KIBERNUM
IT Academy

Módulo 7
Clase 1

Análisis Exploratorio de Datos

Librería Seaborn

Librería Seaborn

Seaborn es una librería de Python creada para facilitar el análisis visual estadístico. Su filosofía es hacer fácil las líneas de código para destinar menor tiempo a la codificación y mayor tiempo al análisis. A continuación, se resumen sus principales características:

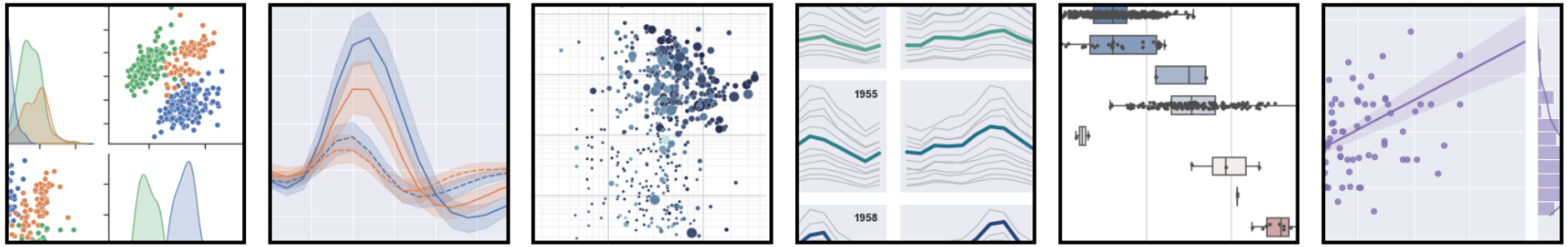
- **Es una librería para análisis estadístico visual**
- **Seaborn utiliza como base la librería Matplotlib, otorgando una capa de alto nivel con una interfaz de programación limpia y sencilla para el usuario**
- **Seaborn provee una amplia variedad de gráficos para la representación de los datos, con muy atractivos estilos**
- **Seaborn trabaja muy bien con Dataframes Pandas**

Web Seaborn

https://seaborn.pydata.org/

seaborn Installing Gallery Tutorial API Releases Citing FAQ

seaborn: statistical data visualization



Seaborn is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics.

For a brief introduction to the ideas behind the library, you can read the introductory notes or the paper. Visit the installation page to see how you can download the package and get started with it. You can browse the example gallery to see some of the things that you can do with seaborn, and then check out the tutorials or API reference to find out how.

To see the code or report a bug, please visit the GitHub repository. General support questions are most at home on stackoverflow, which has a dedicated channel for seaborn.

Contents

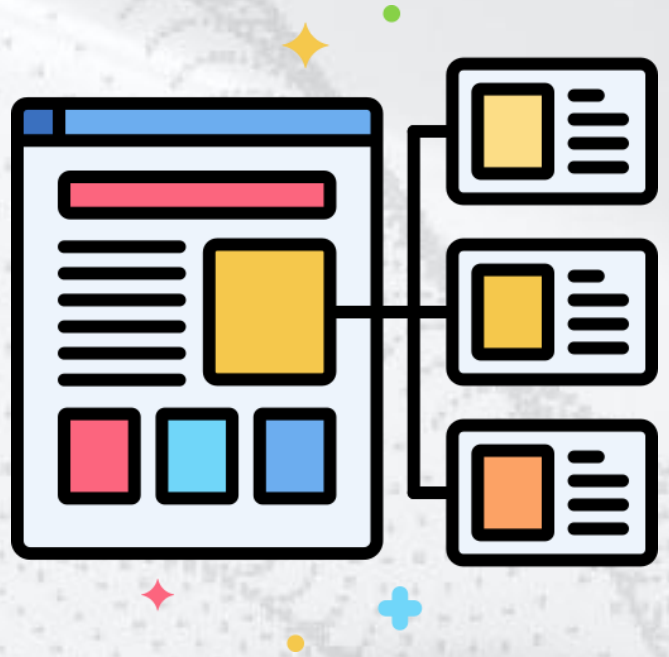
Features

- New Objects: API | Tutorial
- Relational plots: API | Tutorial
- Distribution plots: API | Tutorial
- Categorical plots: API | Tutorial
- Regression plots: API | Tutorial
- Multi-plot grids: API | Tutorial
- Figure theming: API | Tutorial
- Color palettes: API | Tutorial

Tabla de Contenidos

En esta sesión, utilizaremos la librería Seaborn, para realizar los siguientes tipos de análisis:

- **Análisis univariado**
- **Análisis bivariado**
- **Análisis multivariado**
- **Análisis con variables categóricas**
- **Visualización de matrices**
- **Grillas de visualización**



Set de Datos

- La propina en los restaurantes puede verse influida por diversos factores, incluyendo la naturaleza del restaurant, tamaño de la celebración y la locación de la mesa, entre otros.
- En un restaurant, un garzón dejó registro de los siguientes datos en todos los clientes que atendió durante dos meses y medio, a principios de los 90’s.
- El restaurant, localizado en un mall suburbano, era parte de una cadena nacional y servía un variado menú. Acorde a la ley, ofrecía mesas para fumadores y no-fumadores.

Set de Datos

Los datos registrados fueron los siguientes:

total_bill	tip	sex	smoker	day	time	size
16.99	1.01	Female	No	Sun	Dinner	2
10.34	1.66	Male	No	Sun	Dinner	3
21.01	3.50	Male	No	Sun	Dinner	3
23.68	3.31	Male	No	Sun	Dinner	2
24.59	3.61	Female	No	Sun	Dinner	4
25.29	4.71	Male	No	Sun	Dinner	4
8.77	2.00	Male	No	Sun	Dinner	2
26.88	3.12	Male	No	Sun	Dinner	4
15.04	1.96	Male	No	Sun	Dinner	2
14.78	3.23	Male	No	Sun	Dinner	2

Importando la librería

Primero realizamos las importaciones

```
In [1]: import seaborn as sns
        %matplotlib inline
```

Variable de ambiente para desplegar en línea los gráficos

Ahora realizaremos la carga de un dataset de ejemplos que trae incorporado la librería Seaborn.

```
In [3]: propinas = sns.load_dataset('tips')
```

```
In [4]: propinas.head()
```

Out[4]:

	total_bill	tip	sex	smoker	day	time	size
0	16.99	1.01	Female	No	Sun	Dinner	2
1	10.34	1.66	Male	No	Sun	Dinner	3
2	21.01	3.50	Male	No	Sun	Dinner	3
3	23.68	3.31	Male	No	Sun	Dinner	2
4	24.59	3.61	Female	No	Sun	Dinner	4

Análisis univariado

KIBERNUM

Qué es el análisis

- El análisis univariado es una técnica estadística utilizada para analizar un solo conjunto de datos o variable a la vez. Se trata de una técnica descriptiva que se enfoca en explorar y resumir las características de una variable sin considerar su relación con otras variables.

- En el análisis univariado, se utilizan medidas estadísticas como la media, la mediana, la moda, el rango y la desviación estándar para describir y resumir la distribución de los datos de una variable en particular. También se pueden utilizar gráficos como histogramas, diagramas de caja y bigotes, y gráficos de barras para visualizar la distribución de los datos.

- El análisis univariado se utiliza comúnmente en diferentes campos, como la investigación de mercado, la psicología, la medicina y la biología, entre otros. En la investigación de mercado, por ejemplo, el análisis univariado puede utilizarse para identificar las características demográficas de los consumidores que compran un producto en particular. En la psicología, se puede utilizar para analizar las respuestas de los participantes en una encuesta sobre su estado de ánimo.

KIBERNUM

Análisis de Distribución

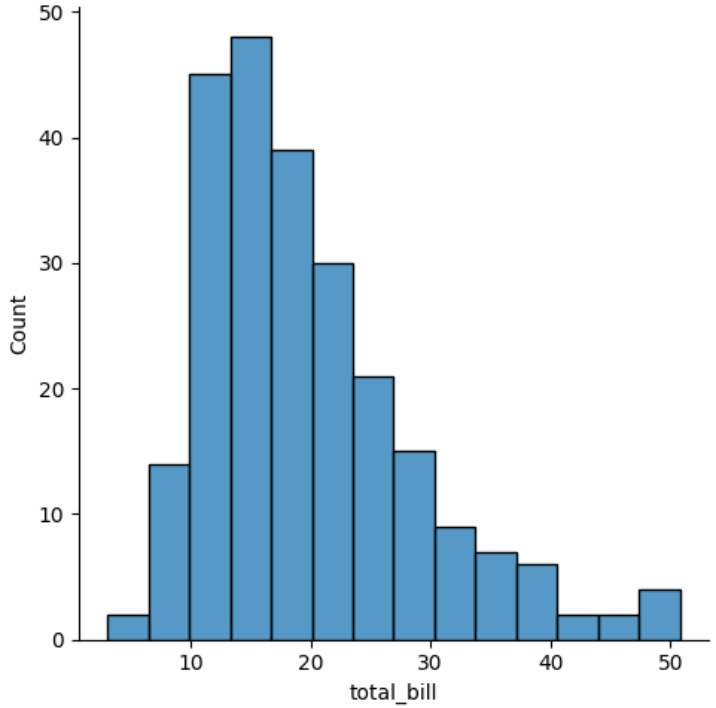
Permite graficar diagramas de distribución de observaciones (similar a la función hist() de matplotlib)

