



Ciencia y analítica de datos

DRA. GRETTEL BARCELÓ ALONSO

1

MÓDULO 3

ANÁLISIS, VISUALIZACIÓN Y TRANSFORMACIÓN DE
DATOS

2

1

Análisis exploratorio de datos

EDA (EXPLORATORY DATA ANALYSIS)

Permite evaluar de manera inicial la **calidad de los datos**

- Errores de captura o entrada
- Inconsistencias en formatos
- Falta de estandarización
- Duplicados

Pasos de EDA



3

Tipos de datos vs ESCALA DE MEDICIÓN

Tipos en pandas

- int64
- float64
- object

Escalas de medición

- Qualitativas o categóricas: Representan categorías o atributos.
 - Nominales: No hay orden entre clases
 - Ordinales: Cada categoría es más alta o mejor que la anterior
- Quantitativas: Números que suelen representar un conteo o una medición.
 - Discretas: Suelen tomar solamente valores enteros
 - Continuas: Entre dos valores cualesquiera, puede haber valores intermedios (valores reales)

4

2

Tipo estadístico

ESCALA DE MEDICIÓN

- En algunos casos, una variable puede estar representada numéricamente, pero en realidad ser de naturaleza categórica.

- Antes de realizar el EDA, es recomendable revisar cuidadosamente el **tipo de datos** (`dtypes`), la cantidad de **valores únicos** (`nunique()`) y considerar el contexto de cada variable para convertirlas al tipo más adecuado

	Nominal	Ordinal
Ejemplos	Códigos postales: 12345, 67890, 54321 Identificadores: 1001, 1002, 1003	Niveles educativos: 1, 2, 3 Niveles de satisfacción: 1, 2, 3, 4, 5
Conversión recomendada	<code>object</code>	<code>category</code>
Implementación	<code>df['zipcode'] = df['zipcode'].astype('object')</code>	<code>df['education_level'] = pd.Categorical(df['education_level'], categories=[1, 2, 3], ordered=True)</code>

* Para las variables binarias, puedes usar directamente:
`df['gender'] = df['gender'].astype('category')`

5

Análisis univariado

DESCRIPTIVAS

Medidas

Tendencia central

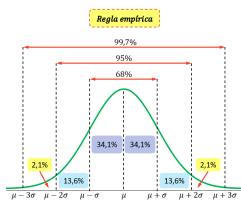
- `Media mean()`
- `Mediana median()`
- `Moda mode()`

Variabilidad

- `Rango max() - min()`
- `Varianza var()`
- `Desviación estándar std()`

Forma

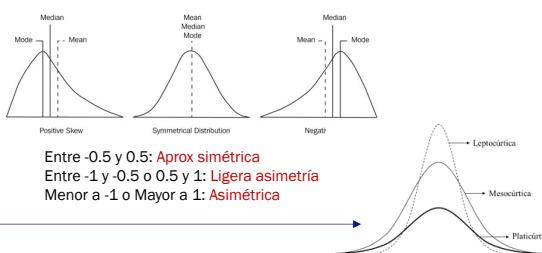
- `Sesgo skew()`
- `Curtosis kurt()`



Valores atípicos

Observaciones significativamente diferentes del resto.

- Límite con conocimiento de dominio
- Media y desviación estándar
- Método IQR



Entre -0.5 y 0.5: Aprox simétrica
 Entre -1 y -0.5 o 0.5 y 1: Ligera asimetría
 Menor a -1 o Mayor a 1: Asimétrica

Entre -0.5 y 0.5: Aprox mesocúrtica
 Menor a -0.5: Platicúrtica (achatada)
 Mayor a 0.5: Leptocúrtica (picuda)

6

3

Análisis univariado

DESCRIPTIVAS

- Las tablas de frecuencias y proporciones complementan las descriptivas de las variables categóricas.
- Estas tablas permiten conocer cuántas veces aparece cada categoría (frecuencia absoluta) y qué porcentaje representan respecto al total (frecuencia relativa o proporción).

