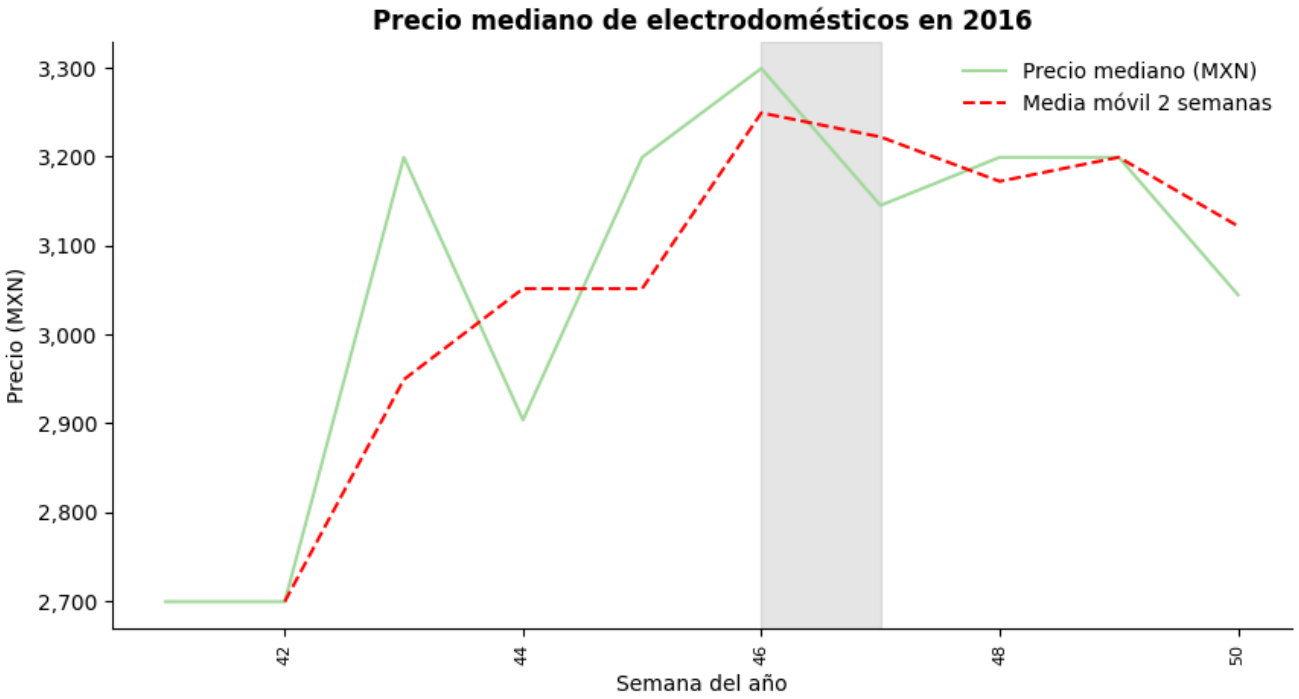
```
    for period in periods:
        start_date, end_date = period.split("_")
        start_week = datetime.datetime.strptime(start_date, "%Y-%m-%d").isocalendar()[1]
        end_week = datetime.datetime.strptime(end_date, "%Y-%m-%d").isocalendar()[1]
        if year == int(start_date[:4]):
            plt.axvspan(start_week, end_week, alpha=0.2, color='grey')
            #plt.text(start_week+1, (electrodomesticos_year["precio"].min() + electrodomesticos_year["precio"].max())/2, "Buen Fin", size=8, weight="bold")


            # Add titles, labels and legend
    plt.xticks(rotation=90, size=8)
    plt.grid(False)
    plt.title("Precio mediano de electrodomésticos en "+str(year), size=12, weight="bold")
    plt.xlabel("Semana del año")
    plt.ylabel("Precio (MXN)")
    plt.legend(['Precio mediano (MXN)','Media móvil 2 semanas'], frameon=False)
        #Eliminar los bordes
    plt.gca().spines['top'].set_visible(False)
    plt.gca().spines['right'].set_visible(False)
    #Eje y con separadores de miles
    plt.gca().get_yaxis().set_major_formatter(plt.FuncFormatter(lambda x, loc: "{:,}".format(int(x))))
    #Añadir fuente

    plt.text(0.22, -0.15, "Nota: El área sombreada corresponde al período del Buen Fin de cada año", size=8, ha="center", transform=plt.gca().transAxes)
    plt.text(0.5, -0.2, "Fuente: Secretaría de Economía. Dirección General de Competitividad y Competencia con base en información de Profeco. Quién es quién e
    #Salvar la gráfica
    plt.savefig("profeco/precio_med_elect_sem_"+str(year)+".png", bbox_inches="tight",transparent=True)
```

```
C:\Users\claudio.pacheco\AppData\Local\Temp\ipykernel_12172\2943616951.py:5: FutureWarning: The default value of numeric_only in DataFrameGroupBy.median is depr
ecated. In a future version, numeric_only will default to False. Either specify numeric_only or select only columns which should be valid for the function.
  electrodomesticos_year = electrodomesticos_year.groupby("semana_registro").median()
C:\Users\claudio.pacheco\AppData\Local\Temp\ipykernel_12172\2943616951.py:5: FutureWarning: The default value of numeric_only in DataFrameGroupBy.median is depr
ecated. In a future version, numeric_only will default to False. Either specify numeric_only or select only columns which should be valid for the function.
  electrodomesticos_year = electrodomesticos_year.groupby("semana_registro").median()
C:\Users\claudio.pacheco\AppData\Local\Temp\ipykernel_12172\2943616951.py:5: FutureWarning: The default value of numeric_only in DataFrameGroupBy.median is depr
ecated. In a future version, numeric_only will default to False. Either specify numeric_only or select only columns which should be valid for the function.
  electrodomesticos_year = electrodomesticos_year.groupby("semana_registro").median()
C:\Users\claudio.pacheco\AppData\Local\Temp\ipykernel_12172\2943616951.py:5: FutureWarning: The default value of numeric_only in DataFrameGroupBy.median is depr
ecated. In a future version, numeric_only will default to False. Either specify numeric_only or select only columns which should be valid for the function.
  electrodomesticos_year = electrodomesticos_year.groupby("semana_registro").median()
C:\Users\claudio.pacheco\AppData\Local\Temp\ipykernel_12172\2943616951.py:5: FutureWarning: The default value of numeric_only in DataFrameGroupBy.median is depr
ecated. In a future version, numeric_only will default to False. Either specify numeric_only or select only columns which should be valid for the function.
  electrodomesticos_year = electrodomesticos_year.groupby("semana_registro").median()
C:\Users\claudio.pacheco\AppData\Local\Temp\ipykernel_12172\2943616951.py:5: FutureWarning: The default value of numeric_only in DataFrameGroupBy.median is depr
ecated. In a future version, numeric_only will default to False. Either specify numeric_only or select only columns which should be valid for the function.
  electrodomesticos_year = electrodomesticos_year.groupby("semana_registro").median()
C:\Users\claudio.pacheco\AppData\Local\Temp\ipykernel_12172\2943616951.py:5: FutureWarning: The default value of numeric_only in DataFrameGroupBy.median is depr
ecated. In a future version, numeric_only will default to False. Either specify numeric_only or select only columns which should be valid for the function.
  electrodomesticos_year = electrodomesticos_year.groupby("semana_registro").median()
C:\Users\claudio.pacheco\AppData\Local\Temp\ipykernel_12172\2943616951.py:5: FutureWarning: The default value of numeric_only in DataFrameGroupBy.median is depr
ecated. In a future version, numeric_only will default to False. Either specify numeric_only or select only columns which should be valid for the function.
  electrodomesticos_year = electrodomesticos_year.groupby("semana_registro").median()
```