Más información:
https://seaborn.pydata.org/generated/seaborn.displot.html

Objeto dataframe

```
sns.displot(data=df, x='total_bill')
```

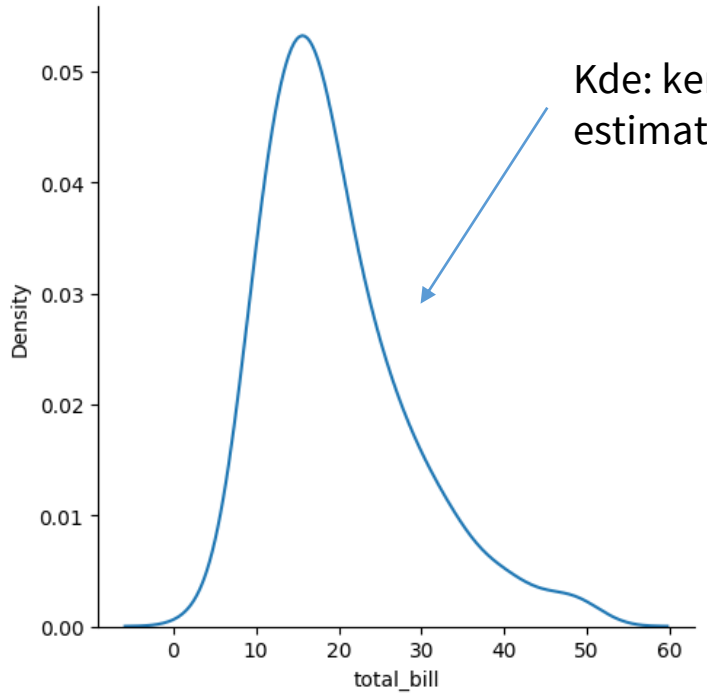
na de datos



Análisis de Distribución

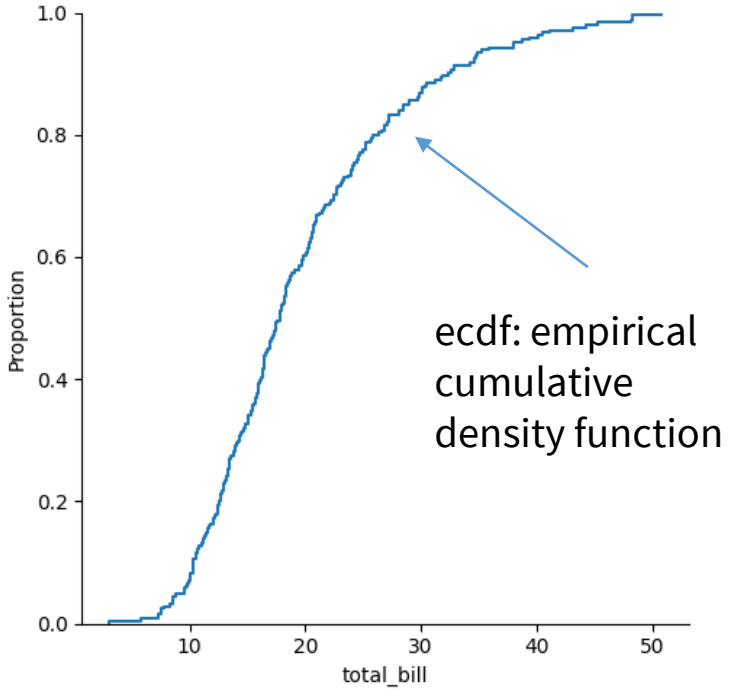
El parámetro kind permite el despliegue de otros tipos de diagrama de distribución.

```
sns.displot(data=df, x='total_bill', kind='kde')
```



Kde: kernel density estimate

```
sns.displot(data=df, x='total_bill', kind='ecdf')
```



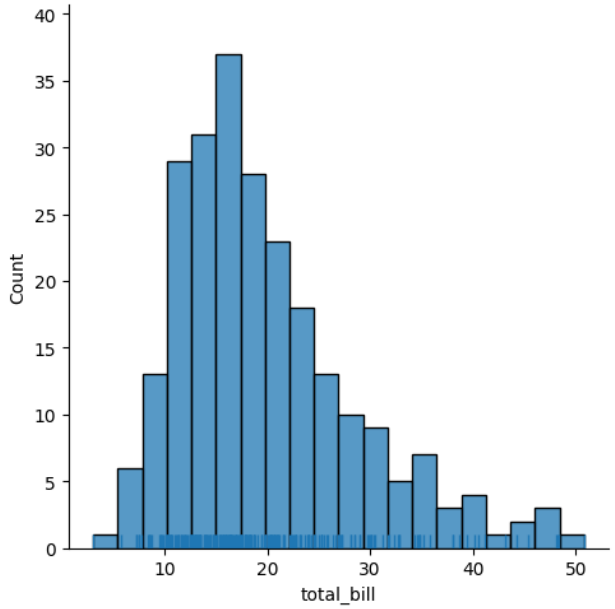
ecdf: empirical cumulative density function

Más información:
https://seaborn.pydata.org/generated/seaborn.displot.html

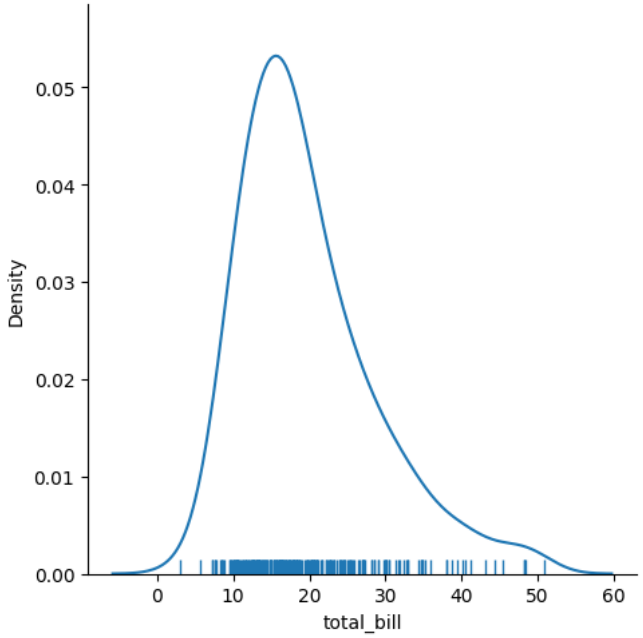
Análisis de Distribución

Mejorando el diagrama.

```
sns.displot(data=df, x='total_bill', bins=20, rug=True)
```




```
sns.displot(data=df, x='total_bill', rug=True, kind='kde')
```



Más información:
https://seaborn.pydata.org/generated/seaborn.displot.html

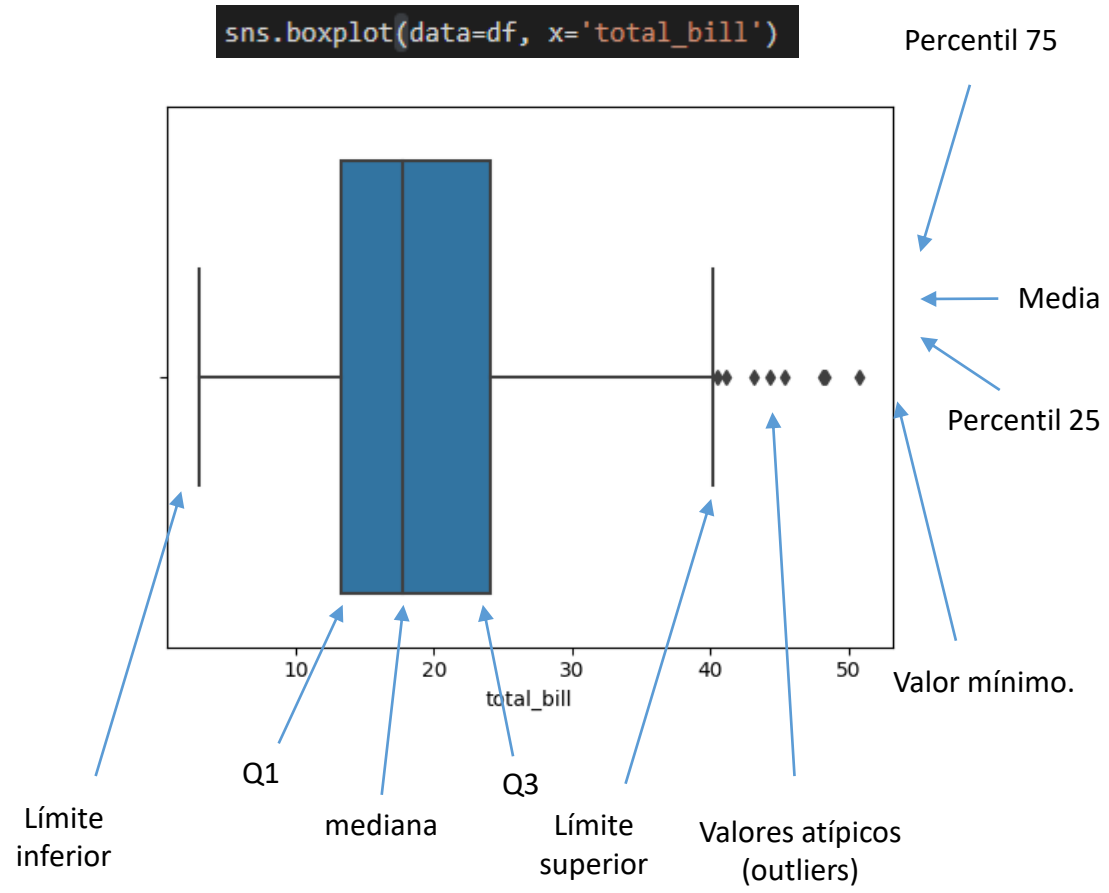
Diagrama de Caja

Más información:
https://seaborn.pydata.org/generated/seaborn.boxplot.html

- Un diagrama de caja, también conocido como boxplot en inglés, es una herramienta gráfica utilizada para representar la distribución de un conjunto de datos numéricos. El diagrama de caja muestra el rango intercuartílico (IQR), que es la distancia entre el primer y tercer cuartil del conjunto de datos, y los valores mínimo y máximo. También muestra los valores atípicos o fuera de rango, que se representan como puntos individuales fuera de la caja.

- La caja en sí misma representa el rango intercuartílico, y la línea dentro de la caja representa la mediana. Los cuartiles dividen el conjunto de datos en cuatro partes iguales, y el primer cuartil (Q1) representa el valor que es más pequeño que el 25% de los datos, mientras que el tercer cuartil (Q3) representa el valor que es más pequeño que el 75% de los datos.

- El diagrama de caja es una herramienta útil para detectar valores atípicos y para comparar distribuciones entre diferentes conjuntos de datos.



un valor atípico (en inglés **outlier**) es una observación que es numéricamente distante del resto de los datos

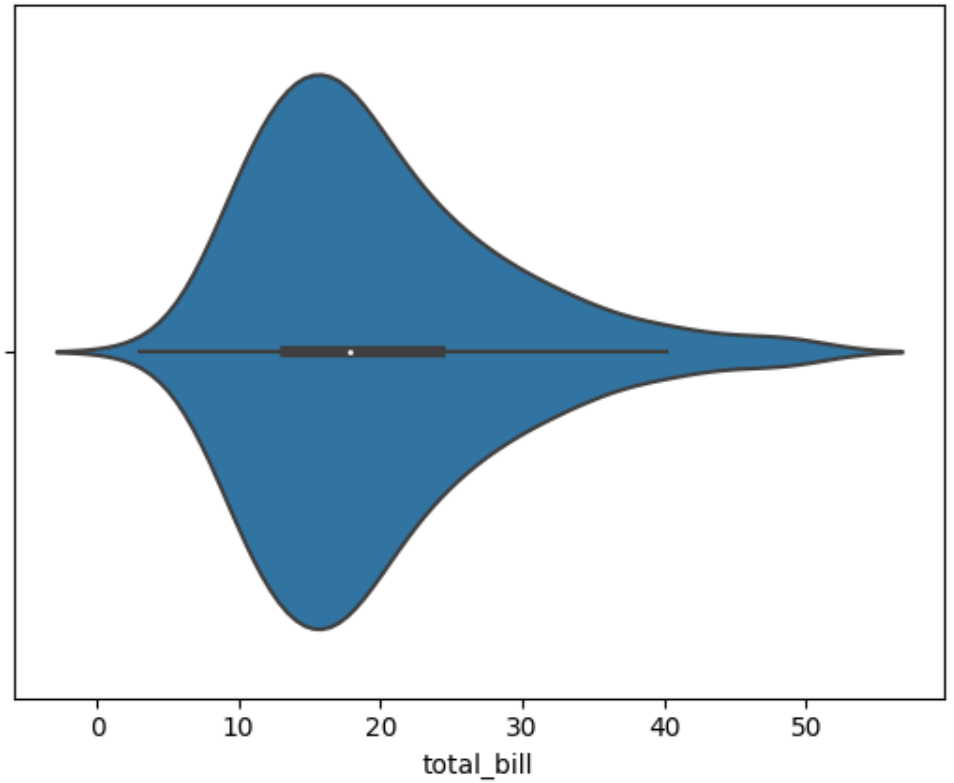
Diagrama de violín

- En un diagrama de violín, la forma de violín representa la distribución de los datos. La forma se dibuja simétrica alrededor de la mediana y muestra la densidad de los datos en diferentes valores. El ancho del violín en cualquier punto representa la densidad de los datos en ese punto. Por lo tanto, los violines más anchos indican áreas donde hay más datos, mientras que los violines más estrechos indican áreas con menos datos.

