

Historico de Bitacoras Onomásticas del SEMEFO-DF

Análisis de datos

Las **Historico de Bitacoras Onomásticas del Servicio Médico Forense del Distrito Federal (SEMEFO-DF)**, han sido sistematizada por la Comisión Nacional de Búsqueda (CNB). Este conjunto de datos se obtuvo a través de una solicitud de acceso a la información disponible en el Portal de Transparencia del Instituto Nacional de Acceso a la Información (INA) en noviembre de 2023 y existe una copia en el portal ciudadano *datamx* de Codeando México.

El **SEMEFO-DF (Servicio Médico Forense Distrito Federal)** ahora **INCIFO-CDMx** es una institución oficial encargada de realizar autopsias y estudios forenses para esclarecer las causas de muerte, principalmente en casos de muertes violentas, sospechosas o relacionadas con hechos delictivos. Las **Bitacoras Onomásticas (BO)** son registros administrativos históricos utilizados por el SEMEFO para documentar el ingreso de cadáveres o restos humanos a sus instalaciones. Estas bitacoras contienen información detallada sobre cada ingreso, como el nombre de la persona fallecida, su edad, sexo, fecha de ingreso, institución de procedencia y diagnóstico preliminar de causa de muerte. En muchos casos, estos registros incluyen datos clave para identificar patrones históricos de mortalidad, enfermedades comunes y factores de riesgo, lo que las convierte en una fuente valiosa para estudios demográficos, epidemiológicos y de salud pública.

Podemos acceder a esta información con la **Plataforma para Estadísticos** en *datamx*.

Código de color

1. Poblacion
2. S-D (S-D) (DF)
3. Masculino Masculino (DF)
4. Femenino Femenino (DF)
5. Desconocido
6. Desconocido
7. Cadáver conocido
8. Cadáver desconocido
9. Recién nacido
10. Feto
11. Miembros
12. Restos óseos

```
In [1]: from joblib import load
import pandas as pd
import numpy as np
from numpy import nan
from helper_pandas import *

import matplotlib.pyplot as plt
import seaborn as sns
from wordcloud import WordCloud

from scipy.stats import chi2_contingency
import statsmodels.api as sm

import warnings
warnings.simplefilter(action='ignore', category=FutureWarning)
pd.options.mode.chained_assignment = None

in_file = "data/clean/HBO_clean.pkl"
with open(in_file, 'rb') as f:
    df = load(f)

#Quitar filas vacías
df = df[df['Identificación'] != 'S-D']
df = df[df['Identificación'] != 'S-D']
display(df.shape)
df.tail()
```

(9825, 31)

```
Out[1]:
```

	Numero progresivo transcritor	Nombre completo transcritor	Primer apellido	Segundo apellido	Nombres propios	Fecha transcritor	Fecha estandar
	ID						
BO_1962_07489	S-D	placenta	NaN	NaN	NaN	1962-06-05	1962-06-05
BO_1962_07490	S-D	5 dedos del pie derecho de desconocido	NaN	NaN	NaN	1962-06-05	1962-06-05
BO_1962_07491	S-D	dedo de desconocido	NaN	NaN	NaN	1962-11-19	1962-11-19
BO_1962_07492	S-D	4 dedos de desconocido	NaN	NaN	NaN	1962-11-28	1962-11-28
BO_1962_07493	S-D	osamenta de desconocido	NaN	NaN	NaN	1962-10-11	1962-10-11

5 rows × 31 columns

```
In [2]: from matplotlib.colors import LinearSegmentedColormap

dic_color = {'Poblacion': '#984aa4', 'S-D': '#999999', 'S-D (DF)': '#d9d9d9', 'Masculino': '#377eb8', 'Masculino (DF)': '#80b1d3',
             'Femenino': '#984aa4', 'Femenino (DF)': '#e6bada', 'Desconocido': '#a64d49', 'Desconocido (DF)': '#d9d9d9',
             'Cadáver conocido': '#984aa4', 'Cadáver desconocido': '#fdd552',
             'Recién nacido': '#c875c4',
             'Feto': '#f7f7f7',
             'Miembros': '#7f7b1f',
             'Restos óseos': '#c85a33', }

gray_to_green = LinearSegmentedColormap.from_list('gray_to_green', ['#d9d9d9', '#2a7e19'])
```