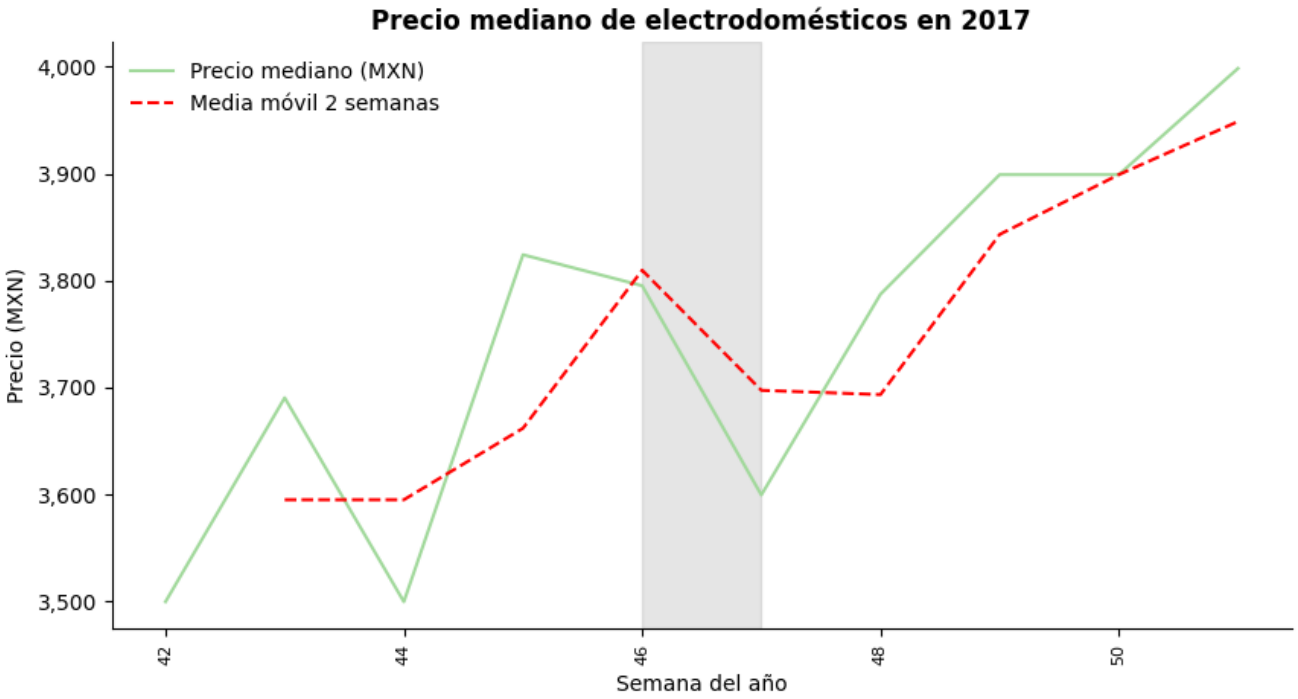
Nota: El área sombreada corresponde al período del Buen Fin de cada año

Fuente: Secretaría de Economía. Dirección General de Competitividad y Competencia con base en información de Profeco. Quién es quién en los precios



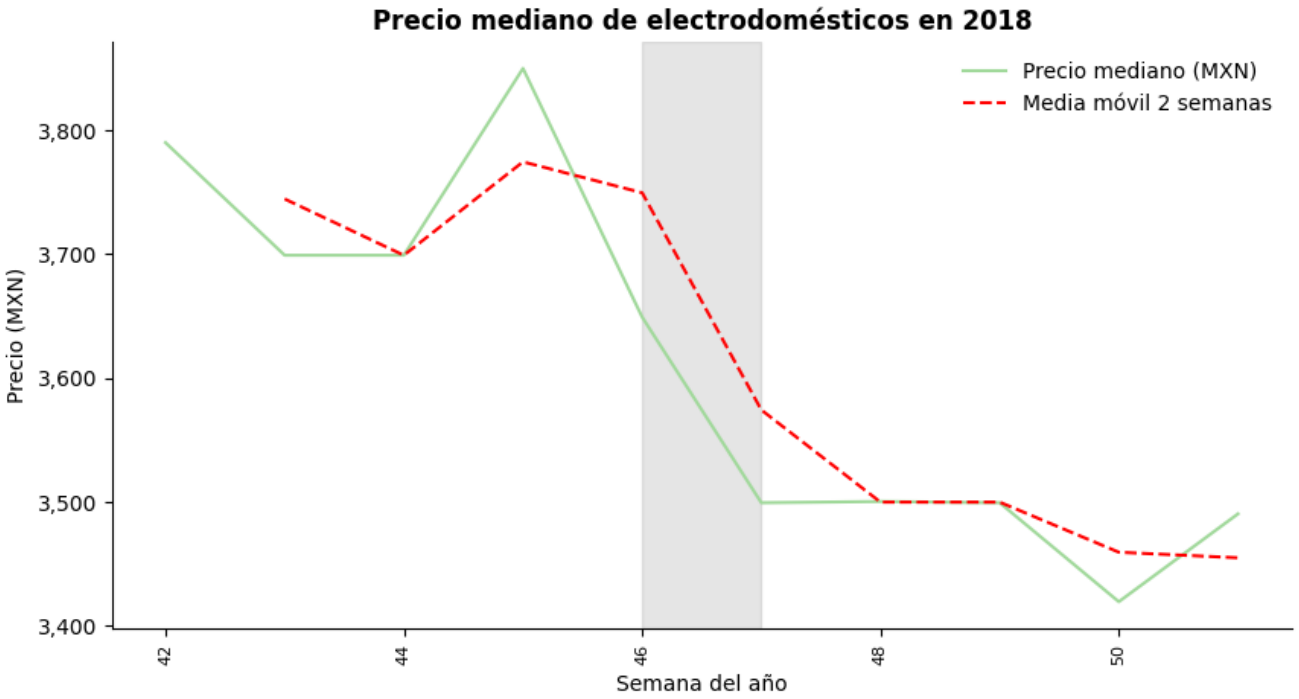
Nota: El área sombreada corresponde al período del Buen Fin de cada año

Fuente: Secretaría de Economía. Dirección General de Competitividad y Competencia con base en información de Profeco. Quién es quién en los precios



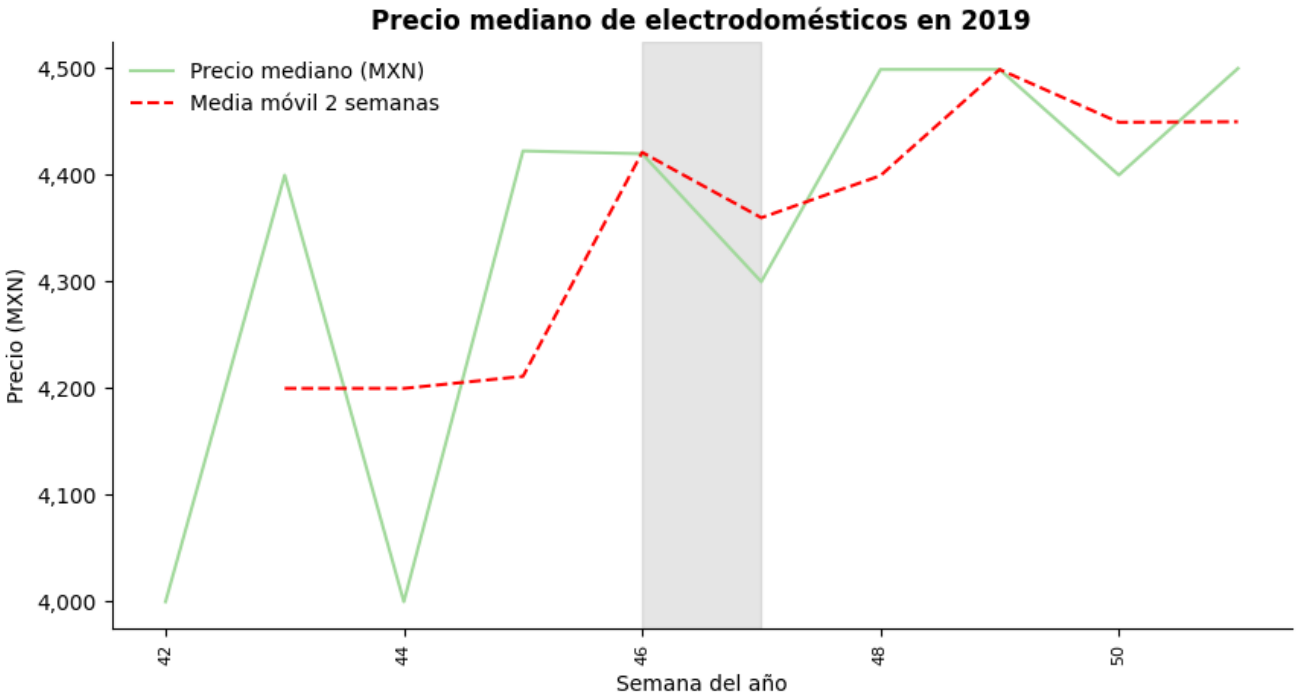
Nota: El área sombreada corresponde al período del Buen Fin de cada año

Fuente: Secretaría de Economía. Dirección General de Competitividad y Competencia con base en información de Profeco. Quién es quién en los precios