- Además, los puntos blancos dentro del violín representan la mediana y, a veces, también se muestran los valores mínimo y máximo. Los diagramas de violín también pueden ser combinados con un diagrama de caja para mostrar información adicional, como los cuartiles y los valores atípicos.

- El diagrama de violín es útil para comparar la distribución de los datos entre diferentes grupos o subconjuntos de datos. Al igual que el diagrama de caja, el diagrama de violín es una herramienta gráfica muy útil en estadística exploratoria y análisis de datos

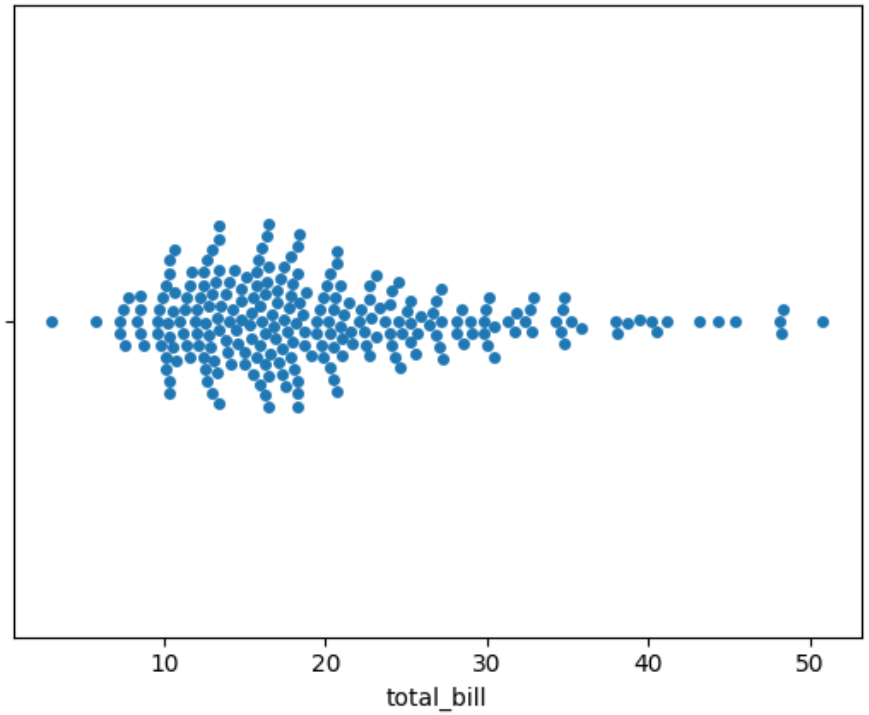
```
sns.violinplot(data=df, x='total_bill')
```



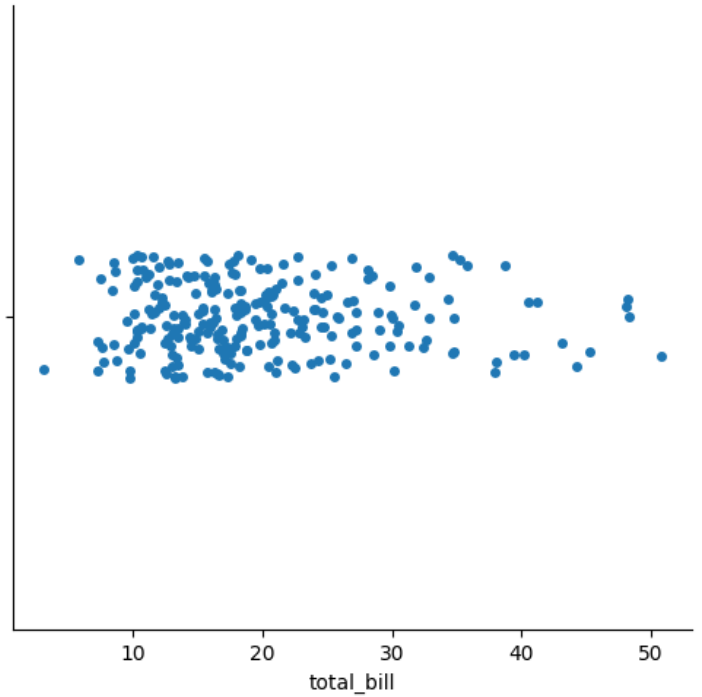
Más información:
https://seaborn.pydata.org/generated/seaborn.violinplot.html

Catplot y Swarmplot

```
sns.swarmplot(data=df, x='total_bill')
```




```
sns.catplot(data=df, x='total_bill')
```



Estos diagramas permiten armarse una idea de la distribución y dispersión de las mediciones.

Análisis Bivariado

KIBERNUM

Análisis Bivariado

- El análisis bivariado es una técnica estadística utilizada para explorar la relación entre dos variables diferentes en un conjunto de datos. Se analiza la relación entre dos variables, una variable independiente y otra variable dependiente, para determinar si existe alguna asociación o relación entre ellas.
- En el análisis bivariado, se pueden utilizar diversas técnicas estadísticas para analizar la relación entre las dos variables, como la correlación, la regresión lineal, la prueba de chi-cuadrado, el coeficiente de contingencia, entre otros.
- Por ejemplo, si se quisiera examinar la relación entre el nivel de educación y los ingresos, se podría llevar a cabo un análisis bivariado comparando los niveles de educación con los ingresos correspondientes. A través del análisis bivariado, se podría determinar si existe alguna asociación significativa entre el nivel de educación y los ingresos.

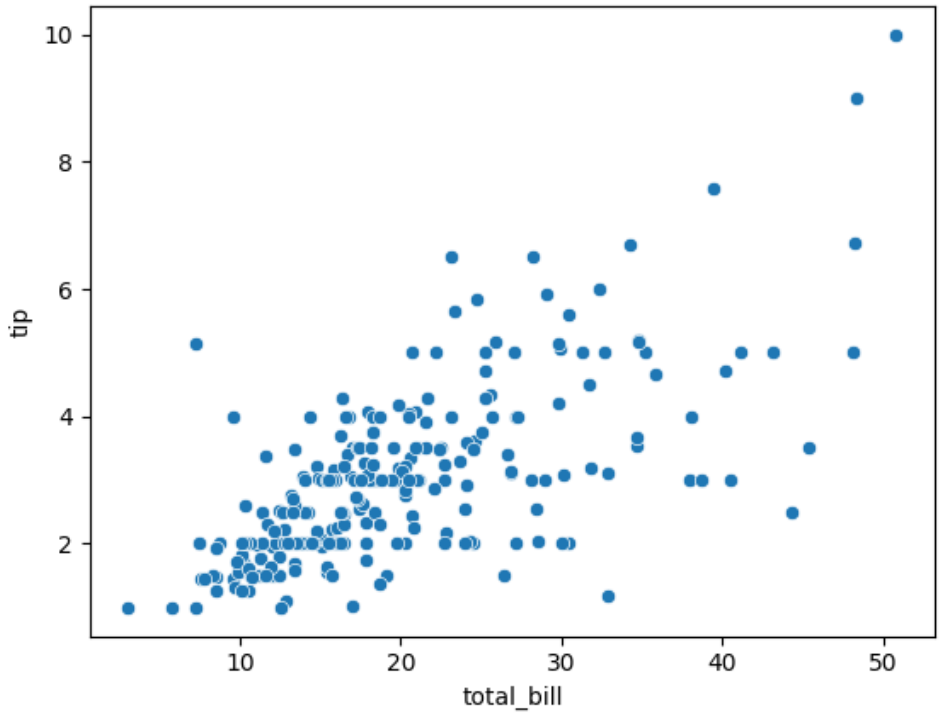
Diagrama de dispersión

Un diagrama de dispersión es una herramienta gráfica utilizada para visualizar la relación entre dos variables en un conjunto de datos. Se utiliza para identificar si existe una relación entre las dos variables y, de ser así, qué tipo de relación es.

En un diagrama de dispersión, los valores de una variable se trazan en el eje horizontal (eje X) y los valores de la otra variable se trazan en el eje vertical (eje Y). Cada punto en el diagrama representa una observación en el conjunto de datos y la ubicación de cada punto muestra los valores correspondientes de ambas variables.

Como se puede apreciar, existe una correlación positiva entre el total de la cuenta y el valor de la propina.

```
sns.scatterplot(data=df, x="total_bill", y="tip")
```

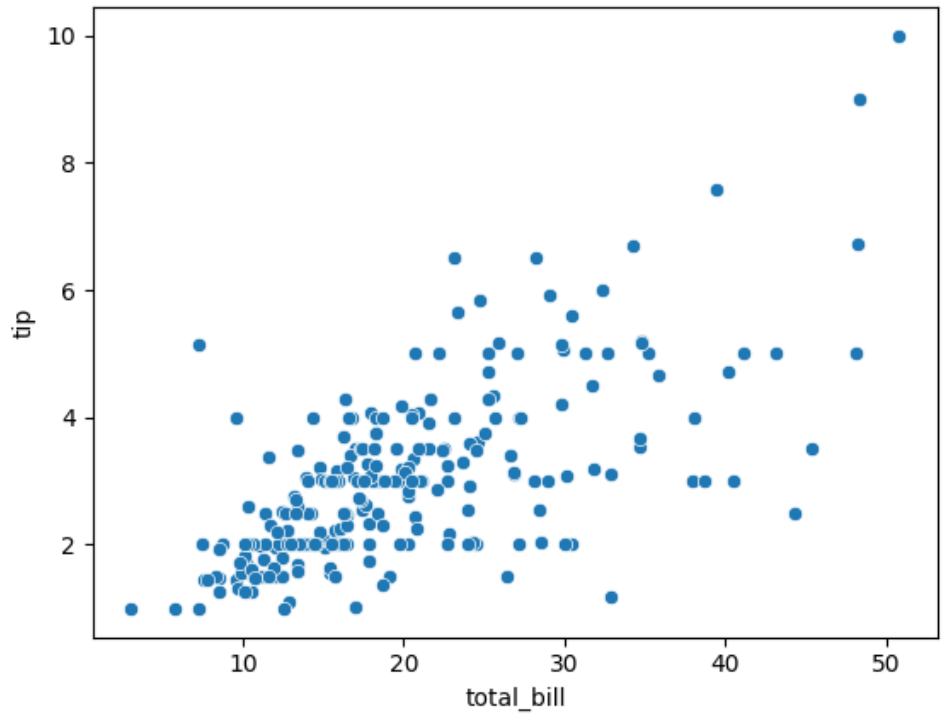


Más información:
https://seaborn.pydata.org/generated/seaborn.scatterplot.html

Diagrama de dispersión

También es posible colorear los puntos de acuerdo a alguna variable categórica con el parámetro hue. En este caso, se observa cómo la diferencia de sexo de quien paga la cuenta podría influir en el total de la cuenta y el valor de la propina.

```
sns.scatterplot(data=df, x="total_bill", y="tip")
```

