**Dibujar un diagrama con Python**

La construcción de diagramas de Venn para los conjuntos parte correctamente de la correcta instalación de las librerías matplotlib y matplotlib\_venn.

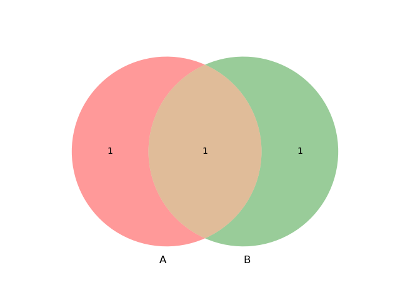
La librería “matplotlib\_venn” es un paquete de Python que provee funciones para dibujar diagramas de Venn de dos conjuntos (venn2) y de tres conjuntos (venn3), la cual se instala, vía pip:

pip install matplotlib\_venn

El comando “pip” también instalará las dependencias requeridas: SciPy, NumPy y matplotlib.

El procedimiento mínimo para dibujar un diagrama de Venn es el siguiente.

1. **from** matplotlib **import** pyplot **as** plt
2. **from** matplotlib\_venn **import** venn2
3. venn2((1, 1, 1))
4. plt.show()

[](https://recursospython.com/wp-content/uploads/2017/12/venn2.png)La función venn2() genera un diagrama de dos conjuntos, como se muestra a continuación.

Para identificar a cada uno de las regiones o subconjuntos (tres en diagramas de dos conjuntos), el módulo utiliza una nomenclatura que consiste en colocar un 1 para indicar que la sección está incluida en el conjunto y un 0 para indicar que está excluida.

El argumento indica el tamaño relativo de los tres subconjuntos, corresponden las regiones formadas, cuando se intersecan los 2 conjuntos por la intersección, en este orden: “10” (región de A que no está en B), “01” (región de B que no está en A) y “11” región de A y de B (intersección).

La tupla (1,1,1) dibujará proporcionalmente las regiones. Si la tupla es (1,2,1), entonces, el conjunto B será el doble del tamaño de A.

El parámetro set\_labels permite cambiar los nombres de los conjuntos mostrados en el diagrama.

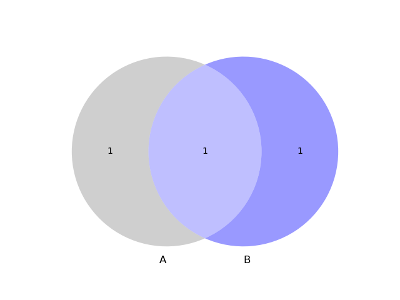
1. venn2((1, 1, 1), set\_labels=("A", "B"))

Indicando set\_labels=None se remueven las referencias de la imagen.

set\_colors determina el color de los conjuntos.

1. venn2((1, 1, 1), set\_colors=("#888888", "#0000FF"))

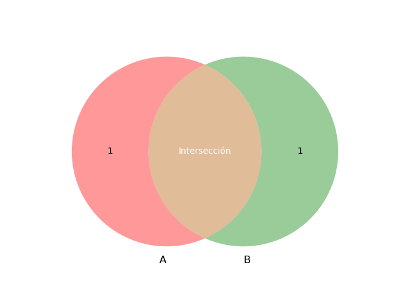
El color de las intersecciones es calculado automáticamente. Nótese que, por defecto, tienen un valor alpha de 0.4, por ejemplo, 60% de transparencia. Utilícense valores de 0.0 a 1.0 en el parámetro de ese mismo nombre.

[](https://recursospython.com/wp-content/uploads/2017/12/venn2-colored.png)# Deshabilita la transparencia.

1. venn2((1, 1, 1), set\_colors=("#888888", "#0000FF"), alpha=1)

En particular, los métodos de ésta que nos interesan son get\_label\_by\_id() y get\_patch\_by\_id(). Ambas toman un subconjunto y retornan instancias de matplotlib.text.Text y matplotlib.patches.PathPatchs, respectivamente. A partir de ellas podemos modificar individualmente el aspecto de cada subconjunto. Por ejemplo, el siguiente código cambia el color y el texto del subconjunto 11 (A∩B).

1. diagram = venn2((1, 1, 1))
2. diagram.get\_label\_by\_id("11").set\_text("Intersección")
3. diagram.get\_label\_by\_id("11").set\_color("#FFFFFF")

[](https://recursospython.com/wp-content/uploads/2017/12/venn2-text.png)

O bien alterar el color de un subconjunto en particular

1. # Establece el color del conjunto B.
2. diagram.get\_patch\_by\_id("01").set\_color("#0000FF")

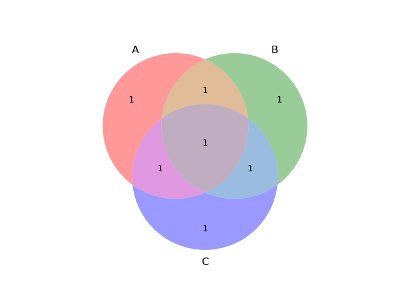
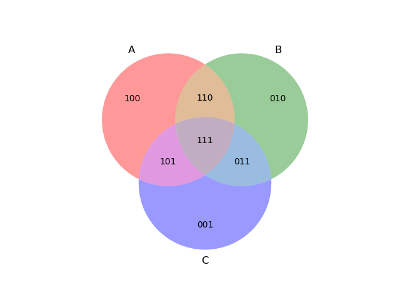
 El procedimiento mínimo para dibujar un diagrama de Venn para 3 conjuntos es el siguiente:

1. **from** matplotlib **import** pyplot **as** plt
2. **from** matplotlib\_venn **import** venn3
3. Venn3((1, 1, 1, 1,1, 1, 1))
4. plt.show()

En este orden de ideas, la función venn3() para 3 conjuntos “ABC” es esencialmente similar, a la de venn2(). Las regiones o subconjuntos formados que se distinguen son 7: “100” (región de A que no está ni en B n en C); “010” (región de B que no está ni en A ni en C); “001” (región de C que no está ni en A ni en B); “110” (región de A y B que no está en C); “101” (región de A y C que no está en B); “011” (región de B y C que no está en A); “111” (región de A, B y C).

La tupla (1,1,1,1,1,1,1) dibujará proporcionalmente las 7 regiones que se forman con los 3 conjuntos. El primer argumento será ahora una tupla de siete elementos que se corresponden con los siete subconjuntos en el siguiente orden: Abc (100), aBc(101), ABc(110), abC(001), AbC(101), aBC(011), ABC(111).

1. **from** matplotlib\_venn **import** venn3
2. diagram = venn3((1, 1, 1, 1, 1, 1, 1))

[](https://recursospython.com/wp-content/uploads/2018/01/venn3.png)[](https://recursospython.com/wp-content/uploads/2018/01/venn3-subconjuntos.png)

La imagen a continuación muestra cada uno de ellos con su respectivo identificador.

Generado por el siguiente código:

1. diagram = venn3((3, 3, 1, 3, 1, 1, 1))
2. **for** subset **in** ("111", "110", "101", "100", "011", "010", "001"):
3. diagram.get\_label\_by\_id(subset).set\_text(subset)

Para esta función también aplican los argumentos alpha, set\_labels y set\_colors. Estos últimos dos argumentos requieren, lógicamente, tuplas de tres elementos.