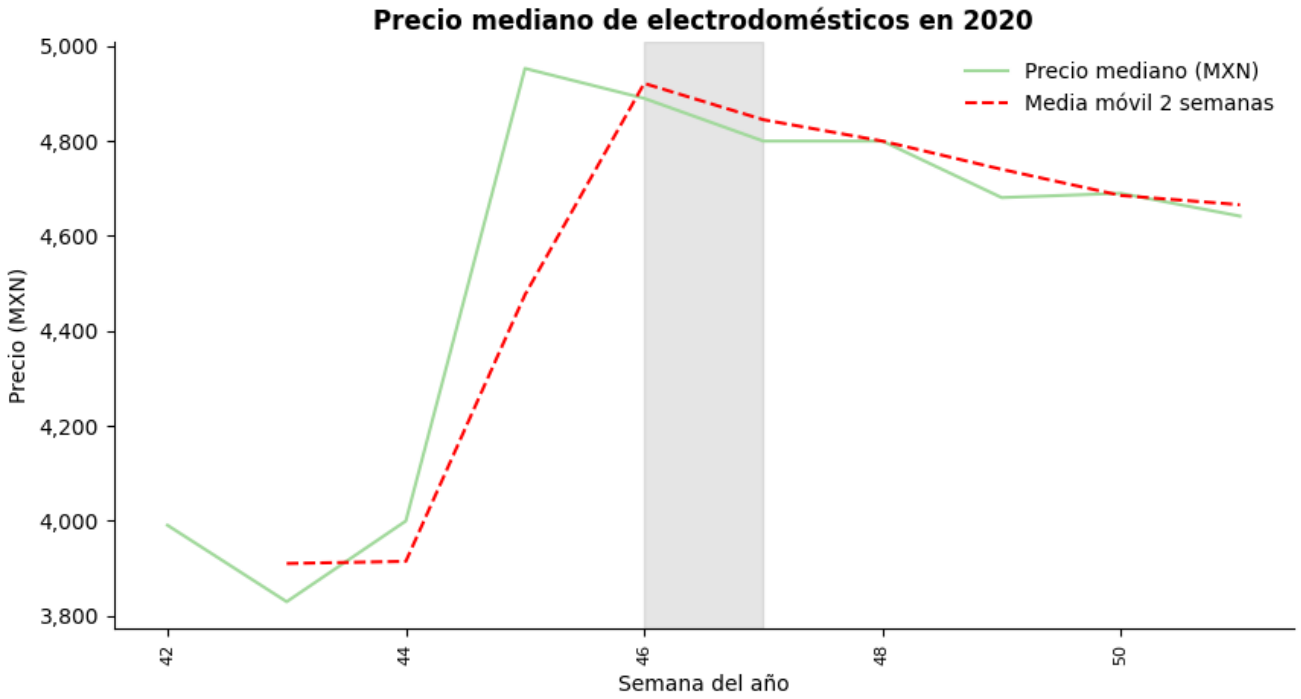
Nota: El área sombreada corresponde al período del Buen Fin de cada año

Fuente: Secretaría de Economía. Dirección General de Competitividad y Competencia con base en información de Profeco. Quién es quién en los precios



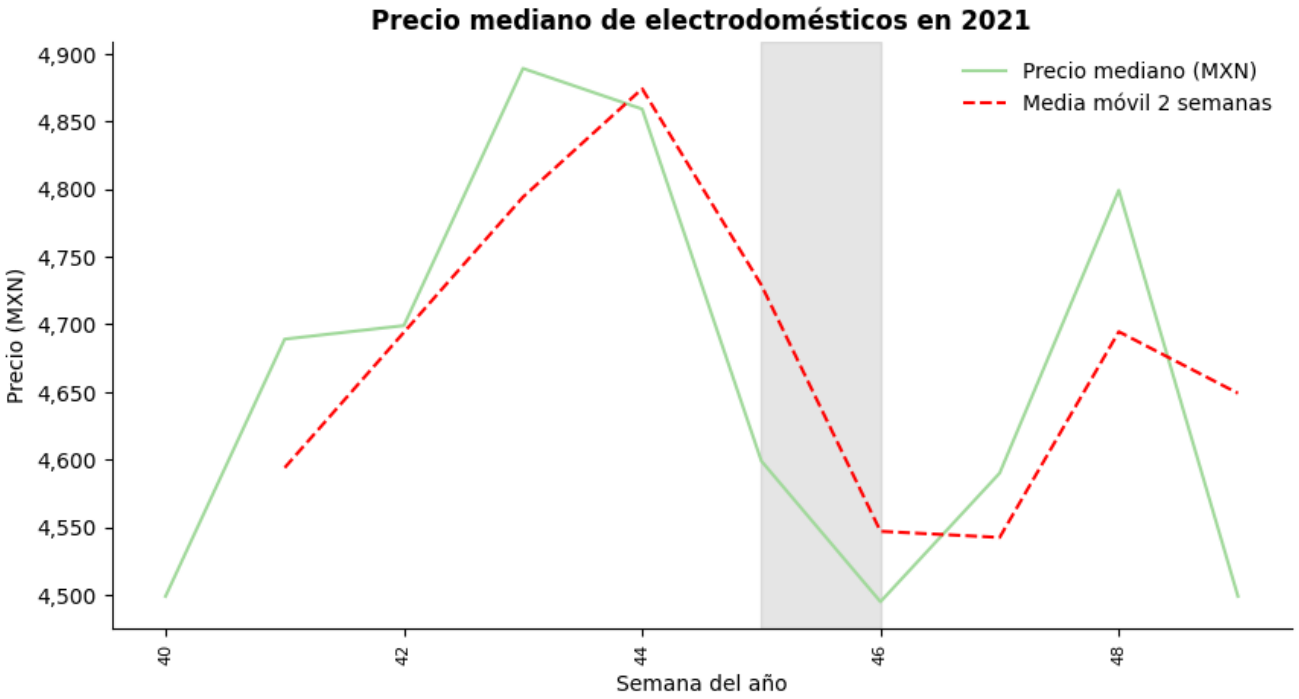
Nota: El área sombreada corresponde al período del Buen Fin de cada año

Fuente: Secretaría de Economía. Dirección General de Competitividad y Competencia con base en información de Profeco. Quién es quién en los precios



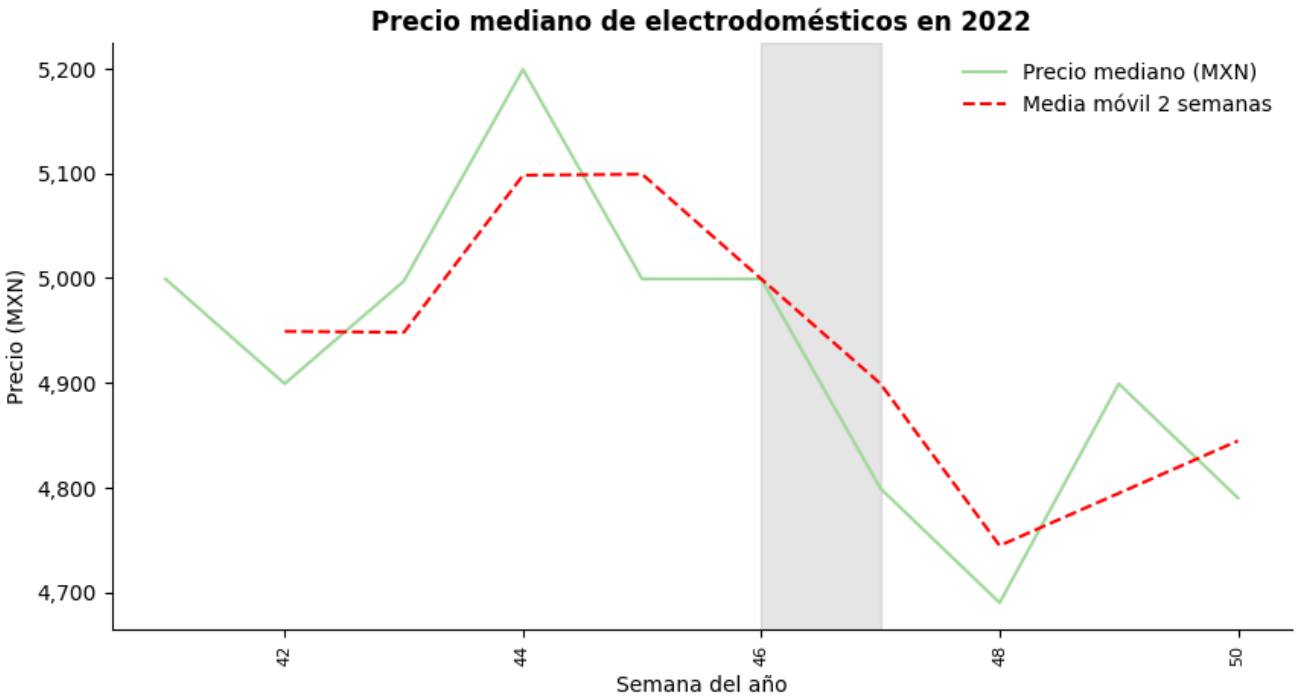
Nota: El área sombreada corresponde al período del Buen Fin de cada año