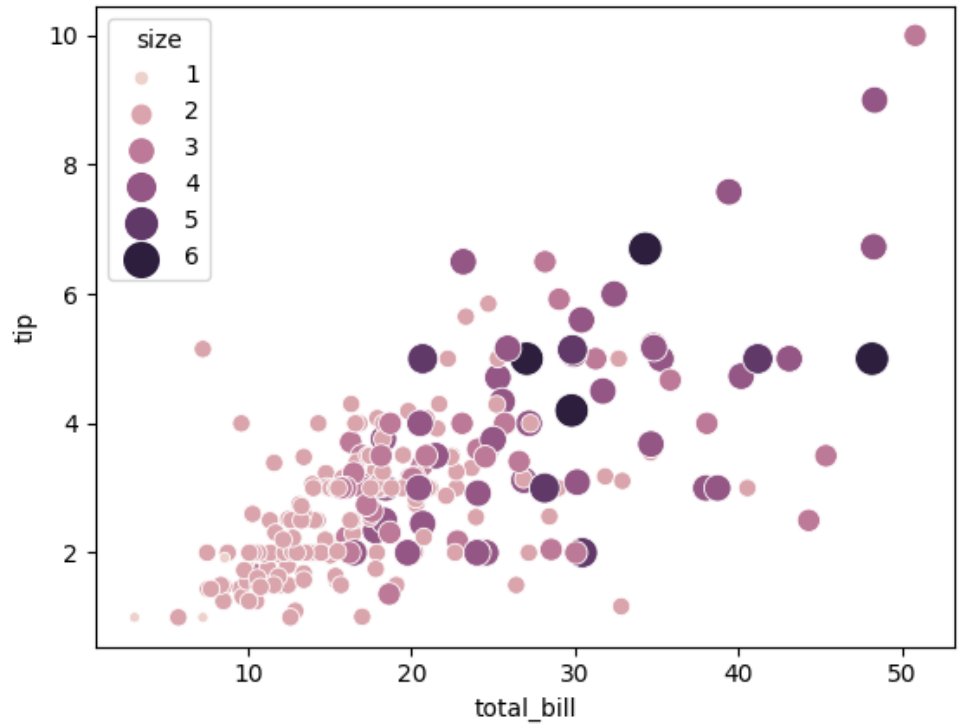


Más información:
https://seaborn.pydata.org/generated/seaborn.scatterplot.html

Diagrama de dispersión

También es posible colorear y reflejar la cantidad de comenzales en la mesa, jugando con la estética y el tamaño de cada medición.

```
sns.scatterplot(
    data=df, x="total_bill", y="tip", hue="size", size="size",
    sizes=(20, 200), legend="full"
)
```

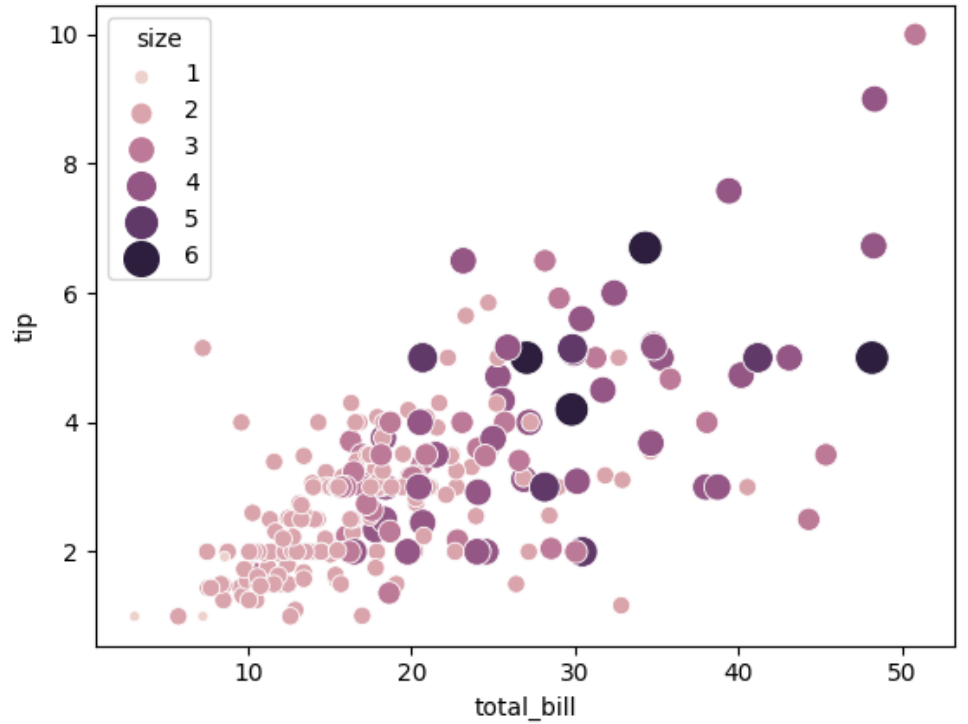


Más información:
https://seaborn.pydata.org/generated/seaborn.scatterplot.html

Diagrama de dispersión

También es posible colorear y reflejar la cantidad de comenzales en la mesa, jugando con la estética y el tamaño de cada medición.

```
sns.scatterplot(
    data=df, x="total_bill", y="tip", hue="size", size="size",
    sizes=(20, 200), legend="full"
)
```



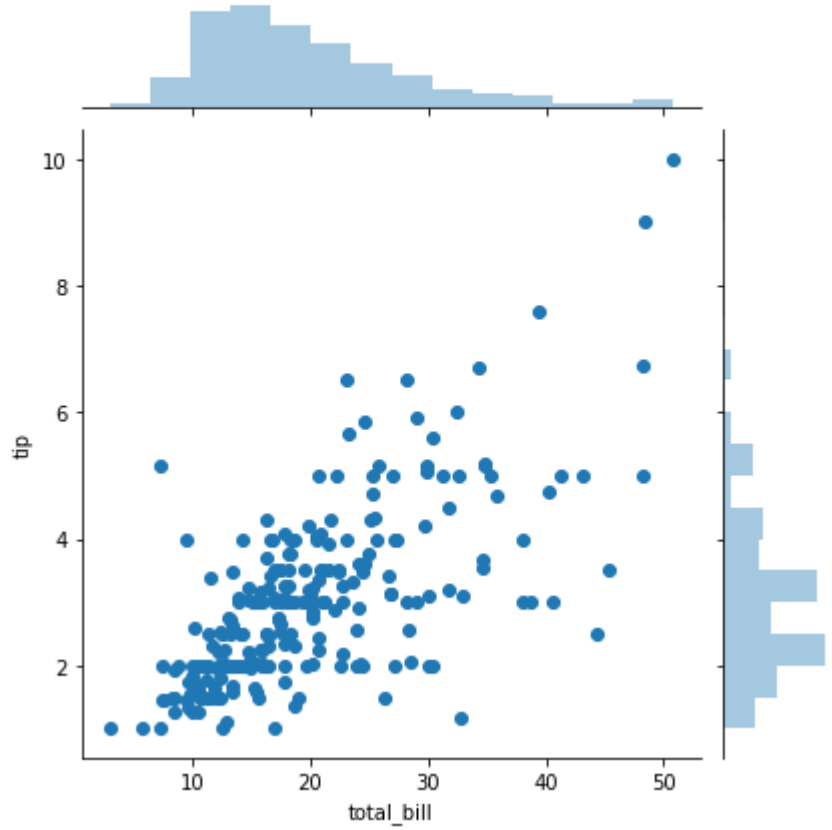
Más información:
https://seaborn.pydata.org/generated/seaborn.scatterplot.html

JointPlot

Es un tipo de gráfico combinado que permite ver la correlación entre dos variables. Para esto, despliega un diagrama de dispersión (scatterplot) y dos diagramas de distribución (uno para cada variable).

```
sns.jointplot(data=df, x='total_bill', y='tip')
```

```
<seaborn.axisgrid.JointGrid at 0x1a732e5cac8>
```

