

Texto y anotaciones en coordenadas

La función `text()` de Matplotlib recibe como parámetros la posición y la cadena de texto a escribir; y puedes usarla en cualquier gráfico, independientemente de la plataforma de trazado que hayas empleado. Para exemplificar su uso, leeremos en un `dataframe` (`languages`) el contenido del archivo **Languages.csv** (ubicado en el apartado "Archivos adjuntos" de la plataforma), que almacena los códigos de los países y las lenguas habladas, indicando el porcentaje y si es oficial o no.

	CountryCode	Language	IsOfficial	Percentage
0	ABW	Dutch	T	5.3
1	ABW	English	F	9.5
2	ABW	Papiamento	F	76.7
3	ABW	Spanish	F	7.4
...
980	ZWE	English	T	2.2
981	ZWE	Ndebele	F	16.2
982	ZWE	Nyanja	F	2.2
983	ZWE	Shona	F	72.1

984 rows × 4 columns

Filtremos el `dataframe` anterior para mantener únicamente la información del idioma español en México y sus fronteras, y las columnas `CountryCode` y `Percentage`; así el código para la escritura resultará más legible:

```
mexico_borders = languages.loc[(languages['CountryCode'].isin(['MEX', 'USA', 'BLZ', 'GTM'])) &
(languages['Language']=='Spanish'), ['CountryCode', 'Percentage']]
mexico_borders.set_index('CountryCode', inplace=True)
```

CountryCode	Percentage
BLZ	31.6
GTM	64.7
MEX	92.1
USA	7.5

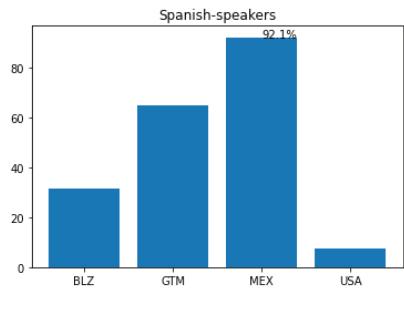
Para incluir, por ejemplo, el porcentaje de personas que hablan español en México (92.1%) en un gráfico de barras generado del `dataframe` anterior, puedes utilizar el siguiente código:

```
plt.text(2, 92.1, '92.1%')
```

Puedes incorporarlo a cualquier gráfico, como lo haces con otras propiedades disponibles únicamente desde Matplotlib.

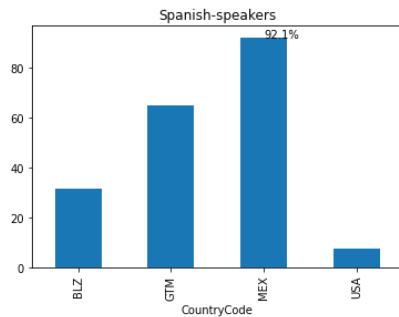
Con Matplotlib

```
plt.bar(mexico_borders.index, 'Percentage', data=mexico_borders)
plt.text(2, 92.1, '92.1%')
```



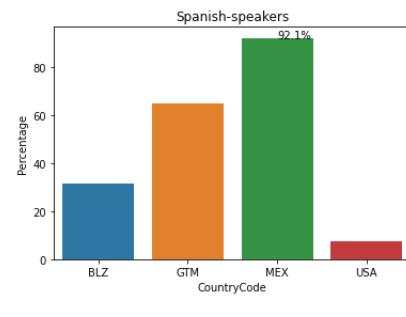
Con Pandas

```
mexico_borders['Percentage'].plot.bar()
plt.text(2, 92.1, '92.1%')
```



Con Seaborn

```
sns.barplot(x=mexico_borders.index, y='Percentage', data=mexico_borders)
plt.text(2, 92.1, '92.1%')
```



Observa de la función `text()` que, por defecto, las coordenadas donde se ubica el texto están especificadas en coordenadas de datos: 2 en el eje horizontal porque MEX está en la posición 2 (la serie inicia en 0 para los gráficos de barras) y 92.1 en el eje vertical, que corresponde al porcentaje que queremos representar. Pero si en lugar de indicar los valores fijados, la información se extrae del *dataframe*, el código a emplear sería el siguiente:

```
value = mexico_borders.loc['MEX', 'Percentage']
> 92.1
plt.text('MEX', value, str(value)+'%)')
```