Fuente: Secretaría de Economía. Dirección General de Competitividad y Competencia con base en información de Profeco. Quién es quién en los precios



Nota: El área sombreada corresponde al período del Buen Fin de cada año

Fuente: Secretaría de Economía. Dirección General de Competitividad y Competencia con base en información de Profeco. Quién es quién en los precios



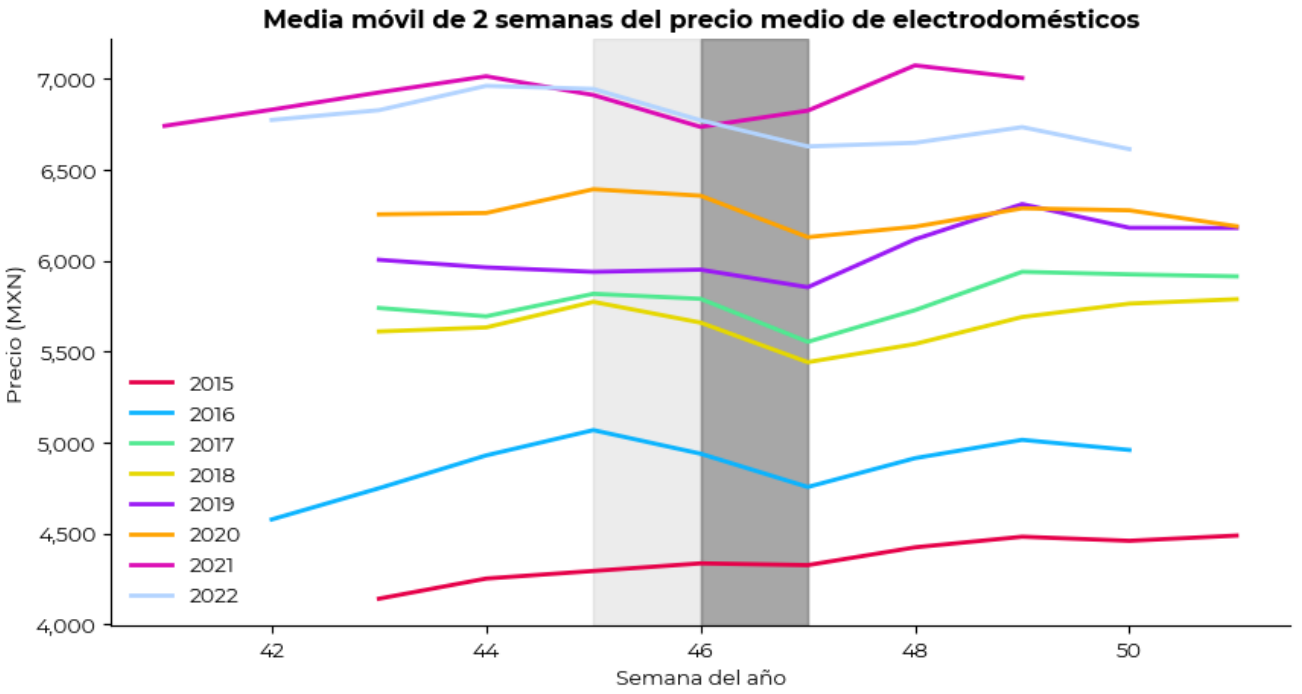
Nota: El área sombreada corresponde al período del Buen Fin de cada año

Fuente: Secretaría de Economía. Dirección General de Competitividad y Competencia con base en información de Profeco. Quién es quién en los precios

In [ ]:
```
import matplotlib.pyplot as plt

colores=["#e60049", "#0bb4ff", "#50e991", "#e6d800", "#9b19f5", "#ffa300", "#dc0ab4", "#b3d4ff", "#00bfa0"]
plt.rcParams["axes.prop_cycle"] = plt.cycler("color", colores)


plt.rcParams["font.family"] = "Montserrat"


# create the figure and axis objects
fig, ax = plt.subplots(figsize=(10, 5))

# loop over years and add each line to the same axis
for year in electrodomesticos["fecharegistro"].dt.year.unique():

    electrodomesticos_year = electrodomesticos[electrodomesticos["fecharegistro"].dt.year==year]
    electrodomesticos_year = electrodomesticos_year.groupby("semana_registro").mean()
    #Media movil de 2 semanas
    #electrodomesticos_year["precio"].rolling(2).mean().plot(figsize=(10,5),style="r--")
    ax.plot(electrodomesticos_year["precio"].rolling(2).mean(), label=str(year), linestyle="-", linewidth=2)
    #x = electrodomesticos_year.index[-1]
    #y = electrodomesticos_year["precio"].rolling(2).mean().iloc[-1]
    #ax.text(x, y, str(year), ha="left", va="center", fontsize=8)
```

```
    # add the year label at the end of the line
    #ax.plot([x, x], [y, y-2000], color="gray", linestyle="--", linewidth=1)

#Añadir área de Buen Fin con base en la lista periods. Poner el área en gris con transparencia 0.2 y etiquetarla con "Buen Fin"
for period in periods:
    start_date, end_date = period.split("_")
    start_week = datetime.datetime.strptime(start_date, "%Y-%m-%d").isocalendar()[1]
    end_week = datetime.datetime.strptime(end_date, "%Y-%m-%d").isocalendar()[1]
    ax.axvspan(start_week, end_week, alpha=0.15, color='grey')
    #plt.text(start_week+1, (electrodomesticos_year["precio"].min() + electrodomesticos_year["precio"].max())/2, "Buen Fin", size=8, weight="bold")


# add titles, labels and legend
ax.set_xlabel("Semana del año")
ax.set_ylabel("Precio (MXN)")
ax.set_title("Media móvil de 2 semanas del precio medio de electrodomésticos", size=12, weight="bold")
ax.legend(frameon=False)
# remove the top and right spines
ax.spines['top'].set_visible(False)
ax.spines['right'].set_visible(False)

# format the y-axis with separators
ax.yaxis.set_major_formatter(plt.FuncFormatter(lambda x, loc: "{:,}".format(int(x))))

#Fuente de los datos
plt.text(0.22, -0.15, "Nota: El área sombreada corresponde al período del Buen Fin de cada año", size=8, ha="center", transform=plt.gca().transAxes)
plt.text(0.5, -0.2, "Fuente: Secretaría de Economía. Dirección General de Competitividad y Competencia con base en información de Profeco. Quién es quién en lo
#Salvar la gráfica
plt.savefig("profeco/precio_med_elect_sem_media_movil.png", bbox_inches="tight",transparent=True)
```

```
C:\Users\claudio.pacheco\AppData\Local\Temp\ipykernel_12172\2499541933.py:17: FutureWarning: The default value of numeric_only in DataFrameGroupBy.mean is depre
cated. In a future version, numeric_only will default to False. Either specify numeric_only or select only columns which should be valid for the function.
  electrodomesticos_year = electrodomesticos_year.groupby("semana_registro").mean()
C:\Users\claudio.pacheco\AppData\Local\Temp\ipykernel_12172\2499541933.py:17: FutureWarning: The default value of numeric_only in DataFrameGroupBy.mean is depre
cated. In a future version, numeric_only will default to False. Either specify numeric_only or select only columns which should be valid for the function.
  electrodomesticos_year = electrodomesticos_year.groupby("semana_registro").mean()
C:\Users\claudio.pacheco\AppData\Local\Temp\ipykernel_12172\2499541933.py:17: FutureWarning: The default value of numeric_only in DataFrameGroupBy.mean is depre
cated. In a future version, numeric_only will default to False. Either specify numeric_only or select only columns which should be valid for the function.
  electrodomesticos_year = electrodomesticos_year.groupby("semana_registro").mean()
C:\Users\claudio.pacheco\AppData\Local\Temp\ipykernel_12172\2499541933.py:17: FutureWarning: The default value of numeric_only in DataFrameGroupBy.mean is depre
cated. In a future version, numeric_only will default to False. Either specify numeric_only or select only columns which should be valid for the function.
  electrodomesticos_year = electrodomesticos_year.groupby("semana_registro").mean()
C:\Users\claudio.pacheco\AppData\Local\Temp\ipykernel_12172\2499541933.py:17: FutureWarning: The default value of numeric_only in DataFrameGroupBy.mean is depre
cated. In a future version, numeric_only will default to False. Either specify numeric_only or select only columns which should be valid for the function.
  electrodomesticos_year = electrodomesticos_year.groupby("semana_registro").mean()
C:\Users\claudio.pacheco\AppData\Local\Temp\ipykernel_12172\2499541933.py:17: FutureWarning: The default value of numeric_only in DataFrameGroupBy.mean is depre
cated. In a future version, numeric_only will default to False. Either specify numeric_only or select only columns which should be valid for the function.
  electrodomesticos_year = electrodomesticos_year.groupby("semana_registro").mean()
C:\Users\claudio.pacheco\AppData\Local\Temp\ipykernel_12172\2499541933.py:17: FutureWarning: The default value of numeric_only in DataFrameGroupBy.mean is depre
cated. In a future version, numeric_only will default to False. Either specify numeric_only or select only columns which should be valid for the function.
  electrodomesticos_year = electrodomesticos_year.groupby("semana_registro").mean()
C:\Users\claudio.pacheco\AppData\Local\Temp\ipykernel_12172\2499541933.py:17: FutureWarning: The default value of numeric_only in DataFrameGroupBy.mean is depre
cated. In a future version, numeric_only will default to False. Either specify numeric_only or select only columns which should be valid for the function.
  electrodomesticos_year = electrodomesticos_year.groupby("semana_registro").mean()
```