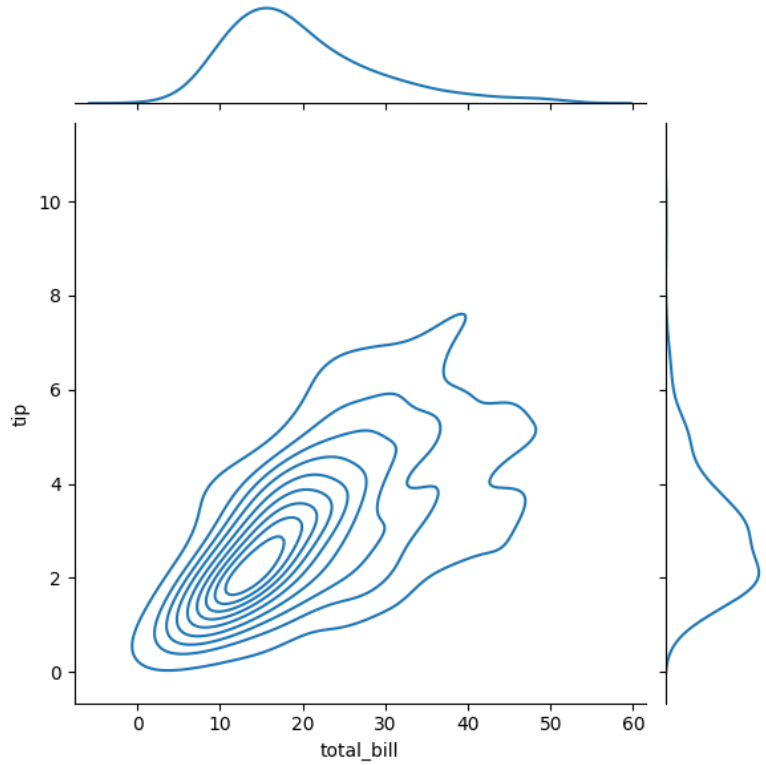


Más información:
https://seaborn.pydata.org/generated/seaborn.scatterplot.html

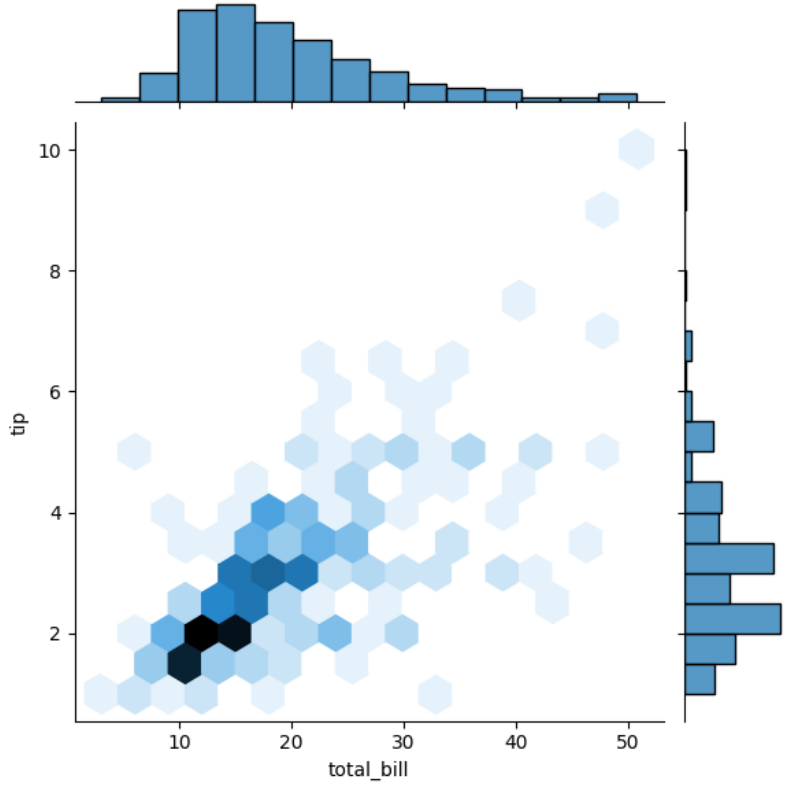
JointPlot

Algunas opciones adicionales de JointPlot, utilizando parámetro kind.

```
sns.jointplot(data=df, x='total_bill', y='tip', kind='kde')
```




```
sns.jointplot(data=df, x='total_bill', y='tip', kind='hex')
```



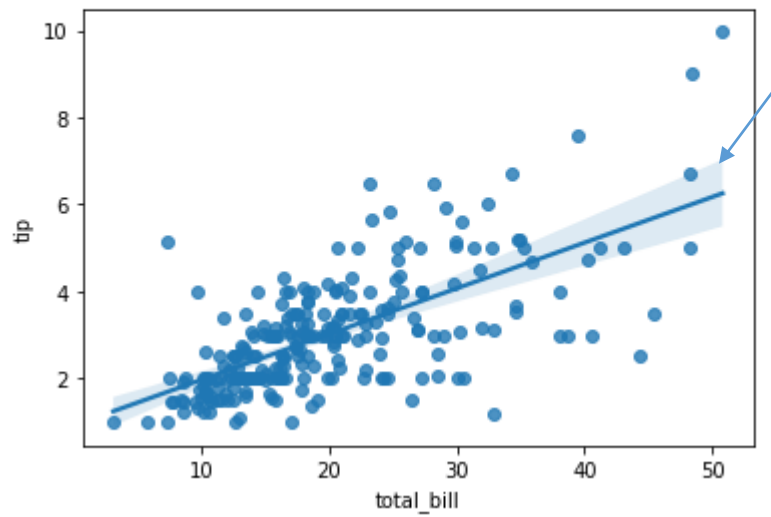
Más información:
https://seaborn.pydata.org/generated/seaborn.scatterplot.html

Gráfico de Regresión

La función regplot() permite crear un gráfico de dispersión y ajusta una recta de regresión con el intervalo de confianza de 95%. También es posible ajustar una curva de orden superior.

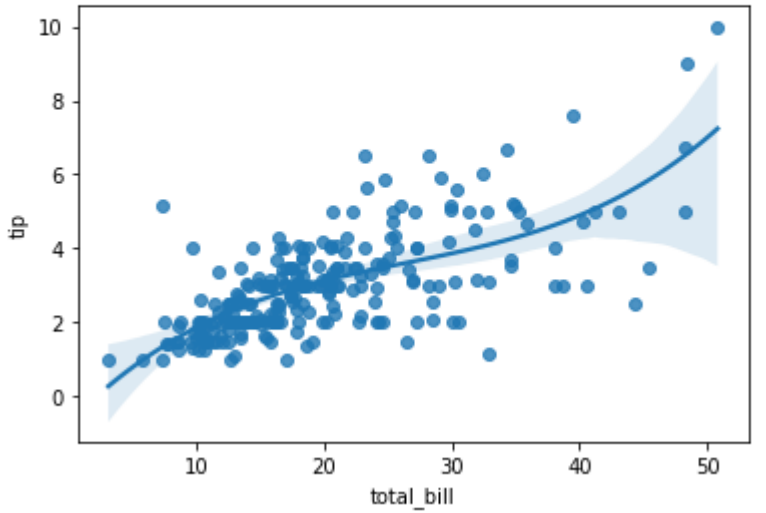
```
sns.regplot(x='total_bill', y='tip', data=df)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x1a734390688>
```

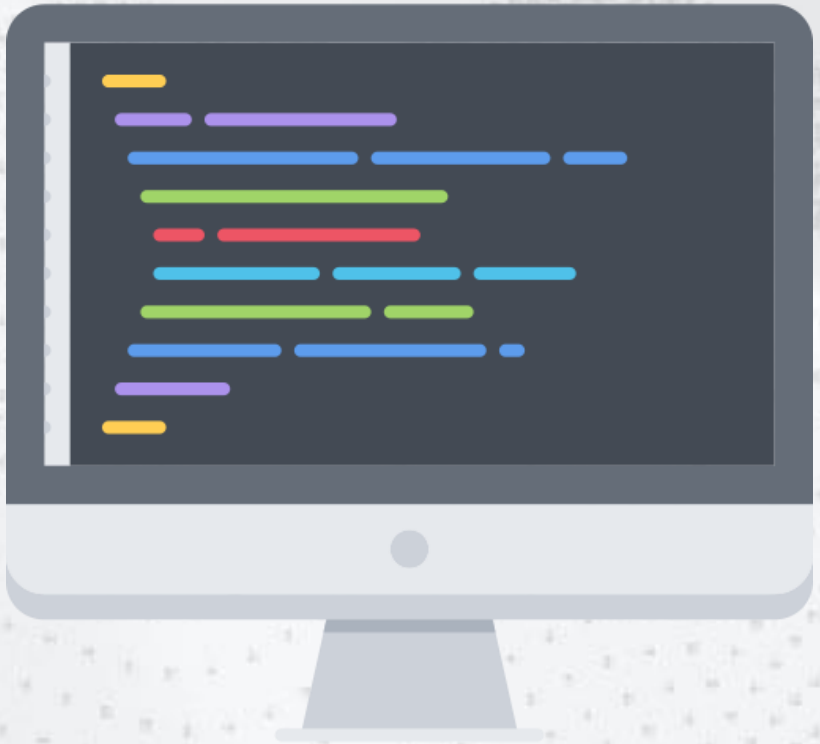



```
sns.regplot(x='total_bill', y='tip', data=df, order=3)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x1a7344cce48>
```



Más información:
https://seaborn.pydata.org/generated/seaborn.scatterplot.html



Análisis Multivariado

KIBERNUM

Análisis Multivariado

- El análisis multivariado es una técnica estadística utilizada para analizar y entender la relación entre tres o más variables en un conjunto de datos. A diferencia del análisis bivariado, que examina la relación entre dos variables, el análisis multivariado considera múltiples variables al mismo tiempo.

- El análisis multivariado incluye una variedad de técnicas estadísticas avanzadas que permiten identificar patrones y relaciones entre múltiples variables. Algunas de estas técnicas incluyen el análisis de regresión múltiple, el análisis factorial, el análisis de componentes principales, el análisis de conglomerados y la discriminación.

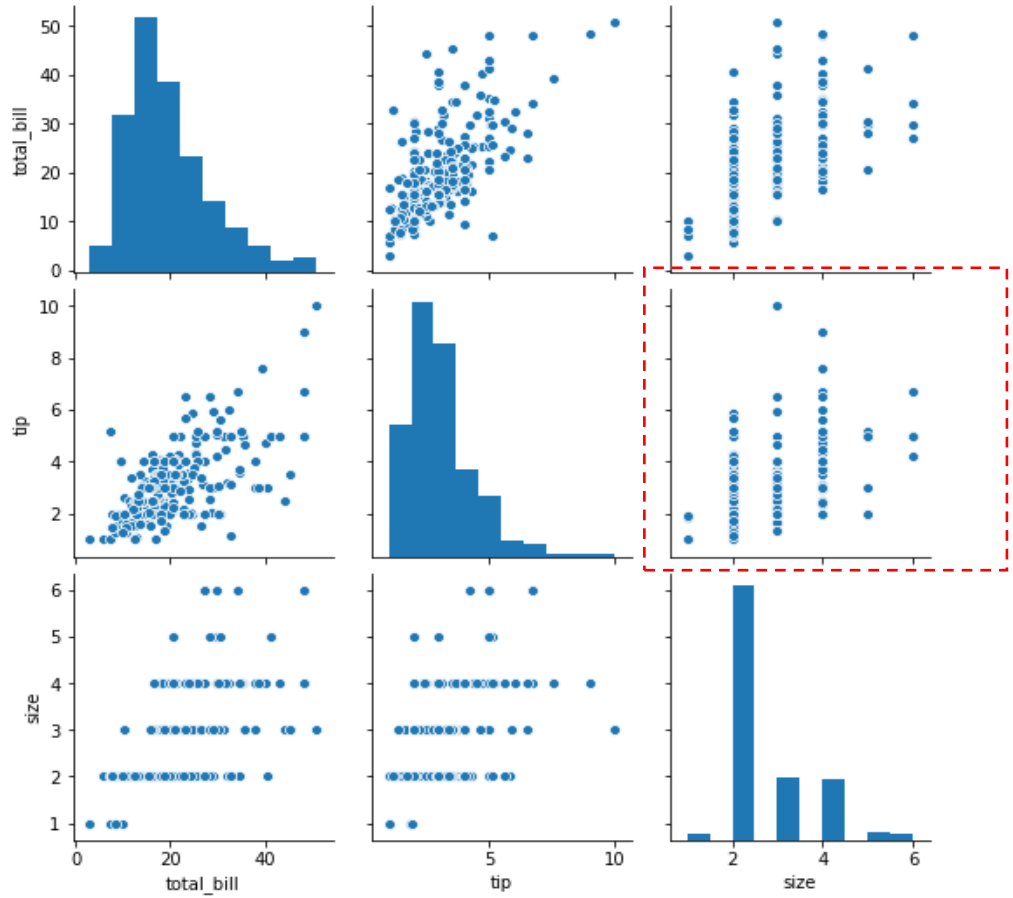
- Estas técnicas permiten a los investigadores identificar las variables que son más importantes en un conjunto de datos, identificar patrones complejos entre múltiples variables y proporcionar una comprensión más profunda de los datos y las relaciones entre ellos.

Análisis de Distribución

Permite crear de manera rápida gráficos que permitan vusualizar la relación entre las variables del dataframe.