Frecuencias absolutas

```
credit['loan_intent'].value_counts()
```

loan_intent	count
EDUCATION	6246
MEDICAL	5871
VENTURE	5520
PERSONAL	5347
DEBTCONSOLIDATION	5045
HOMEIMPROVEMENT	3501

Frecuencias relativas

```
credit['loan_intent'].value_counts(normalize=True)
```

loan_intent	proportion
EDUCATION	0.20
MEDICAL	0.19
VENTURE	0.18
PERSONAL	0.17
DEBTCONSOLIDATION	0.16
HOMEIMPROVEMENT	0.11

7

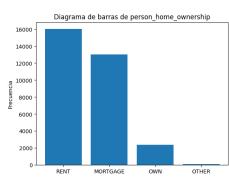
Análisis univariado

VISUALIZACIÓN

La **visualización** de datos permite ver la información de forma comprensible y cohesiva, facilitando la identificación de patrones y la comunicación asertiva de las conclusiones o descubrimientos.

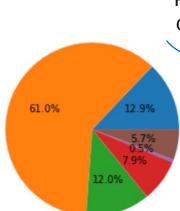
Categóricos

Nominales



Ordinales

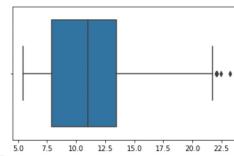
Gráfico de barras
Gráfico circular



Cuantitativos

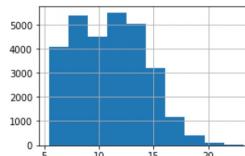
Discretos

Pocos datos



Continuos

Muchos datos



8

4

Visualización EN PYTHON

- El panorama de visualización de Python puede parecer abrumador al principio.
- Se ha creado [PyViz.org](https://pyviz.org), un sitio para ayudar a los usuarios a decidir cuáles son las mejores herramientas de visualización de código abierto de Python para sus propósitos.

<https://pyviz.org/overviews/index.html>



9

Plataformas de visualización EN PYTHON

Matplotlib es una de las denominadas bibliotecas núcleo (core), sobre la que se construyen varias plataformas de nivel superior.

Tiene una API completa y potente, que permite personalizar cualquier atributo de la figura

Las plataformas de alto nivel que ocupan Matplotlib, proporcionan una API más simple para cubrir las tareas más comunes de manera concisa y conveniente. Dos de las más usadas son:

- API `.plot()` de Pandas
- Seaborn

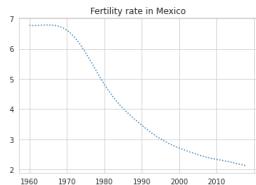


10

5

Gráfico de líneas

ANÁLISIS UNIVARIADO



Un gráfico de **líneas** se utiliza para visualizar información que cambia continuamente con el tiempo.

Para utilizar estas plataformas debes importarlas en los scripts de Python

- import `matplotlib.pyplot as plt`
- import `pandas as pd`
- import `seaborn as sns`

Y a través de ellas llamar a los métodos o atributos

Por ejemplo, la función de graficado básica es `plot(x,y)`, que grafica valores de y contra x como líneas y/o marcadores

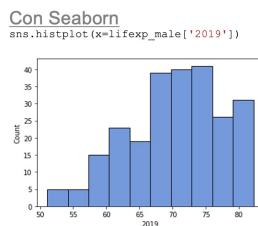
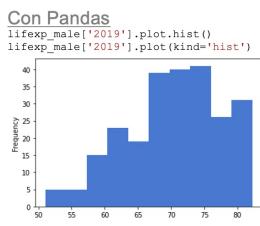
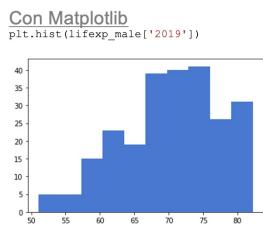
- Con Matplotlib: `plt.plot(df.index, df[columna])`
- Con Pandas: `df.plot()`
- Con Seaborn: `sns.lineplot(x=df.index, y=df[columna])`

11

Histogramas

ANÁLISIS UNIVARIADO

- Un **histograma** es una representación en barras de la distribución de los datos.
- En el eje horizontal se indican los valores de la variable y en el vertical sus frecuencias.
- Las frecuencias se agrupan en clases o **bins**.
- Utiliza este tipo de diagrama cuando deseas observar el grado de homogeneidad o **variabilidad** de las columnas **cuantitativas continuas** del dataframe.