Así, para la ubicación en el eje x se busca '`'MEX'`' porque, en este caso, el `CountryCode` es el índice en el *dataframe*; y para el eje y, el valor extraído con la función `loc()` que es `value = 92.1`

Pero ¿qué pasaría si no conoces directamente el índice a buscar y lo tienes que extraer de una condición? Por ejemplo, si deseas representar el valor del porcentaje máximo en el *dataframe* (que en este caso coincide con MEX):

```
value = mexico_borders.loc[mexico_borders['Percentage']==mexico_borders['Percentage'].max(),
                            'Percentage']
> CountryCode
MEX      92.1
```

Como ves, el resultado almacenado en `value` es una serie, y para tener acceso al 92.1, es necesario utilizar la función `to_list()` y extraer el primer valor de la misma con el índice 0:

```
value = mexico_borders.loc[mexico_borders['Percentage']==mexico_borders['Percentage'].max(), 'Percentage']
value = value.to_list()[0]
> 92.1
```

Seguiremos ocupando los valores fijados, para facilitar la compresión de los ejemplos.



Puedes indicar la ubicación seleccionando un sistema de coordenadas diferente a través del parámetro `transform`. Este te permite posicionar el texto en:

- Un pixel en específico:

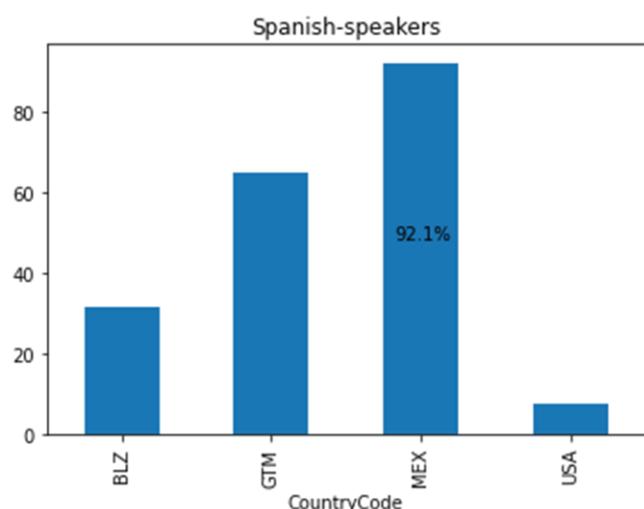
```
mexico_borders['Percentage'].plot.bar()  
plt.text(220,150,'92.1%',transform=None)
```

- Relación a los ejes:

```
ax = mexico_borders['Percentage'].plot.bar()  
plt.text(0.58,0.5,'92.1%',transform=ax.transAxes)
```

- Relación a la figura:

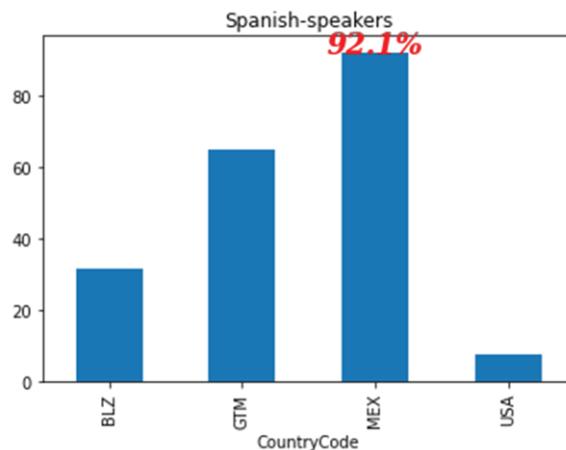
```
fig,ax = plt.subplots()  
mexico_borders['Percentage'].plot.bar(ax=ax)  
plt.text(0.58,0.5,'92.1%',transform=fig.transFigure)
```



Observa que cuando usas los ejes o la figura como referencia para las coordenadas, debes generar el objeto de gráfica (`ax`) o el contenedor (`fig`), respectivamente, y se considera que (0,0) es el límite inferior izquierdo y (1,1) es el límite superior derecho.