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

```
<seaborn.axisgrid.PairGrid at 0x1a7330a3088>
```



Por ejemplo, este es un diagrama de dispersión que compara las variables **size** y **tip**.

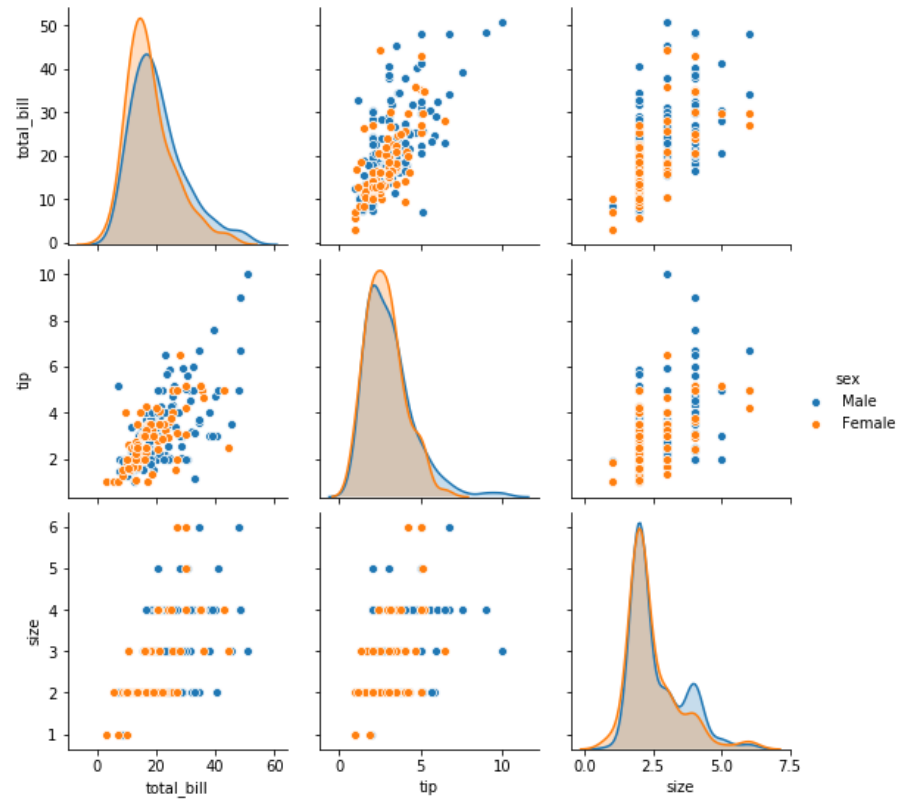
Los gráficos diagonales corresponden a histogramas, ya que no tiene sentido realizar un diagrama de dispersión de la variable comparada consigo misma.

Pairplot

- **Si bien es cierto que, en estricto rigor, un pairplot no permite ver la relación de todas las variables a la vez, al menos da una buena idea de cómo se parean las variables.**
- **Con el parámetro hue podemos considerar, además, la incidencia de alguna variable categórica con lo cual ya se podría apreciar un análisis multivariable.**

```
sns.pairplot(df, hue='sex')
```

```
<seaborn.axisgrid.PairGrid at 0x1a733ba49c8>
```

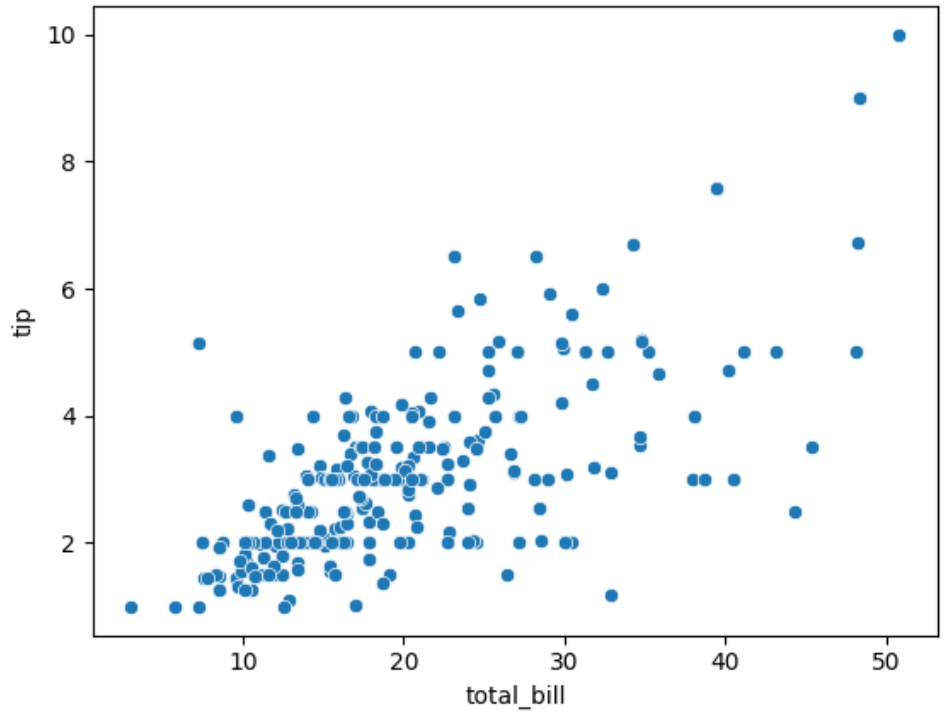


Más información:
https://seaborn.pydata.org/generated/seaborn.scatterplot.html

Utilizando Hue, Size y Style

Como se puede observar en este ejemplo, es posible agregar parámetros para diferenciar el color, el tamaño y el estilo del marcador, con lo cual podemos apreciar la incidencia de 5 variables en esta visualización.

```
sns.scatterplot(data=df, x="total_bill", y="tip", hue='sex', size='size', style="time")
```



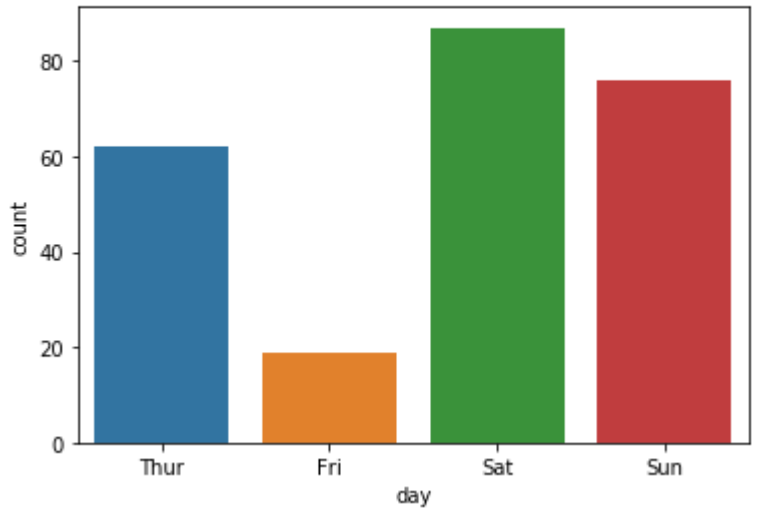
Análisis de Variables Categóricas

KIBERNUM

Countplot

Muestra el conteo de mediciones de los valores de una variable categórica en un set de datos. El resultado es similar al agrupamiento.

```
sns.countplot(x="day", data=df);
```



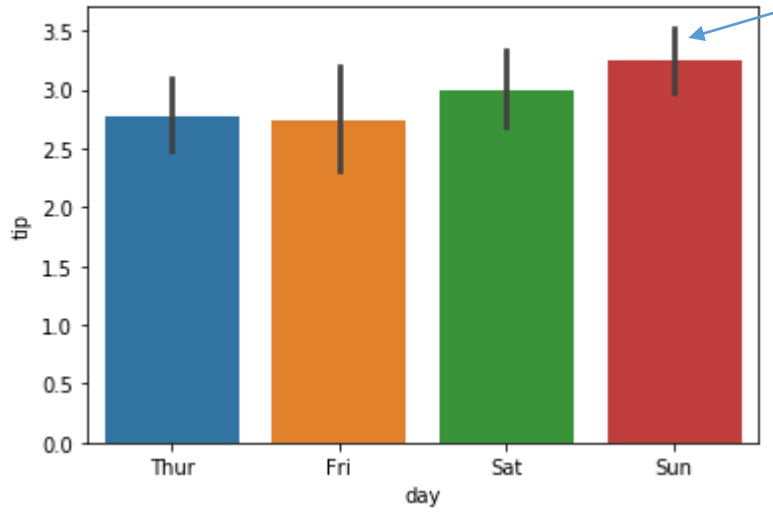

```
df.groupby('day').count()
```

day	total_bill	tip	sex	smoker	time	size
Thur	62	62	62	62	62	62
Fri	19	19	19	19	19	19
Sat	87	87	87	87	87	87
Sun	76	76	76	76	76	76

Barplot

Muestra estimaciones de puntos (por defecto la media) e intervalos de confianza.

```
sns.barplot(x='day', y='tip', data=df);
```



Intervalo de confianza

```
df.groupby('day').mean()
```

day	total_bill	tip	size
Thur	17.682742	2.771452	2.451613
Fri	17.151579	2.734737	2.105263
Sat	20.441379	2.993103	2.517241
Sun	21.410000	3.255132	2.842105

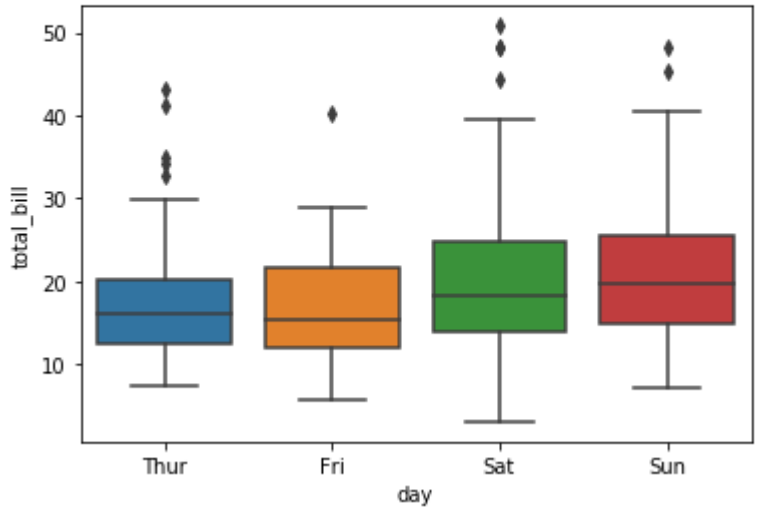
Por defecto muestra el **promedio**, pero con el parámetro estimate, se pueden setear otros estimadores.

Diagrama de cajas

El diagram de cajas muestra la distribución de datos cuantitativos en una forma que facilita la comparación entre variables entre los distintos nivels de una variable categórica. Utilizando el parámetro hue podemos especificar otra variable categórica para hacer comparaciones.