Nota: El área sombreada corresponde al período del Buen Fin de cada año

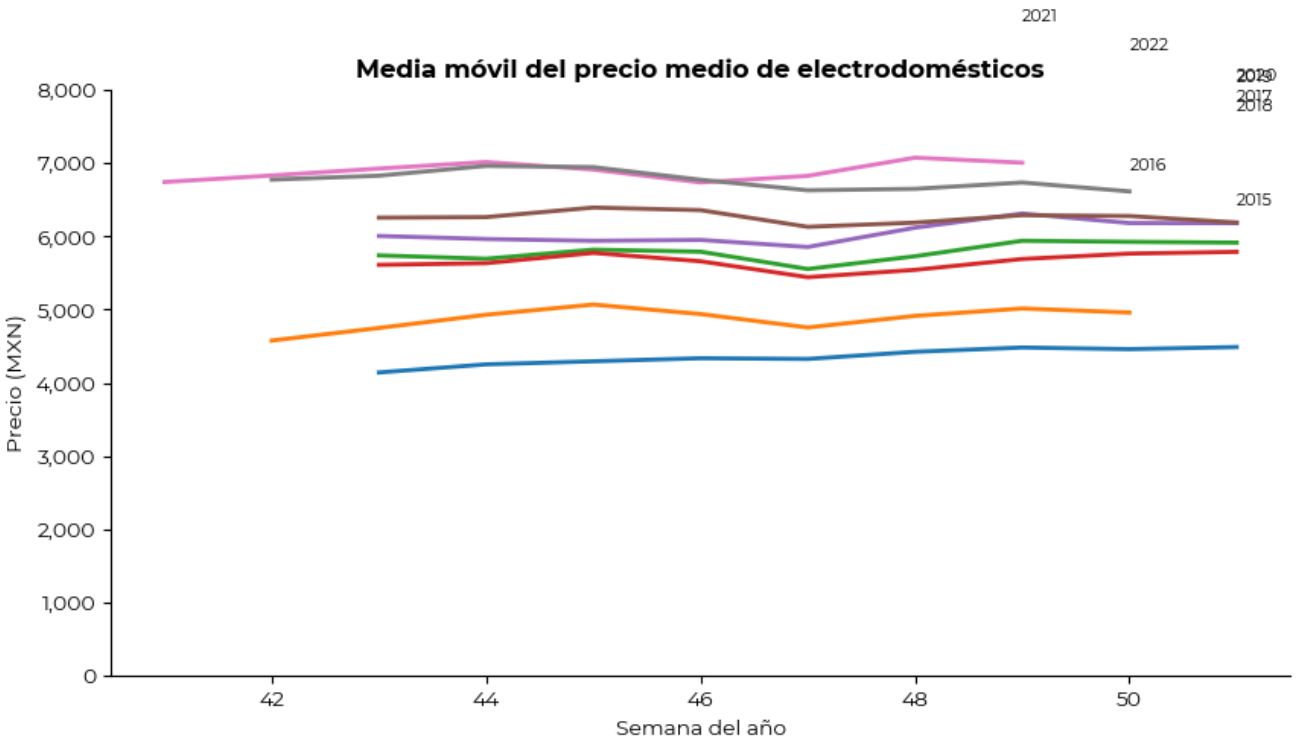
Fuente: Secretaría de Economía. Dirección General de Competitividad y Competencia con base en información de Profeco. Quién es quién en los precios

In [ ]:
```
import matplotlib.pyplot as plt
plt.rcParams["font.family"] = "Montserrat"


# create the figure and axis objects
fig, ax = plt.subplots(figsize=(10, 5))

# set the y-axis limits
ax.set_ylim(bottom=0, top=8000)

# loop over years and add each line to the same axis
for year in electrodomesticos["fecharegistro"].dt.year.unique():

    electrodomesticos_year = electrodomesticos[electrodomesticos["fecharegistro"].dt.year==year]
    electrodomesticos_year = electrodomesticos_year.groupby("semana_registro").mean()

    # plot the line and add the year label
    ax.plot(electrodomesticos_year["precio"].rolling(2).mean(), label=str(year), linestyle="-", linewidth=2)
    x = electrodomesticos_year.index[-1]
    y = electrodomesticos_year["precio"].rolling(2).mean().iloc[-1]
    ax.text(x, y+2000, str(year), ha="left", va="center", fontsize=8)


# add titles and labels
```

```
ax.set_xlabel("Semana del año")
ax.set_ylabel("Precio (MXN)")
ax.set_title("Media móvil del precio medio de electrodomésticos", size=12, weight="bold")

# remove the top and right spines
ax.spines['top'].set_visible(False)
ax.spines['right'].set_visible(False)

# format the y-axis with separators
ax.yaxis.set_major_formatter(plt.FuncFormatter(lambda x, loc: "{:,}".format(int(x))))

#Fuente de los datos
plt.text(0.5, -0.2, "Fuente: Secretaría de Economía. Dirección General de Competitividad y Competencia con base en información de Profeco. Quién es quién en lo
```

```
C:\Users\claudio.pacheco\AppData\Local\Temp\ipykernel_12172\261648785.py:15: FutureWarning: The default value of numeric_only in DataFrameGroupBy.mean is deprec
ated. In a future version, numeric_only will default to False. Either specify numeric_only or select only columns which should be valid for the function.
  electrodomesticos_year = electrodomesticos_year.groupby("semana_registro").mean()
C:\Users\claudio.pacheco\AppData\Local\Temp\ipykernel_12172\261648785.py:15: FutureWarning: The default value of numeric_only in DataFrameGroupBy.mean is deprec
ated. In a future version, numeric_only will default to False. Either specify numeric_only or select only columns which should be valid for the function.
  electrodomesticos_year = electrodomesticos_year.groupby("semana_registro").mean()
C:\Users\claudio.pacheco\AppData\Local\Temp\ipykernel_12172\261648785.py:15: FutureWarning: The default value of numeric_only in DataFrameGroupBy.mean is deprec
ated. In a future version, numeric_only will default to False. Either specify numeric_only or select only columns which should be valid for the function.
  electrodomesticos_year = electrodomesticos_year.groupby("semana_registro").mean()
C:\Users\claudio.pacheco\AppData\Local\Temp\ipykernel_12172\261648785.py:15: FutureWarning: The default value of numeric_only in DataFrameGroupBy.mean is deprec
ated. In a future version, numeric_only will default to False. Either specify numeric_only or select only columns which should be valid for the function.
  electrodomesticos_year = electrodomesticos_year.groupby("semana_registro").mean()
C:\Users\claudio.pacheco\AppData\Local\Temp\ipykernel_12172\261648785.py:15: FutureWarning: The default value of numeric_only in DataFrameGroupBy.mean is deprec
ated. In a future version, numeric_only will default to False. Either specify numeric_only or select only columns which should be valid for the function.
  electrodomesticos_year = electrodomesticos_year.groupby("semana_registro").mean()
C:\Users\claudio.pacheco\AppData\Local\Temp\ipykernel_12172\261648785.py:15: FutureWarning: The default value of numeric_only in DataFrameGroupBy.mean is deprec
ated. In a future version, numeric_only will default to False. Either specify numeric_only or select only columns which should be valid for the function.
  electrodomesticos_year = electrodomesticos_year.groupby("semana_registro").mean()
C:\Users\claudio.pacheco\AppData\Local\Temp\ipykernel_12172\261648785.py:15: FutureWarning: The default value of numeric_only in DataFrameGroupBy.mean is deprec
ated. In a future version, numeric_only will default to False. Either specify numeric_only or select only columns which should be valid for the function.
  electrodomesticos_year = electrodomesticos_year.groupby("semana_registro").mean()
C:\Users\claudio.pacheco\AppData\Local\Temp\ipykernel_12172\261648785.py:15: FutureWarning: The default value of numeric_only in DataFrameGroupBy.mean is deprec
ated. In a future version, numeric_only will default to False. Either specify numeric_only or select only columns which should be valid for the function.
  electrodomesticos_year = electrodomesticos_year.groupby("semana_registro").mean()
```

```
Out[ ]: Text(0.5, -0.2, 'Fuente: Secretaría de Economía. Dirección General de Competitividad y Competencia con base en información de Profeco. Quién es quién en los p
recios')
```