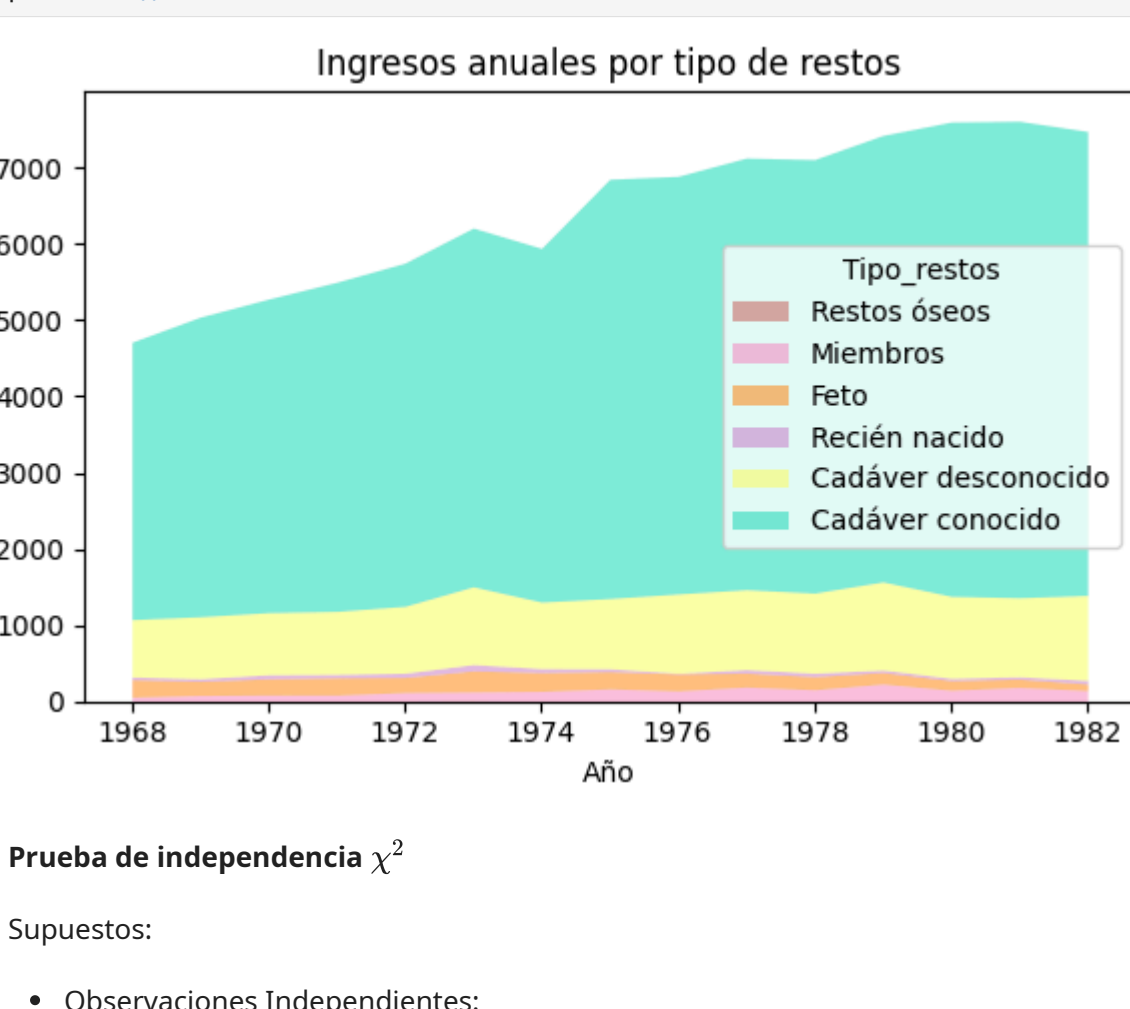
Fecha

'Fecha_transcritor', 'Fecha_estandar', 'Fecha_año', 'Fecha_mes', 'Fecha_dia_semana',