Hay muchas propiedades del texto que podrás personalizar. El siguiente código muestra algunas de ellas:

```
plt.text(2,92.1,'92.1%',family='serif',fontsize=18,fontweight='bold',style='italic',  
color='red',horizontalalignment='center')
```

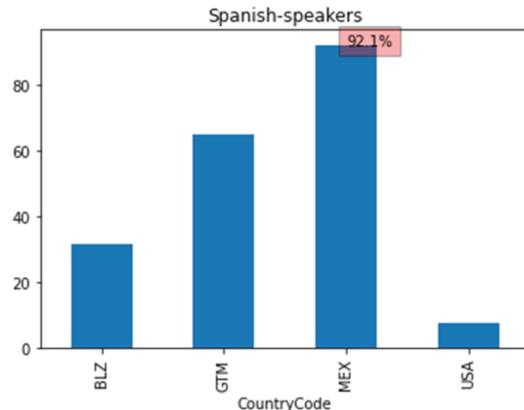


También puedes pasar las propiedades anteriores como un diccionario a la función `text()`:

```
props_font = {'family': 'serif',  
             'size': 18,  
             'weight': 'bold',  
             'style': 'italic',  
             'color': 'red',  
             'horizontalalignment': 'center'}  
plt.text(2,92.1,'92.1%',fontdict=props_font)
```

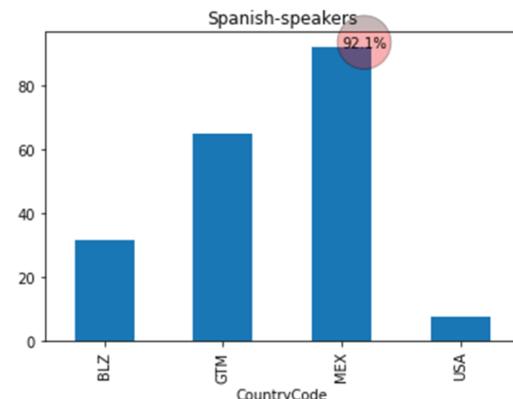
El texto también puedes resaltarlo a través de un objeto `bbox`, que posee entre sus propiedades el nivel de transparencia (`alpha`) y el espacio entre el contenido y el borde del elemento (`pad`):

```
plt.text(2, 92.1, '92.1%', bbox=dict(facecolor='red', alpha=0.3, pad=5))
```



Puedes especificar, además, el tipo de contenedor e indicar las propiedades creando previamente el diccionario:

```
props_bbox = {'boxstyle': 'circle',
              'facecolor': 'red',
              'alpha': 0.3}
plt.text(2, 92.1, '92.1%', bbox=props_bbox)
```

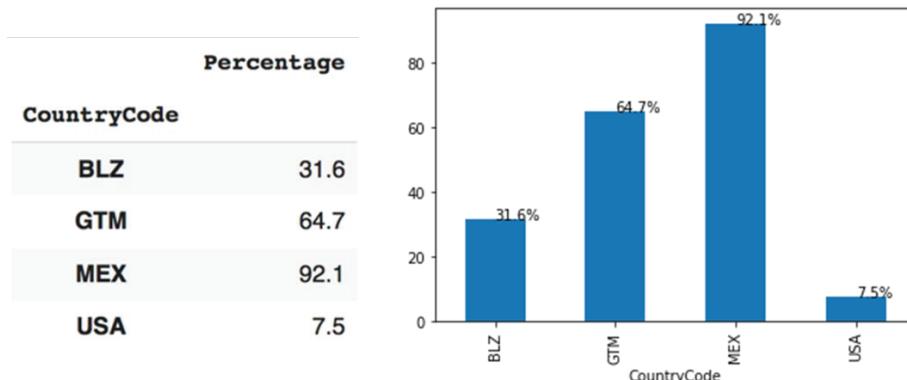


Algunos otros valores del estilo son:

'larrow' flecha izquierda, 'rarrow' flecha derecha,
'round' rectángulo con bordes redondeados