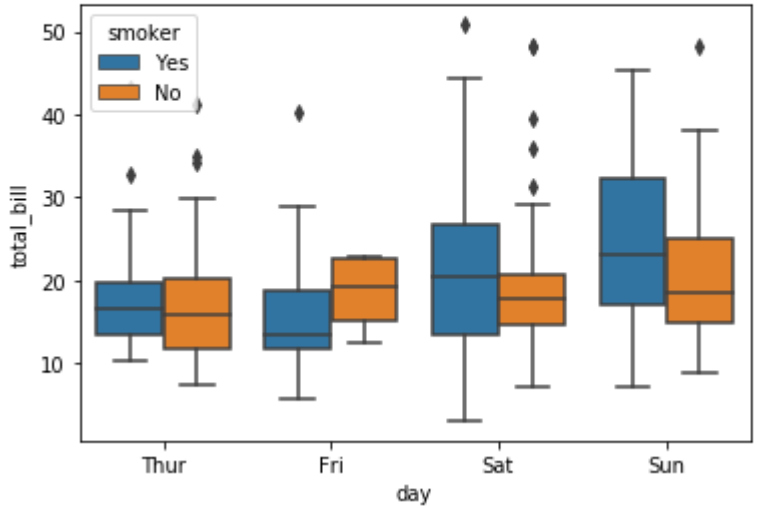
```
sns.boxplot(x='day', y='total_bill', data=df)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x1a73572c888>
```




```
sns.boxplot(x='day', y='total_bill', data=df, hue='smoker')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x1a735825c48>
```



Más información:
https://seaborn.pydata.org/generated/seaborn.violinplot.html

Gráficos de Matrices

KIBERNUM

Diagrama de Calor

El diagrama de calor representa los valores de una matriz con distintos tonos de colores. Para ejemplificar, utilizaremos el set de datos flights que seaborn contiene.

```
import seaborn as sns
%matplotlib inline
```

```
vuelos = sns.load_dataset('flights')
```

```
vuelos.head()
```

	year	month	passengers
0	1949	January	112
1	1949	February	118
2	1949	March	132
3	1949	April	129
4	1949	May	121

Más información:
https://seaborn.pydata.org/generated/seaborn.displot.html

KIBERNUM

Diagrama de Calor

Para realizar un diagrama de calor, debemos tener una matriz en donde los índices sean las variables categóricas y el contenido de la matriz los valores a graficar. Para realizar esto, utilizaremos la función de pivoteo.

```
p = vuelos.pivot_table(index='month', columns='year', values='passengers')
p
```

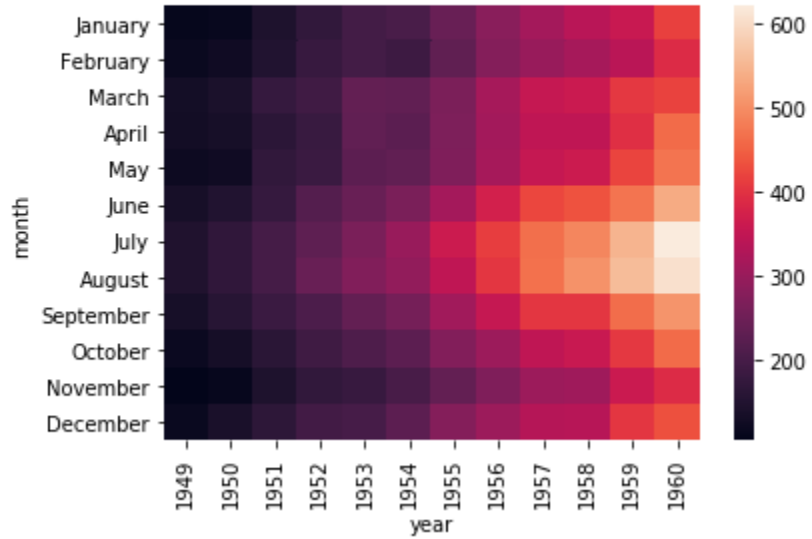
year	1949	1950	1951	1952	1953	1954	1955	1956	1957	1958	1959	1960
month												
January	112	115	145	171	196	204	242	284	315	340	360	417
February	118	126	150	180	196	188	233	277	301	318	342	391
March	132	141	178	193	236	235	267	317	356	362	406	419
April	129	135	163	181	235	227	269	313	348	348	396	461
May	121	125	172	183	229	234	270	318	355	363	420	472
June	135	149	178	218	243	264	315	374	422	435	472	535
July	148	170	199	230	264	302	364	413	465	491	548	622
August	148	170	199	242	272	293	347	405	467	505	559	606
September	136	158	184	209	237	259	312	355	404	404	463	508
October	119	133	162	191	211	229	274	306	347	359	407	461
November	104	114	146	172	180	203	237	271	305	310	362	390
December	118	140	166	194	201	229	278	306	336	337	405	432

Diagrama de Calor

Ahora que contamos con la información en una estructura matricial, podemos desplegar el diagrama de calor.

```
sns.heatmap(p)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x1ca3652bc50>
```



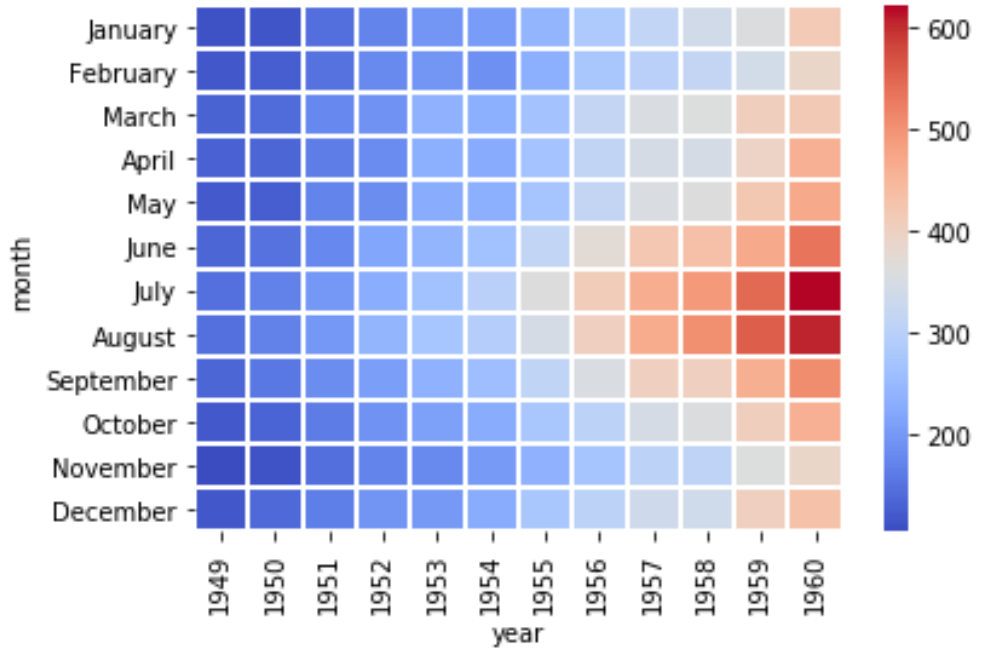
Nótese que con este gráfico podemos rápidamente concluir que en el transcurso de los años, los vuelos se han ido concentrando en los meses de Julio y Agosto. Esto corresponde a vacaciones de verano en USA, por lo tanto tiene mucho sentido.

Diagrama de Calor

En el siguiente mapa de calor hemos personalizado el mapa de colores (cmap) y hemos espaciado con una línea grande entre cada elemento (linecolor y linewidth).

```
sns.heatmap(p, cmap='coolwarm', linecolor='white', linewidths=1)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x1ca36e01cc0>
```



Grillas de Diagramas

KIBERNUM

PairGrid

Para ilustrar el uso de grillas en seaborn, utilizaremos el dataset iris.

```
import seaborn as sns
%matplotlib inline
```

```
iris = sns.load_dataset('iris')
```

```
iris.head()
```

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa

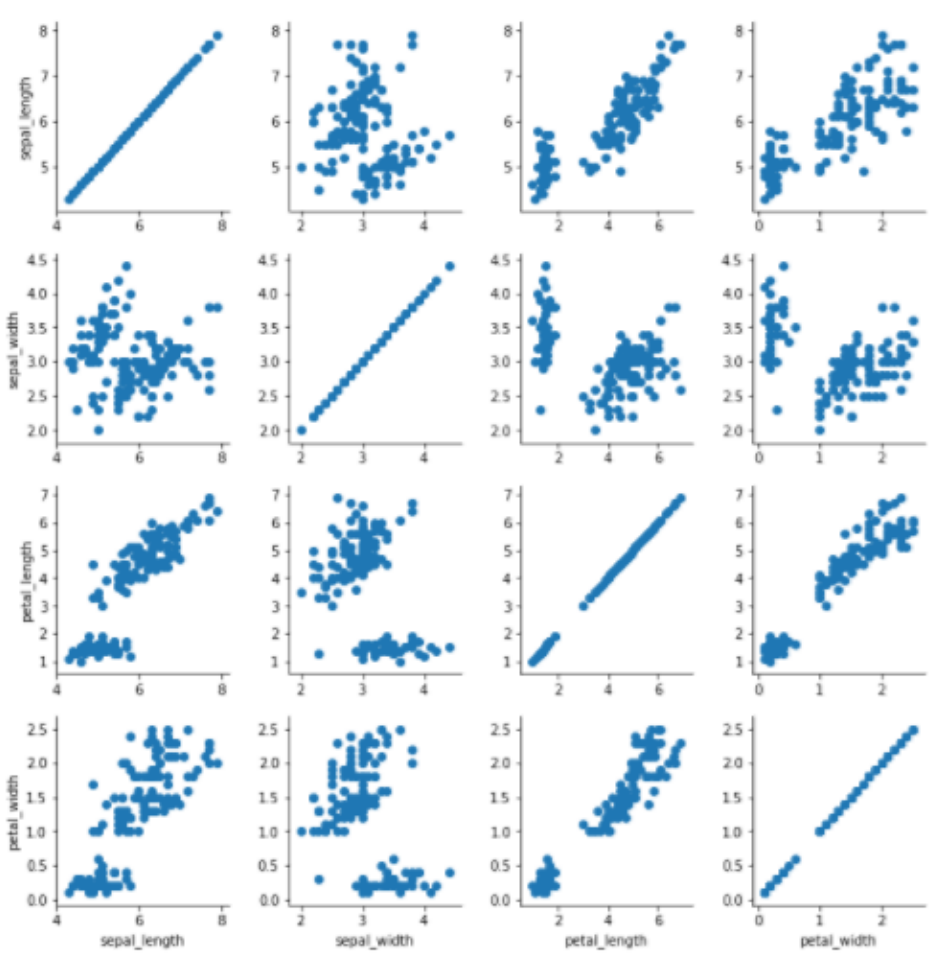
```
iris['species'].unique()
```

```
array(['setosa', 'versicolor', 'virginica'], dtype=object)
```

PairGrid

Ahora crearemos una grilla y la llenaremos con diagramas de dispersión.

```
grid = sns.PairGrid(iris)
grid.map(plt.scatter)
```