Ingresos anuales y diarios

```
In [3]: df_ano_tipo = df.pivot_table(index='Fecha_año', columns='Tipo_restos', aggfunc='size')
df_ano_tipo = df_ano_tipo[df['Tipo_restos'].cat.categories.iloc[:, :-1]]
df_ano_tipo = df_ano_tipo

fig, ax = plt.subplots(figsize=(6, 4))
df_ano_tipo.plot.area(ax=ax, color=dic_color, linewidth=0, alpha=0.5)
ax.set_xlabel('Año')
ax.set_ylabel('Ingresos anuales por tipo de restos')
ax.set_xticklabels(x.get_text().replace('0', '')) for x in ax.get_xticklabels()) # Remove negative sign from tick labels
plt.tight_layout()
plt.savefig('imagenes/BO_TipoRestos-Sexo.svg')
plt.show()
```



Prueba de independencia χ^2

Supuestos:

- Observaciones Independientes:
- Frecuencias Esperadas Suficientes: quitar Restos óseos
- Muestra Aleatoria Simple

Resultado: **Muestras no independientes:** Chi-square: 721.48 **Estadísticamente significativa:** P-value: 2.36e-116

```
In [4]: data = df_ano_tipo.drop(columns=['Restos óseos', 'Recién nacido'])

# Perform Chi-square test
chi2, p_value, dof, expected = chi2_contingency(data)
print(f'Chi-square statistic: {chi2}')
print(f'P-value: {p_value}')

# visualize residuals
thr_resid = 5
resid = data - expected / np.sqrt(expected)
resid = pd.DataFrame(resid, index=data.index, columns=data.columns)
resid = resid.where(np.abs(resid) > thr_resid)
#display(resid)
```

Chi-square statistic: 598.3101962444517

Degrees of freedom: 42

P-value: 1.7391436182219905e-99

Regresión Lineal

Variable independiente: año (X)

Variable dependiente: decesos/ingresos (Y)

$$Y = \beta_0 + \beta_1 X + \epsilon$$

Supuestos

- Linealidad
- Independencia
- Homocedasticidad: La varianza de los residuos es constante
- Normalidad residuos

Regresión Cadáver conocido anual **R-squared: 0.957 Pendiente β_1 193.043**

Regresión Cadáver desconocido anual **R-squared: 0.825 Pendiente β_1 25.882**

Regresión Miembros anual **R-squared: 0.687 Pendiente β_1 8.968**

Regresión Feto anual **R-squared: 0.556 Pendiente β_1 -8.918**

Regresión Restos óseos anual **R-squared: 0.210 NO Pendiente β_1 -0.439**

Regresión Recién nacido anual **R-squared: 0.058 NO Pendiente β_1 -0.893**

```
In [5]: X = sm.add_constant([i for i in data.index])

for col in data.columns:
    print(f'Regresion (col) anual')
    y1 = data[col]
    model1 = sm.OLS(y1, X).fit()
    #print(f'R-squared: {model1.rsquared}, f(1)in**Pendiente $1beta_1$** (model1.params.x1:3f)(n")
    print(model1.summary())

Regresión Miembros anual
```

	coef	std err	t	P> t	[0.025	0.975]
const	-1.757e+04	3312.433	-5.306	0.000	-2.47e+04	-1.04e+04
x1	8.9679	1.677	5.347	0.000	5.545	12.591
Omnibus:	2.562	Durbin-Watson:				
Prob(Omnibus):	0.278	Jarque-Bera (JB):				
Skew:	0.516	Prob(JB):				
Kurtosis:	3.542	Cond. No.				

Notes:

[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

[2] The condition number is large, 9.03e+05. This might indicate that there are strong multicollinearity or other numerical problems.

Regresión Feto anual

	coef	std err	t	P> t	[0.025	0.975]
const	1.787e+04	4366.890	4.075	0.001	8363.817	2.72e+04
x1	-8.9179	2.212	-4.031	0.001	-13.697	-4.139
Omnibus:	0.692	Durbin-Watson:				
Prob(Omnibus):	0.697	Jarque-Bera (JB):				
Skew:	0.379	Prob(JB):				
Kurtosis:	2.268	Cond. No.				