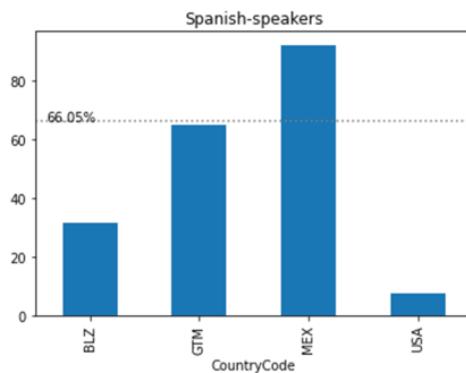
Para incluir los valores de todos los países, es necesario que hagas un recorrido por las filas del `dataframe` con un ciclo y extraer el porcentaje de hablantes con `mexico_borders.iloc[i,0]`:

```
mexico_borders['Percentage'].plot.bar()
for i in range(mexico_borders.shape[0]):
    plt.text(i,mexico_borders.iloc[i,0],str(mexico_borders.iloc[i,0])+'%')
```



Además de texto, puedes insertar líneas horizontales y/o verticales para indicar ciertos valores relevantes o de referencia, utilizando las funciones `axhline()` y `axvlines()`. Si en los gráficos anteriores quisieras añadir, por ejemplo, el promedio de hablantes del español en todos los países que incluyen este idioma, podrías hacer:

```
spanish_mean = (languages.loc[languages['Language']=='Spanish','Percentage']).mean()
plt.axhline(y=spanish_mean,color='red',linestyle=':')
plt.text(-0.4,spanish_mean,str(f'{spanish_mean:.2f}')+'%')
```



Para incluir expresiones matemáticas, puedes usar un subconjunto del sistema de tipografía **TEX**. Para ello, deberás utilizar cadenas sin formato precedidas con una `r` antes de las comillas y rodear el texto matemático con el signo de peso (`$`). El texto normal y el texto matemático se pueden intercalar dentro de la misma cadena.

Si en el gráfico anterior, deseas incluir la fórmula que calcula el valor (66.05%) como:

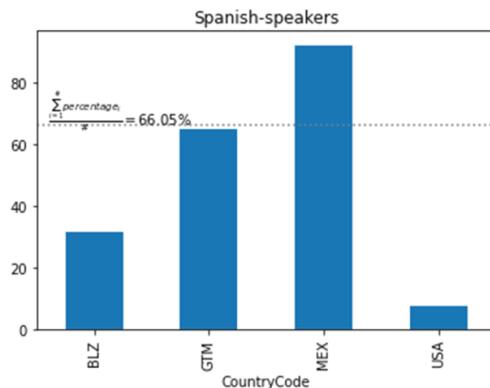
$$\frac{\sum_{i=1}^{\#} \text{percentage}_i}{\#}$$

Donde `#` es el número de países en cuyas lenguas se incluye el español, se usaría:

```
r'$\frac{\sum_{i=1}^{\#} \text{percentage}_i}{\#}$'
```

Este fragmento de texto matemático se puede combinar con el resto, como se muestra a continuación:

```
plt.text(-0.4,spanish_mean,r'$\frac{\sum_{i=1}^{\#} \text{percentage}_i}{\#}=$'
+str(f'{spanish_mean:.2f}')+'%')
```



Si ya has usado algún sistema de composición de textos basado en **T_EX**, los comandos te resultarán familiares.

Habrá casos en los que además de ubicar un texto en una posición determinada del gráfico, necesites puntualizar en alguna característica. Para ello, puedes usar la función `annotate()`, que además de considerar la ubicación del texto, recibe la ubicación de la anotación. Ambas ubicaciones deben ser enviadas como tuplas (x, y). Para probar su funcionamiento, veamos cómo están distribuidas todas las lenguas en México:

```
mexico = languages.loc[languages['CountryCode']=='MEX', ['Language', 'Percentage']]  
mexico.set_index('Language', inplace=True)
```

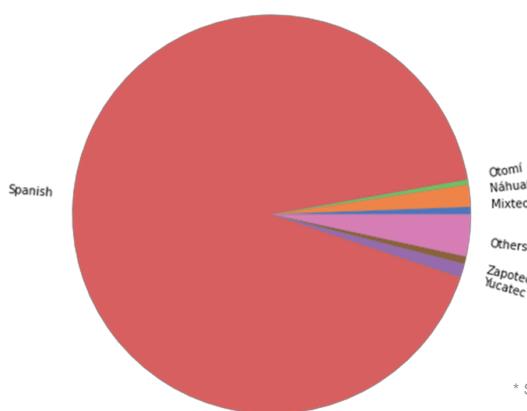
Language	Percentage
Mixtec	0.6
Náhuatl	1.8
Otomí	0.4
Spanish	92.1
Yucatec	1.1
Zapotec	0.6