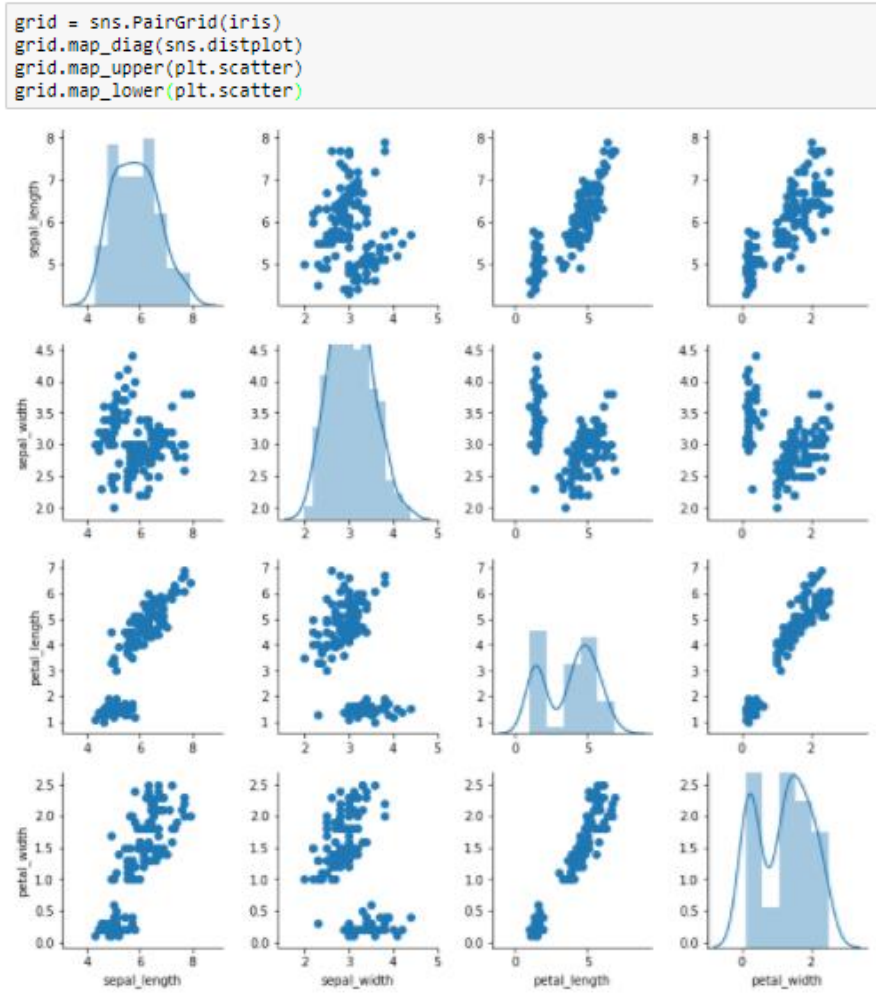


PairGrid

Ahora crearemos una grilla y la llenaremos con diagramas de dispersión.

Podemos personalizar aún más los gráficos seleccionando el diagrama que estimemos conveniente. Se pueden utilizar tanto objetos de diagrama seaborn como matplotlib.

```
grid = sns.PairGrid(iris)
grid.map_diag(sns.distplot)
grid.map_upper(plt.scatter)
grid.map_lower(plt.scatter)
```



Grilla de Facetas

KIBERNUM

Grilla de Facetas

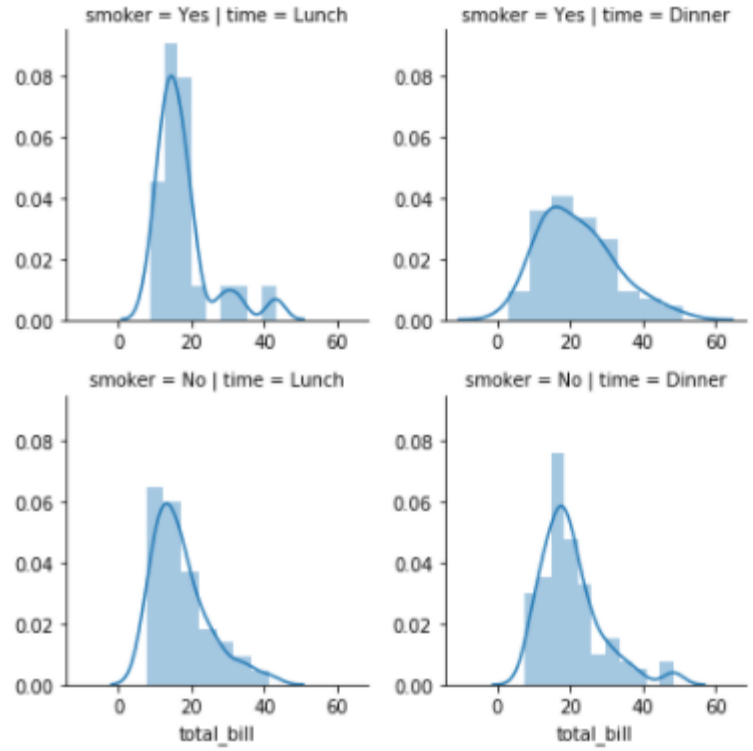
- Una grilla de facetas es una herramienta de visualización de datos que se utiliza para comparar varias visualizaciones de datos en una sola página. También se conoce como grilla de múltiples paneles o grilla de múltiples gráficos.
- La idea detrás de una grilla de facetas es que se pueden mostrar varias visualizaciones de datos al mismo tiempo en una sola página, lo que permite una comparación rápida y fácil entre ellas. Cada visualización de datos se muestra en un panel separado, y los paneles se organizan en filas y columnas en una grilla.
- La grilla de facetas es **particularmente útil para visualizar datos multivariados**, donde hay varias variables que se quieren comparar simultáneamente. Con una grilla de facetas, se pueden explorar las relaciones entre diferentes variables de manera más rápida y eficiente que si se examinara cada visualización de datos por separado.
- Las grillas de facetas se pueden utilizar con varios tipos de gráficos, incluyendo histogramas, diagramas de dispersión, diagramas de caja y bigotes, gráficos de barras, entre otros. También se pueden personalizar para incluir títulos, etiquetas de eje y leyendas para cada panel.

FacetGrid

Volvamos al dataset tips. Ahora crearemos una grilla de gráficos de acuerdo a las combinaciones de las variables categóricas.

```
grid = sns.FacetGrid(data=propinas, col='time', row='smoker')
grid.map(sns.distplot, 'total_bill')
```

```
<seaborn.axisgrid.FacetGrid at 0x1a6718e12b0>
```

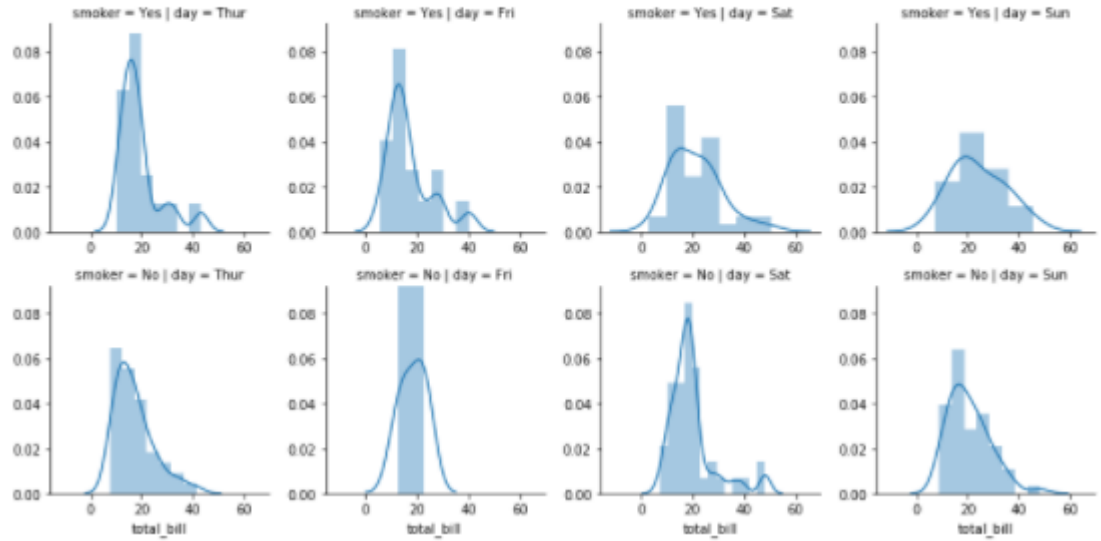


FacetGrid

En este ejemplo, queremos graficar de acuerdo a los distintos días de la semana y si es fumador o no.

```
grid = sns.FacetGrid(data=propinas, col='day', row='smoker')
grid.map(sns.distplot, 'total_bill')
```

```
<seaborn.axisgrid.FacetGrid at 0x1a67337aa58>
```

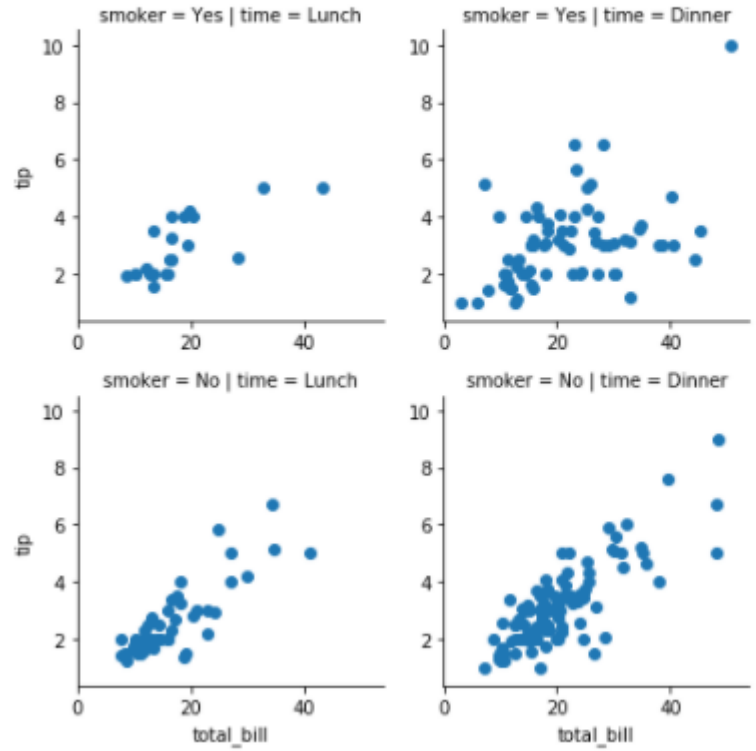


FacetGrid

En este ejemplo, graficamos diagramas de dispersión en la grilla de facetas.

```
grid = sns.FacetGrid(data=propinas, col='time', row='smoker')
grid.map(plt.scatter, 'total_bill','tip')
```

```
<seaborn.axisgrid.FacetGrid at 0x1a671805f28>
```



Gracias

KIBERNUM