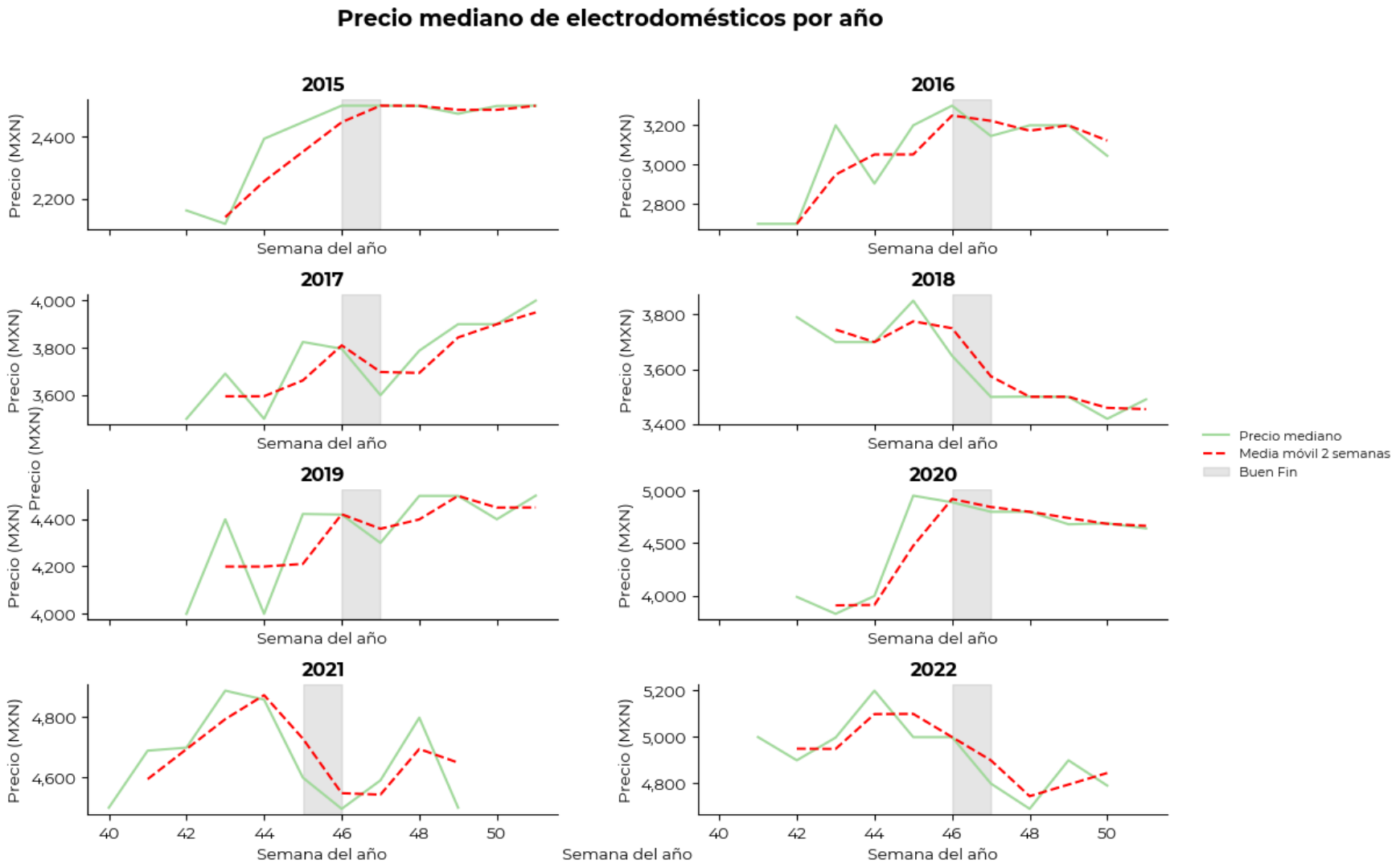


Fuente: Secretaría de Economía. Dirección General de Competitividad y Competencia con base en información de Profeco. Quién es quién en los precios

In [ ]:
```
import matplotlib.pyplot as plt
import numpy as np
import datetime

plt.rcParams["font.family"] = "Montserrat"
# Create a figure with subplots for each year
years = electrodomesticos["fecharegistro"].dt.year.unique()
nrows = np.ceil(len(years) / 2).astype(int)
fig, axs = plt.subplots(nrows=nrows, ncols=2, figsize=(12, 2*nrows), sharex=True, sharey=False)
# Loop over years and plot data on corresponding subplot
for i, year in enumerate(years):
    row = i // 2
    col = i % 2
    ax = axs[row, col]

    # Subset data for current year
    electrodomesticos_year = electrodomesticos[electrodomesticos["fecharegistro"].dt.year == year]
    electrodomesticos_year = electrodomesticos_year.groupby("semana_registro").median()

    # Plot median price and rolling mean
    ax.plot(electrodomesticos_year["precio"], color="#a1d99b", label="Precio mediano")
    ax.plot(electrodomesticos_year["precio"].rolling(2).median(), color="red", linestyle="--", label="Media móvil 2 semanas")
```

```
    # Add shaded area for Buen Fin
    for period in periods:
        start_date, end_date = period.split("_")
        start_week = datetime.datetime.strptime(start_date, "%Y-%m-%d").isocalendar()[1]
        end_week = datetime.datetime.strptime(end_date, "%Y-%m-%d").isocalendar()[1]
        if year == int(start_date[:4]):
            ax.axvspan(start_week, end_week, alpha=0.2, color='grey', label="Buen Fin")

    # Set title and labels
    ax.set_title(f"{year}", size=12, weight="bold")
    ax.set_xlabel("Semana del año")
    ax.set_ylabel("Precio (MXN)")
    plt.suptitle("Precio mediano de electrodomésticos por año", size=14, weight="bold")

    # Format y-axis ticks with thousands separator
    ax.get_yaxis().set_major_formatter(plt.FuncFormatter(lambda x, loc: "{:,}".format(int(x))))

    # Add legend and remove spines
    if i == 5:
        ax.legend(frameon=False, loc="lower right",fontsize=8,bbox_to_anchor=(1.5, 1))
    #ax.legend(frameon=False, loc="lower right",fontsize=6)
    ax.spines['top'].set_visible(False)
    ax.spines['right'].set_visible(False)

# Add common x- and y-axis labels and save the figure
fig.add_subplot(111, frameon=False)
plt.tick_params(labelcolor='none', top=False, bottom=False, left=False, right=False)
plt.grid(False)
plt.xlabel("Semana del año")
plt.ylabel("Precio (MXN)")
fig.subplots_adjust(wspace=0.3, hspace=0.5)
#Fuente de los datos
plt.text(0.5, -0.1, "Fuente: Secretaría de Economía. Dirección General de Competitividad y Competencia con base en información de Profeco. Quién es quién en lo

plt.savefig("profeco/precio_med_elect_subplot.png", bbox_inches="tight",transparent=True)
```