Usaremos un gráfico circular para representarlas, pero antes incluyamos una nueva categoría (**Others**) para completar el 100% de los hablantes:

```
mexico.loc['Others']=[100 - mexico['Percentage'].sum()]
```

Language	Percentage
Mixtec	0.6
Náhuatl	1.8
Otomí	0.4
Spanish	92.1
Yucatec	1.1
Zapotec	0.6
Others	3.4

```
mexico.plot.pie(y='Percentage', labels=mexico.index, rotatelabels=True, figsize=(8, 8))
```

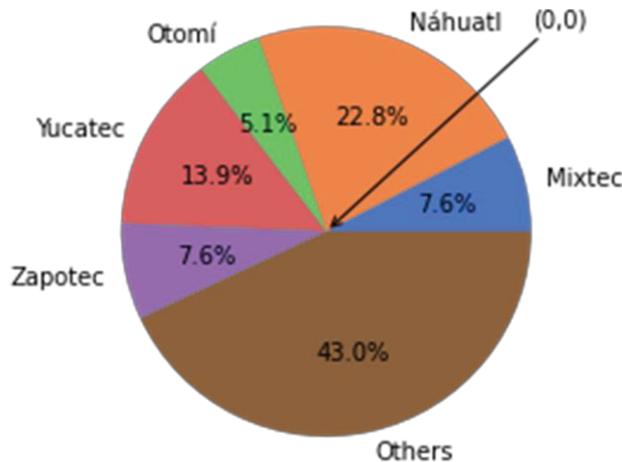


* Se utilizó la paleta muted.

Para tener mayor detalle, dejemos únicamente las lenguas no oficiales:

```
mexico_unofficial = mexico.drop(mexico.loc[mexico.index=='Spanish'].index)
mexico_unofficial.plot.pie(y='Percentage',labels=mexico_unofficial.index,autopct='%.1f%%')
```

En un gráfico circular, las coordenadas (0,0) se encuentran al centro. Esto se ha indicado en la figura con una anotación:



```
plt.annotate(' (0,0)',xy=(0,0),xytext=(1,1),arrowprops=dict(arrowstyle='->'))
```

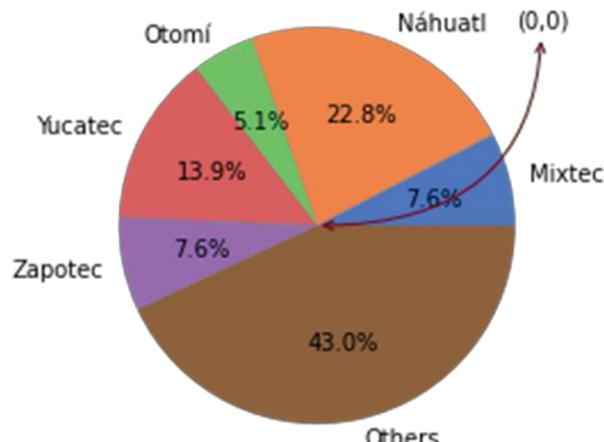
Del código anterior puedes notar que se especifican dos puntos con tuplas de coordenadas: el primero indica dónde irá la anotación; y el segundo, la ubicación del texto. La flecha, cuyas propiedades se definen con el parámetro `arrowprops`, conecta ambos puntos.

Hay muchos estilos de flecha:

'-' sin saeta, '-[' corchete en la anotación, '|-|' líneas en ambos extremos, '<->' saeta en el texto, '->' saeta completa en la anotación

También puedes modificar el estilo y color de la línea de conexión:

```
plt.annotate(' (0,0)',xy=(0,0),xytext=(1,1),
arrowprops=dict(arrowstyle='<->', connectionstyle='angle3',color='maroon'))
```



Si no estableces el estilo, puedes detallar otras propiedades para la flecha. Como en casos anteriores, el diccionario puede ser definido antes:

```
props_arrow = {'facecolor':'white',
               'width':0.5,
               'headwidth':8,
               'headlength':8}
plt.annotate('1.8% including Spanish', xy=(0.45,0.58),xytext=(1,0.6),arrowprops=props_arrow)
```