12

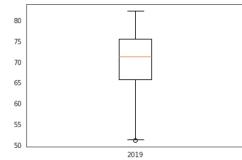
Diagramas de caja y bigote

ANÁLISIS UNIVARIADO

- Los diagramas de **caja y bigote** (*boxplot*) se utilizan para mostrar la distribución de datos cuantitativos continuos.
- Ofrecen un panorama de la distribución de dichos valores, a través de sus **cuartiles**. Para ello, utilizan como representación una caja y segmentos (bigotes) que delimitan los intervalos donde la variable continua concentra la mayoría de las observaciones.

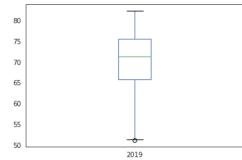
Con Matplotlib

```
plt.boxplot(lifexp_male['2019'].dropna(),
            labels=['2019'])
```



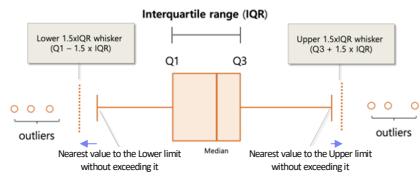
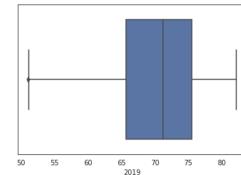
Con Pandas

```
lifexp_male['2019'].plot.box()
lifexp_male['2019'].plot(kind='box')
```



Con Seaborn

```
sns.boxplot(x=lifexp_male['2019'])
```

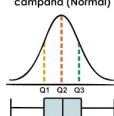


14

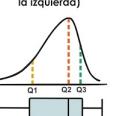
Elementos adicionales

ANÁLISIS UNIVARIADO

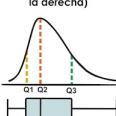
Distribución simétrica o en forma de campana (Normal)



Distribución osimétrica negativa (cola hacia la izquierda)



Distribución osimétrica positiva (cola hacia la derecha)



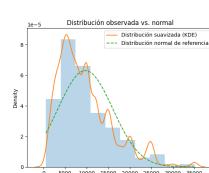
Elementos adicionales en el histograma

- Gráficos KDE

```
sns.kdeplot(x=df['col'])
```

- Referencia de la curva normal

```
from scipy.stats import norm
x_vals = np.linspace(df['col'].min(), df['col'].max(), 1000)
normal_ref = norm.pdf(x_vals, loc=df['col'].mean(), scale=df['col'].std())
plt.plot(x_vals, normal_ref)
```



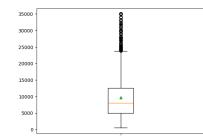
Elementos adicionales en el boxplot

- Promedio **mean**

```
bp = plt.boxplot(df['col'], showmeans=True)
```

- Listado de **outliers**

```
bp['fliers'][0].get_ydata()
```



15

Diagramas de barras

ANÁLISIS UNIVARIADO

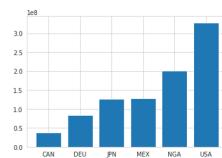
Los gráficos de **barras** se utilizan para mostrar datos categóricos o cuantitativos discretos, con barras rectangulares de longitudes proporcionales a los valores que representan.

Estos valores pueden ser:

1. El total, promedio u otra medida de resumen de cada categoría
2. El conteo o frecuencia de cada categoría

Con Matplotlib

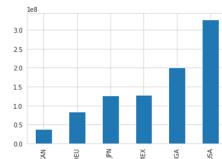
```
plt.bar(population.index, population['2019'])
```



Con Pandas

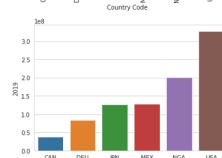
```
population['2019'].plot(kind='bar')
```

```
population ['2019'].plot.bar()
```



Con Seaborn

```
sns.barplot(x=population.index, y=population['2019'])
```



16

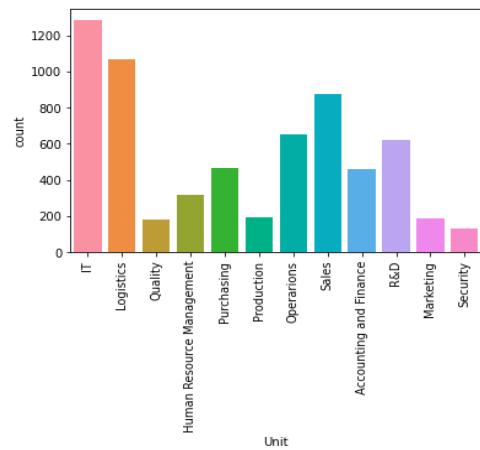
Diagramas de barras

ANÁLISIS UNIVARIADO

Seaborn ofrece además un gráfico de recuento, con variables categóricas, que permite realizar el agrupamiento para *count* de manera automática:

```
sns.countplot(columna)
```

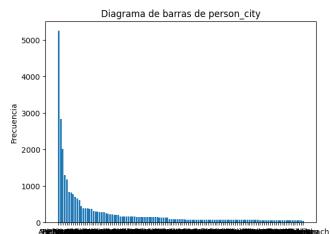
Con pandas y Matplotlib habría que usar *groupby()* o *value_counts()* previo al trazado.