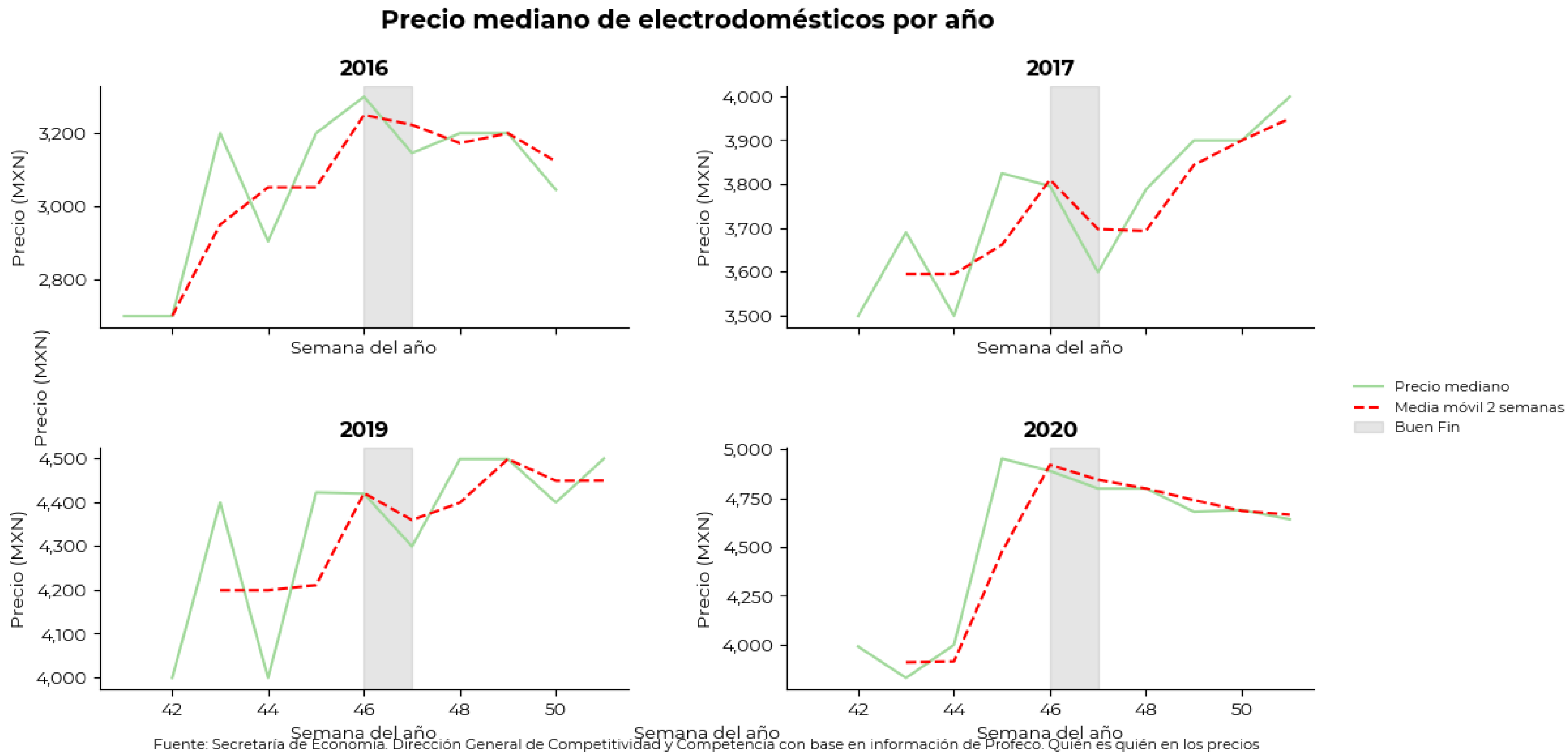
```
C:\Users\claudio.pacheco\AppData\Local\Temp\ipykernel_12172\4140956853.py:18: FutureWarning: The default value of numeric_only in DataFrameGroupBy.median is dep
recated. In a future version, numeric_only will default to False. Either specify numeric_only or select only columns which should be valid for the function.
  electrodomesticos_year = electrodomesticos_year.groupby("semana_registro").median()
C:\Users\claudio.pacheco\AppData\Local\Temp\ipykernel_12172\4140956853.py:18: FutureWarning: The default value of numeric_only in DataFrameGroupBy.median is dep
recated. In a future version, numeric_only will default to False. Either specify numeric_only or select only columns which should be valid for the function.
  electrodomesticos_year = electrodomesticos_year.groupby("semana_registro").median()
C:\Users\claudio.pacheco\AppData\Local\Temp\ipykernel_12172\4140956853.py:18: FutureWarning: The default value of numeric_only in DataFrameGroupBy.median is dep
recated. In a future version, numeric_only will default to False. Either specify numeric_only or select only columns which should be valid for the function.
  electrodomesticos_year = electrodomesticos_year.groupby("semana_registro").median()
C:\Users\claudio.pacheco\AppData\Local\Temp\ipykernel_12172\4140956853.py:18: FutureWarning: The default value of numeric_only in DataFrameGroupBy.median is dep
recated. In a future version, numeric_only will default to False. Either specify numeric_only or select only columns which should be valid for the function.
  electrodomesticos_year = electrodomesticos_year.groupby("semana_registro").median()
C:\Users\claudio.pacheco\AppData\Local\Temp\ipykernel_12172\4140956853.py:18: FutureWarning: The default value of numeric_only in DataFrameGroupBy.median is dep
recated. In a future version, numeric_only will default to False. Either specify numeric_only or select only columns which should be valid for the function.
  electrodomesticos_year = electrodomesticos_year.groupby("semana_registro").median()
C:\Users\claudio.pacheco\AppData\Local\Temp\ipykernel_12172\4140956853.py:18: FutureWarning: The default value of numeric_only in DataFrameGroupBy.median is dep
recated. In a future version, numeric_only will default to False. Either specify numeric_only or select only columns which should be valid for the function.
  electrodomesticos_year = electrodomesticos_year.groupby("semana_registro").median()
C:\Users\claudio.pacheco\AppData\Local\Temp\ipykernel_12172\4140956853.py:18: FutureWarning: The default value of numeric_only in DataFrameGroupBy.median is dep
recated. In a future version, numeric_only will default to False. Either specify numeric_only or select only columns which should be valid for the function.
  electrodomesticos_year = electrodomesticos_year.groupby("semana_registro").median()
C:\Users\claudio.pacheco\AppData\Local\Temp\ipykernel_12172\4140956853.py:18: FutureWarning: The default value of numeric_only in DataFrameGroupBy.median is dep
recated. In a future version, numeric_only will default to False. Either specify numeric_only or select only columns which should be valid for the function.
  electrodomesticos_year = electrodomesticos_year.groupby("semana_registro").median()
```



In [ ]:
```
import matplotlib.pyplot as plt
import numpy as np
import datetime

plt.rcParams["font.family"] = "Montserrat"
# Create a figure with subplots for 2016,2017,2019,2020
years = [2016,2017,2019,2020]
nrows = np.ceil(len(years) / 2).astype(int)
fig, axs = plt.subplots(nrows=nrows, ncols=2, figsize=(12, 3*nrows), sharex=True, sharey=False)
# Loop over years and plot data on corresponding subplot
```

```
for i, year in enumerate(years):
    row = i // 2
    col = i % 2
    ax = axs[row, col]

    # Subset data for current year
    electrodomesticos_year = electrodomesticos[electrodomesticos["fecharegistro"].dt.year == year]
    electrodomesticos_year = electrodomesticos_year.groupby("semana_registro").median()

    # Plot median price and rolling mean
    ax.plot(electrodomesticos_year["precio"], color="#a1d99b", label="Precio mediano")
    ax.plot(electrodomesticos_year["precio"].rolling(2).median(), color="red", linestyle="--", label="Media móvil 2 semanas")

    # Add shaded area for Buen Fin
    for period in periods:
        start_date, end_date = period.split("_")
        start_week = datetime.datetime.strptime(start_date, "%Y-%m-%d").isocalendar()[1]
        end_week = datetime.datetime.strptime(end_date, "%Y-%m-%d").isocalendar()[1]
        if year == int(start_date[:4]):
            ax.axvspan(start_week, end_week, alpha=0.2, color='grey', label="Buen Fin")

    # Set title and labels
    ax.set_title(f"{year}", size=12, weight="bold")
    ax.set_xlabel("Semana del año")
    ax.set_ylabel("Precio (MXN)")
    plt.suptitle("Precio mediano de electrodomésticos por año", size=14, weight="bold")

    # Format y-axis ticks with thousands separator
    ax.get_yaxis().set_major_formatter(plt.FuncFormatter(lambda x, loc: "{:,}".format(int(x))))

    # Add legend and remove spines
    if i == 3:
        ax.legend(frameon=False, loc="lower right",fontsize=8,bbox_to_anchor=(1.5, 1))
    #ax.legend(frameon=False, loc="lower right",fontsize=6)
    ax.spines['top'].set_visible(False)
    ax.spines['right'].set_visible(False)

# Add common x- and y-axis labels and save the figure
fig.add_subplot(111, frameon=False)
plt.tick_params(labelcolor='none', top=False, bottom=False, left=False, right=False)
plt.grid(False)
plt.xlabel("Semana del año")
plt.ylabel("Precio (MXN)")
fig.subplots_adjust(wspace=0.3, hspace=0.5)
#Fuente de los datos
plt.text(0.5, -0.1, "Fuente: Secretaría de Economía. Dirección General de Competitividad y Competencia con base en información de Profeco. Quién es quién en lo
plt.savefig("profeco/precio_med_elect_subplot_1.png", bbox_inches="tight",transparent=True)
```