Notes:

[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

[2] The condition number is large, 9.03e+05. This might indicate that there are strong multicollinearity or other numerical problems.

Regresión Cadáver desconocido anual

	coef	std err	t	P> t	[0.025	0.975]
const	-5.058e+04	6532.957	-7.602	0.000	-6.43e+04	-3.26e+04
x1	25.8821	3.308	7.823	0.000	18.735	33.029
Omnibus:	1.458	Durbin-Watson:				
Prob(Omnibus):	0.398	Jarque-Bera (JB):				
Skew:	0.672	Prob(JB):				
Kurtosis:	2.723	Cond. No.				

Notes:

[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

[2] The condition number is large, 9.03e+05. This might indicate that there are strong multicollinearity or other numerical problems.

Regresión Cadáver conocido anual

	coef	std err	t	P> t	[0.025	0.975]
const	-3.762e+04	2.24e+04	-1.612	0.000	-6.43e+04	-3.26e+04
x1	193.0429	11.329	17.040	0.000	168.569	217.517
Omnibus:	1.458	Durbin-Watson:				
Prob(Omnibus):	0.398	Jarque-Bera (JB):				
Skew:	0.672	Prob(JB):				
Kurtosis:	2.723	Cond. No.				

Notes:

[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

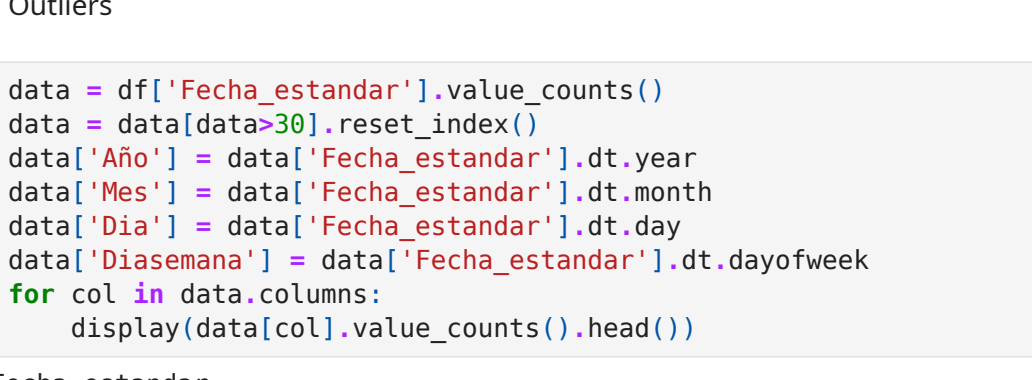
[2] The condition number is large, 9.03e+05. This might indicate that there are strong multicollinearity or other numerical problems.

Ingresos diarios de toda la población

En total, del 1968-01-01 al 1982-12-31 hay 5479 días

- Todos los días hubo ingresos
- Mediana de 17 ingresos diarios
- Min 2 ingresos, max 46 ingresos en un día

```
In [6]: fig, ax = plt.subplots(figsize=(6, 4))
df_ano_tipo = df_ano_tipo[df['Tipo_restos'].cat.categories.iloc[:, :-1]]
df_ano_tipo = df_ano_tipo
df_ano_tipo.plot.hist(ax=ax, color=dic_color, bins=range(46), alpha=0.5,
                    title='Distribución de ingresos diarios')
plt.show()
```



Outliers

```
In [7]: data = df['Fecha_estandar'].value_counts()
data = data[data>0].reset_index()
data['Año'] = data['Fecha_estandar'].dt.year
data['Mes'] = data['Fecha_estandar'].dt.month
data['Dia'] = data['Fecha_estandar'].dt.day
data['Diasemana'] = data['Fecha_estandar'].dt.dayofweek
for col in data.columns:
    display(data[col].value_counts().head())
```