17

Alta cardinalidad

ANÁLISIS UNIVARIADO



- Visualizar tantas categorías en un solo gráfico puede resultar complicado, ya que las barras se vuelven muy pequeñas y las etiquetas del eje x tienden a empalmarse.
- Es recomendable aplicar estrategias como **agrupar categorías** menos frecuentes, mostrando solo las más comunes.

```
counts = credit['person_city'].value_counts()  
top_20 = counts.head(20)  
otros = counts.iloc[20:].sum()  
counts_con_otros = pd.concat([top_20, pd.Series({'Others': otros})])  
plt.bar(counts_con_otros.index, counts_con_otros.values)
```

También podrías agrupar categorías según **criterios o dominios** conocidos. Por ejemplo, en ciudades, agruparlas por estado o región; en productos, agrupar por tipo o categoría mayor.

18

Análisis bivariado

VISUALIZACIÓN

Variable X	Variable Y	Propósito del análisis	Tipo de gráfico
Cuantitativa	Cuantitativa	¿Cómo se correlaciona Y con X?	Scatter plot
Categórica	Categórica	¿Cuál es el número o porcentaje de registros de Y que se incluyen en cada categoría de X?	Stacked bar
Categórica	Cuantitativa	¿Cómo varía el rango de Y para los distintos niveles de categoría de X?	Box plot

19

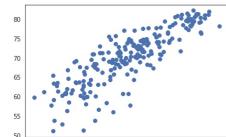
Diagrama de dispersión

ANÁLISIS BIVARIADO

- Los **diagramas de dispersión** (*scatter plot*) muestran la relación entre dos variables.
- Utilizan como representación un conjunto de puntos ubicados en coordenadas cartesianas, según los valores de las dos variables.
- De estos puntos se puede notar si las dos aumentan a la vez (**correlación positiva**), si una aumenta mientras la otra disminuye (**correlación negativa**) o si no tienen relación alguna (**correlación nula**).

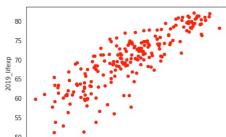
Con Matplotlib

```
plt.scatter(x=population['gni'],y=population['lifexp'])
```



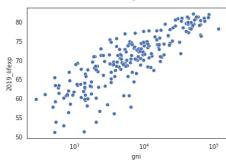
Con Pandas

```
population.plot(kind='scatter',x='gdp',y='lifeexp',c='red')
```



Con Seaborn

```
sns.scatterplot(x=population['gdp'],y=population['lifeexp'])
```

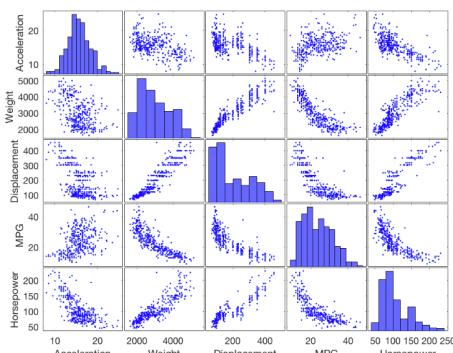


20

Diagrama de dispersión

ANÁLISIS BIVARIADO

- Las relaciones observadas en los diagramas de dispersión pueden cuantificarse con medidas de asociación; particularmente a la asociación entre variables numéricas se le denomina **correlación**.
- Se puede calcular directamente con la función `corr()` de pandas
- Si bien no existe una escala universal para la **fuerza** de correlación, las siguientes son pautas generales para interpretar los coeficientes:
 - | c | < 0.4: Correlación débil o nula
 - 0.4 < | c | < 0.7: Correlación moderada
 - | c | > 0.7: Correlación fuerte



```
sns.pairplot(df)
```