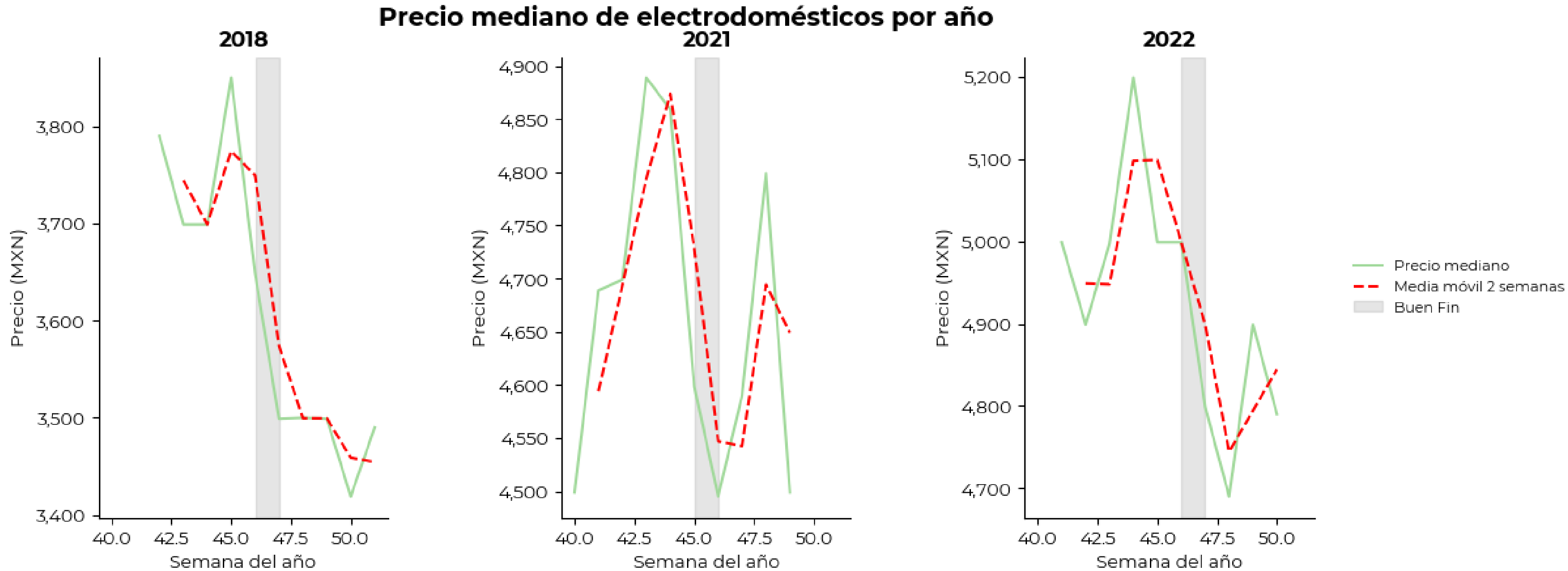
```
C:\Users\claudio.pacheco\AppData\Local\Temp\ipykernel_12172\2824188343.py:18: FutureWarning: The default value of numeric_only in DataFrameGroupBy.median is dep
recated. In a future version, numeric_only will default to False. Either specify numeric_only or select only columns which should be valid for the function.
  electrodomesticos_year = electrodomesticos_year.groupby("semana_registro").median()
C:\Users\claudio.pacheco\AppData\Local\Temp\ipykernel_12172\2824188343.py:18: FutureWarning: The default value of numeric_only in DataFrameGroupBy.median is dep
recated. In a future version, numeric_only will default to False. Either specify numeric_only or select only columns which should be valid for the function.
  electrodomesticos_year = electrodomesticos_year.groupby("semana_registro").median()
C:\Users\claudio.pacheco\AppData\Local\Temp\ipykernel_12172\2824188343.py:18: FutureWarning: The default value of numeric_only in DataFrameGroupBy.median is dep
recated. In a future version, numeric_only will default to False. Either specify numeric_only or select only columns which should be valid for the function.
  electrodomesticos_year = electrodomesticos_year.groupby("semana_registro").median()
C:\Users\claudio.pacheco\AppData\Local\Temp\ipykernel_12172\2824188343.py:18: FutureWarning: The default value of numeric_only in DataFrameGroupBy.median is dep
recated. In a future version, numeric_only will default to False. Either specify numeric_only or select only columns which should be valid for the function.
  electrodomesticos_year = electrodomesticos_year.groupby("semana_registro").median()
```



In [ ]:
```python
import matplotlib.pyplot as plt
import numpy as np
import datetime

plt.rcParams["font.family"] = "Montserrat"
# Create a figure with subplots for 2018, 2021, and 2022. One row 3 columns
years = [2018, 2021, 2022]
nrows = 1
fig, axs = plt.subplots(nrows=nrows, ncols=3, figsize=(12, 4), sharex=True, sharey=False)
```

```
# Loop over years and plot data on corresponding subplot
for i, year in enumerate(years):
    ax = axs[i]

    # Subset data for current year
    electrodomesticos_year = electrodomesticos[electrodomesticos["fecharegistro"].dt.year == year]
    electrodomesticos_year = electrodomesticos_year.groupby("semana_registro").median()

    # Plot median price and rolling mean
    ax.plot(electrodomesticos_year["precio"], color="#a1d99b", label="Precio mediano")
    ax.plot(electrodomesticos_year["precio"].rolling(2).median(), color="red", linestyle="--", label="Media móvil 2 semanas")

    # Add shaded area for Buen Fin
    for period in periods:
        start_date, end_date = period.split("_")
        start_week = datetime.datetime.strptime(start_date, "%Y-%m-%d").isocalendar()[1]
        end_week = datetime.datetime.strptime(end_date, "%Y-%m-%d").isocalendar()[1]
        if year == int(start_date[:4]):
            ax.axvspan(start_week, end_week, alpha=0.2, color='grey', label="Buen Fin")

    # Set title and labels
    ax.set_title(f"{year}", size=12, weight="bold")
    ax.set_xlabel("Semana del año")
    ax.set_ylabel("Precio (MXN)")

    # Format y-axis ticks with thousands separator
    ax.get_yaxis().set_major_formatter(plt.FuncFormatter(lambda x, loc: "{:,}".format(int(x))))

    # Add legend and remove spines
    if i == 2:
        ax.legend(frameon=False, loc="center",fontsize=8,bbox_to_anchor=(1.5, 0.5))
    #ax.legend(frameon=False, loc="lower right",fontsize=6)
    ax.spines['top'].set_visible(False)
    ax.spines['right'].set_visible(False)

# Add common x- and y-axis labels and save the figure
fig.add_subplot(111, frameon=False)
plt.tick_params(labelcolor='none', top=False, bottom=False, left=False, right=False)
plt.grid(False)
#plt.xlabel("Semana del año")
#plt.ylabel("Precio (MXN)")
fig.subplots_adjust(wspace=0.6, hspace=0.8,bottom=0)
#Fuente de los datos
plt.text(0.5, -0.2, "Fuente: Secretaría de Economía. Dirección General de Competitividad y Competencia con base en información de Profeco. Quién es quién en lo

# Set suptitle after adjusting subplots to prevent overlapping
fig.suptitle("Precio mediano de electrodomésticos por año", size=14, weight="bold")

plt.savefig("profeco/precio_med_elect_subplot_2.png", bbox_inches="tight",transparent=True)
```

```
C:\Users\claudio.pacheco\AppData\Local\Temp\ipykernel_12172\837378271.py:16: FutureWarning: The default value of numeric_only in DataFrameGroupBy.median is depr
ecated. In a future version, numeric_only will default to False. Either specify numeric_only or select only columns which should be valid for the function.
  electrodomesticos_year = electrodomesticos_year.groupby("semana_registro").median()
C:\Users\claudio.pacheco\AppData\Local\Temp\ipykernel_12172\837378271.py:16: FutureWarning: The default value of numeric_only in DataFrameGroupBy.median is depr
ecated. In a future version, numeric_only will default to False. Either specify numeric_only or select only columns which should be valid for the function.
  electrodomesticos_year = electrodomesticos_year.groupby("semana_registro").median()
C:\Users\claudio.pacheco\AppData\Local\Temp\ipykernel_12172\837378271.py:16: FutureWarning: The default value of numeric_only in DataFrameGroupBy.median is depr
ecated. In a future version, numeric_only will default to False. Either specify numeric_only or select only columns which should be valid for the function.
  electrodomesticos_year = electrodomesticos_year.groupby("semana_registro").median()
```



Fuente: Secretaría de Economía. Dirección General de Competitividad y Competencia con base en información de Profeco. Quién es quién en los precios