Fecha_estandar

1968-03-27 1

1979-12-26 1

1962-06-21 1

1981-12-26 1

1977-04-19 1

Name: count, dtype: int64

count

32 35

31 32

34 17

33 12

35 10

Name: count, dtype: int64

Año

1968 27

1962 23

1980 17

1976 16

1975 12

Name: count, dtype: int64

Mes

1 27

3 16

1 16

11 16

8 15

Name: count, dtype: int64

Dia

26 12

19 8

8 8

3 7

Name: count, dtype: int64

Diasemana

0 55

6 21

1 21

3 12

5 11

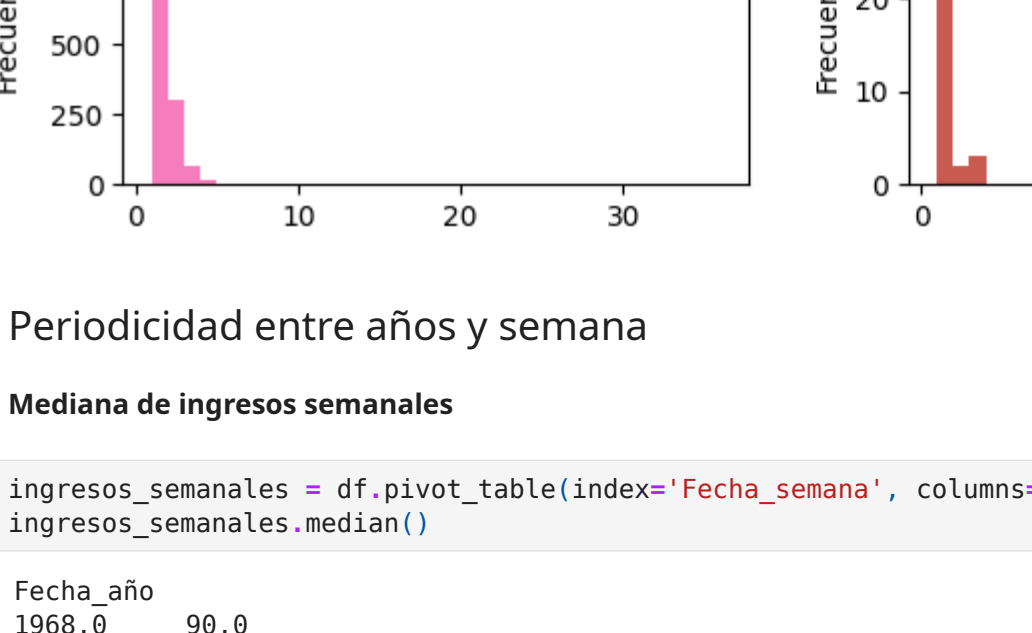
Name: count, dtype: int64

Por tipo de restos

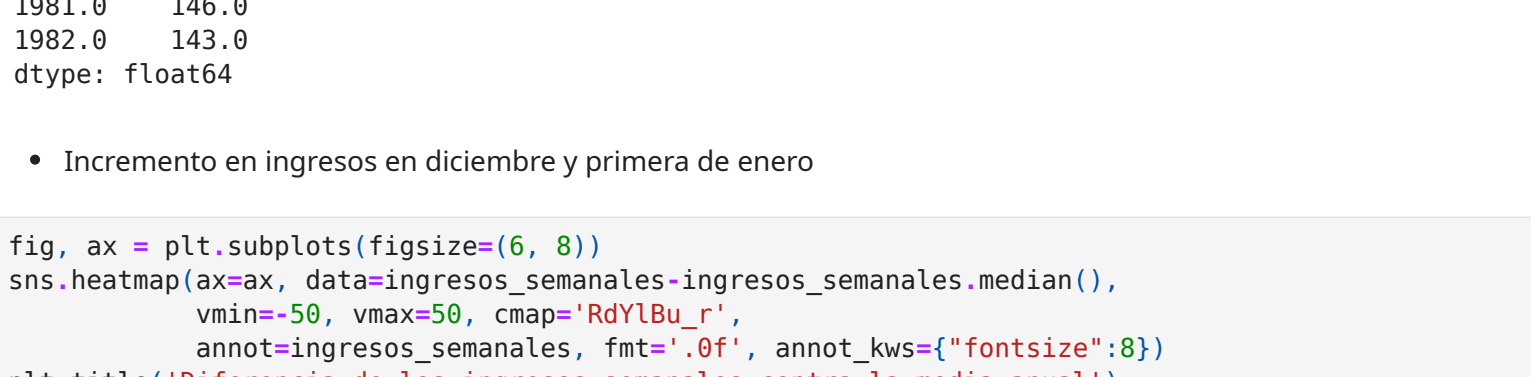
```
In [8]: ing_diarios = df.pivot_table(index='Fecha_estandar', columns='Tipo_restos', aggfunc='size').sort_index().replace(0,nan)
display(pd.concat([ing_diarios.fillna(0).describe(), df['Fecha_estandar'].value_counts().describe()], axis=1))

fig, ax = plt.subplots(figsize=(6, 4))
ing_diarios.plot.hist(ax=ax, color=dic_color, bins=range(37), alpha=0.5,
                    title='Distribución de ingresos diarios')
plt.savefig('imagenes/BO_Distribucion_TipoRestos-IngresosDiarios.svg')
plt.show()
```