21

Coeficiente de correlación

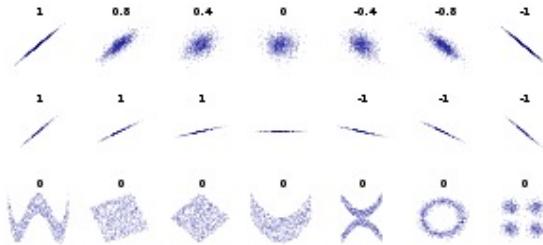
ANÁLISIS BIVARIADO

El coeficiente que se calcula por defecto es el de Pearson (medida de dependencia lineal)

$$r = \frac{\text{cov}[X, Y]}{\sqrt{\text{var}[X]\text{var}[Y]}}$$

$$r = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum(x_i - \bar{x})^2 \sum(y_i - \bar{y})^2}}$$

- Valores de la variable x
- Valores de la variable y
- Promedio de la variable x
- Promedio de la variable y



Varios grupos de puntos, con el coeficiente de correlación (r) para cada grupo.

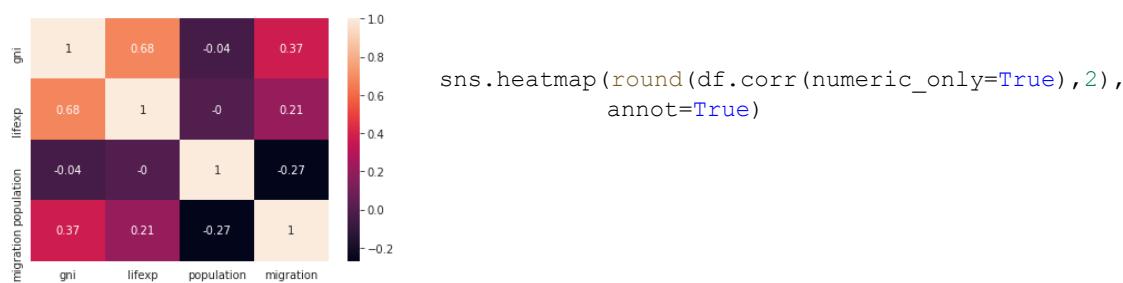
La correlación de Pearson refleja la fuerza y la dirección de una relación lineal entre dos variables. En la figura del centro, la varianza de y es nula, por lo que la correlación es indeterminada. Lo mismo sucedería con una línea vertical.

22

Mapa de calor

ANÁLISIS BIVARIADO

- Seaborn además complementa las matrices de dispersión con mapas de calor (heatmap), incluyendo en estos últimos la correlación numérica entre el par de variables.
- Puesto que en la diagonal se encuentra relacionada una variable con ella misma, el coeficiente de correlación es 1.



Tablas de contingencia

ANÁLISIS BIVARIADO

- La herramienta básica para este análisis suele ser la **tabla de contingencia** (tablas cruzadas), que resume la frecuencia conjunta de las distintas combinaciones de categorías.

```
pd.crosstab(df['col1'], df['col2'])
```

	CODE_GENDER	F	M	XNA
NAME_CONTRACT_TYPE				
Cash loans		140240	85979	0
Revolving loans		16935	8974	4

- También es posible obtener las **frecuencias relativas** por fila utilizando el parámetro `normalize`

- `all`: Devuelve proporciones respecto al total general (la tabla suma 1)
- `index`: Normaliza por fila
- `columns`: Normaliza por columna

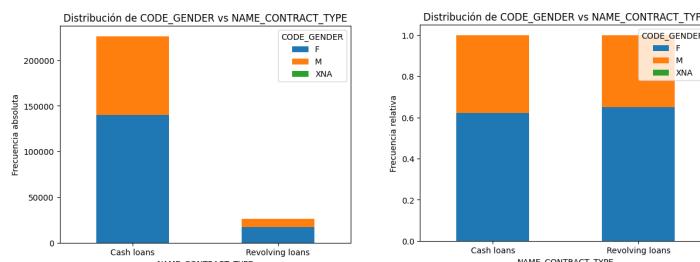
	CODE_GENDER	F	M	XNA
NAME_CONTRACT_TYPE				
Cash loans		0.62	0.38	0.0
Revolving loans		0.65	0.35	0.0

28

Gráfico de barras apiladas

ANÁLISIS BIVARIADO

- Los resultados obtenidos de ambas tablas se pueden mostrar en gráficos de **barras apiladas**.
- La representación gráfica permite visualizar de forma mucho más clara y directa los **patrones de asociación** observados en las tablas de contingencia.



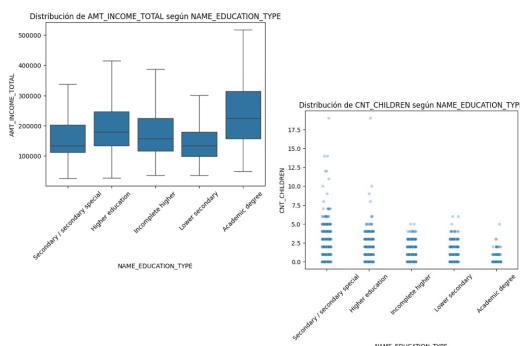
```
contingency_tab.plot(kind='bar', stacked=True)
```

29

Boxplot agrupado por categoría

ANÁLISIS BIVARIADO

- En este tipo de combinaciones **mixtas**, se suele comparar la distribución de la variable numérica a través de los distintos niveles de la categórica, para identificar posibles diferencias entre los grupos.



- Desde el punto de vista de visualización, se puede recurrir a representaciones gráficas como **boxplots por categoría**.

```
sns.boxplot(x='col1', y='col2', data=df)
```

- Si la variable numérica es discreta con pocos valores posibles, esto limitaría la capacidad del boxplot para reflejar variaciones en la distribución. En ese caso si recomienda usar gráficos de dispersión categórica que muestran puntos individuales con posible dispersión (*jitter*) para evitar superposición.

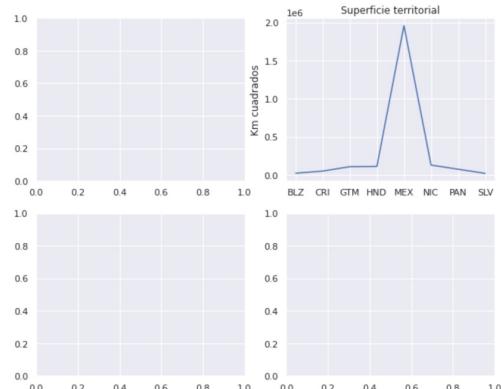
```
sns.stripplot(x='col1', y='col2', data=df)
```

31

Subgráficas EDA

- Las **subgráficas** se ocupan para conjuntar visualizaciones de la misma o diversa naturaleza y sintetizar la información en una única figura.
- Pueden ser colocadas en una dirección o en dos direcciones.
- Para crear la matriz explícitamente, se usa la función **subplots()** de Matplotlib.
- Dentro de la matriz puede generarse cada gráfica de manera independiente usando **cualquiera de las plataformas** estudiadas.

```
fig, axs = plt.subplots(2, 2, figsize=(10, 8))
axs[0, 1].plot(centralAmerica['SurfaceArea'])
axs[0, 1].set_title('Superficie territorial')
axs[0, 1].set_ylabel('Km cuadrados')
```



33

13

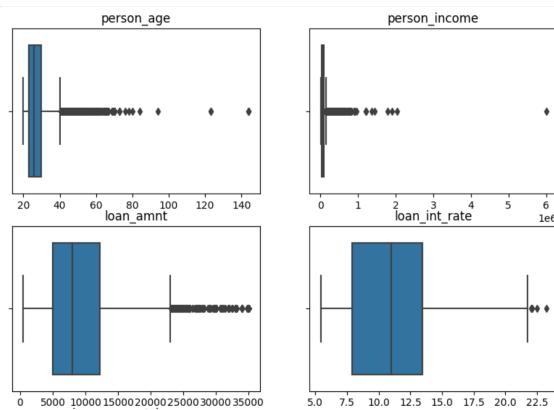
Subgráficas

EDA

Si todas las subgráficas serán de un **mismo tipo**, se puede usar un ciclo:

```
fig, axes = plt.subplots(2, 2)
axes = axes.ravel()

for col, ax in zip(df.columns, axes):
    sns.boxplot(x=df[col], ax=ax)
    ax.set(title=f'{col}', xlabel=None)
```



34



Derechos Reservados 2025 | Tecnológico de Monterrey |
Prohibida la reproducción total o parcial de esta obra sin
expresa autorización del Tecnológico de Monterrey.

35

14