	Cadáver conocido	Cadáver desconocido	Recién nacido	Feto	Miembros	Restos óseos	count
count	5479.000000	5479.000000	5479.000000	5479.000000	5479.000000	5479.000000	5479.000000
mean	13.975908	2.63497	0.115350	0.518160	0.374156	0.000878	17.627122
std	5.212267	1.94737	0.373721	0.780834	0.705295	0.111903	6.060894
min	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000	2.000000
25%	10.000000	1.000000	0.000000	0.000000	0.000000	0.000000	13.000000
50%	14.000000	2.000000	0.000000	0.000000	0.000000	0.000000	17.000000
75%	17.000000	4.000000	0.000000	1.000000	1.000000	0.000000	21.000000
max	36.000000	18.000000	5.000000	6.000000	9.000000	3.000000	46.000000



```
In [9]: fig, axes = plt.subplots(nrows=3, ncols=2, figsize=(8, 6), sharex=True)
axes = axes.flatten()
for ax, col in zip(axes, ing_diarios.columns):
    ing_diarios[col].plot.hist(ax=ax, color=dic_color, bins=range(1,37), ylabel='Frecuencia')
    ax.set_ylabel('Frecuencia')
    ax.set_xlabel('Frecuencia')
    plt.tight_layout()
plt.show()
```



Periodicidad entre años y semana

Mediana de ingresos semanales

```
In [10]: ingresos_semanales = df.pivot_table(index='Fecha_semana', columns='Fecha_año', aggfunc='size').sort_index()
ingresos_semanales.median()
```

Out[10]:

Fecha_año

1968.0 96.0

1969.0 94.0

1970.0 102.0

1971.0 102.0

1972.0 118.0

1973.0 117.0

1974.0 114.0

1975.0 132.0

1976.0 138.0

1977.0 134.0

1978.0 136.0

1979.0 142.0

1980.0 144.0

1981.0 146.0

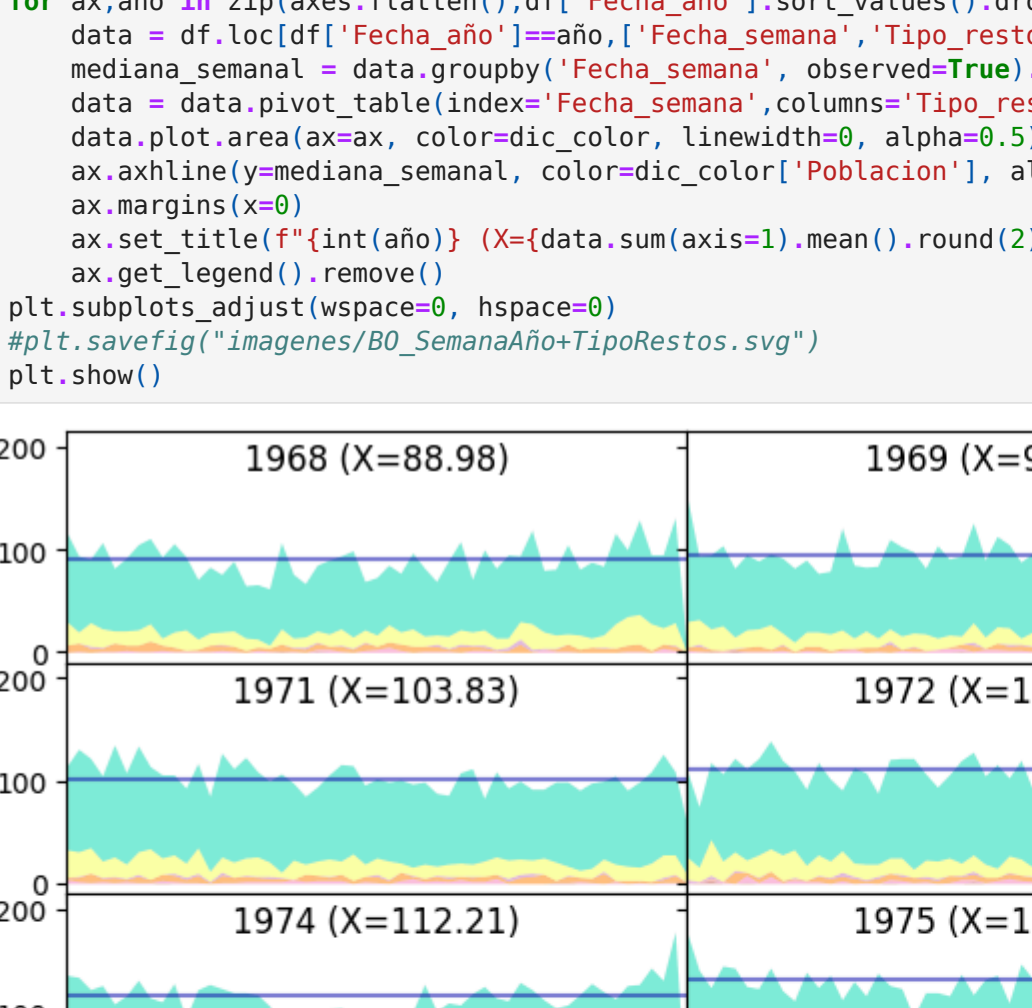
1982.0 143.0

dtype: float64

- Incremento en ingresos en diciembre y primera de enero

```
In [11]: fig, ax = plt.subplots(figsize=(6, 8))
sns.heatmap(ax=ax, data=ingresos_semanales-ingresos_semanales.median(),
            vmin=-50, vmax=50, cmap=rdylbbr,
            annot=ingresos_semanales, fmt='%', annot_kws={'fontsize':8})
plt.title('Diferencia de los ingresos semanales contra la media anual')
plt.tight_layout()
plt.savefig('imagenes/BO_SemanaIngresos.svg')
plt.show()
```

Diferencia de los ingresos semanales contra la media anual



```
In [12]: ingresos_semanales.median()
```

Out[12]:

Fecha_año

1968.0 96.0

1969.0 94.0

1970.0 102.0

1971.0 102.0

1972.0 118.0

1973.0 117.0

1974.0 114.0

1975.0 132.0

1976.0 138.0

1977.0 134.0

1978.0 136.0

1979.0 142.0

1980.0 144.0

1981.0 146.0

1982.0 143.0

dtype: float64

```
In [13]: ncols = 3
fig, axes = plt.subplots(nrows=round(df['Fecha_año'].nunique()/ncols), ncols=ncols,
                        figsize=(12, df['Fecha_año'].nunique()/ncols),
                        sharex=True, sharey=True,)
for ax, año in zip(axes, df['Fecha_año'].sort_values().dropna().unique()):
    data = df.loc[df['Fecha_año']==año, ['Fecha_semana', 'Tipo_restos']]
    mediana_semanal = data.groupby('Fecha_semana', observed=True).size().median()
    ax.set_xlabel('Fecha_semana')
    ax.set_ylabel('Fecha_año')
    ax.set_title(f'Fecha_año={año} (X={data.sum(axis=1).mean().round(2)})')
    ax.get_legend().remove()
plt.subplots_adjust(wspace=0, hspace=0)
plt.savefig('imagenes/BO_SemanaIngresos.svg')
plt.show()
```

