Diagrama Circular con Python

Guatemala, 24 de junio de 2022

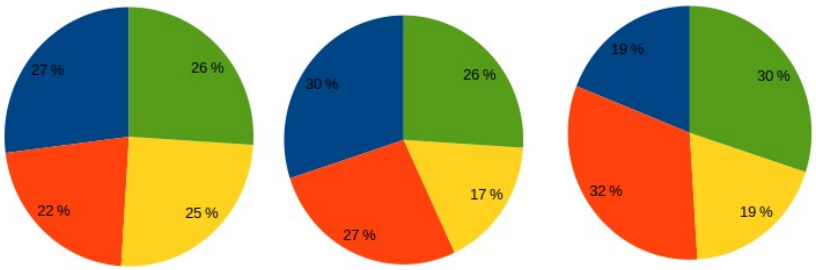
Pablo Sao Alonzo[1]
1. Consultor técnico, Solution Design of Centroamerica, Guatemala.

Los diagramas circulares o de pie (también conocida como pay) no suelen ser recomendados para la presentación de información, teniendo en cuenta que se suele recomendar representar como máximo seis distribuciones de frecuencia o categorías cualitativas, debido a que no son la mejor forma de mostrar comparaciones. En la imagen 1, podemos observar tres diagramas circulares, donde a la vista se pueden ver algunas diferencias; sin embargo, otras áreas suelen ser percibidas como idénticas (Anderson *et al.*, 2016).

Imagen 1: Diagramas circulares con diferentes valores.

Por lo que si se desea utilizar en nuestro análisis este tipo de gráfico, es recomendable agregar los valores dentro del área de la categoría, dando más claridad de la frecuencia relativa que se está representando (imagen 2). Razón por la que se debe de evitar utilizar representaciones con sombra y/o en 3D, debido a que estas distorsionan el "tamaño" del valor agregando más dificultad en la percepción del tamaño del área. Si se desea realizar una mejor comparación se debe utilizar un gráfico de barras, ya que las personas tienen un mayor criterio de análisis con las diferencias en longitudes que en ángulos o porciones (Anderson *et al.*, 2016; Nussbaumer, 2015).

Imagen 2: Diagramas circulares con diferentes valores y etiquetas en áreas.



Para poder elaborar este diagrama debemos obtener la frecuencia relativa de cada elemento que pertenece a una clase. Para un set de datos con *n* observaciones, la frecuencia relativa se determina de la siguiente forma:

$$Frecuencia\,Porcentual\,de\,Clase = \left(\frac{Frecuencia\,de\,Clase}{n}\right) * 100$$

Utilizando el set de datos del muestreo de zarigüeyas extraído de Kaggle desde el siguiente enlace: ***https://www.kaggle.com/abrambeyer/openintro-possum***, o desde el siguiente repositorio en donde se podrá encontrar el código utilizado: ***https://github.com/sdesignca/blog-ps-diagrama-circular-python***

Tomaremos la categoría de sexo (macho y hembra), donde nuestra frecuencia se determinará por la cantidad de veces que se repite '**m**' (macho) y '**f**' (hembra) en nuestro muestreo (tabla 1).

Tabla 1: Cantidad de zarigüeyas por sexo.

Sexo	Cantidad
Macho	61
Hembra	43

Ahora calculamos la frecuencia porcentual de ambas clases:

$$Frecuencia\,Porcentual\,de\,Clase_{Macho} = \left(\frac{61}{104}\right) * 100 = 58.65\,\%$$

$$Frecuencia\,Porcentual\,de\,Clase_{Hembra} = \left(\frac{43}{104}\right) * 100 = 41.35\,\%$$

Teniendo las frecuencias relativas, dividimos el circulo del gráfico con los valores obtenidos. Teniendo 360 grados el circulo, tomamos la frecuencia del macho en valor decimal (siendo 0.5865) y lo multiplicamos por los grados del circulo:

$$0.5865 * 360 = 211.14\,grados$$

Obteniendo que el área de los machos en la muestra, debe ocupar 211.14 grados del circulo. Por lo cual al realizar la multiplicación de la clase hembra, se obtendrá para este muestreo debe ocupar área dentro del circulo de 148.86 grados.

Gráfica Circular con Python

Para poder iniciar la explicación, supondremos que el lector está familiarizado con la sintaxis de Python y con el uso de Jupyter Notebook, además de tener conocimiento sobre la carga de archivos CSV en Python con Pandas.

Las frecuencias porcentuales que estaremos graficando en los gráficos circulares será la columna “sex” (sexo) de las zarigüeyas, utilizados en la explicación del cálculo de frecuencias. Utilizando el archivo “***possum.csv***”, que ubicaremos en el mismo directorio que nuestro archivo de Jupyter Notebook (diagrama_caja.ipynb). Iniciando por la importación de la librería Pandas con la siguiente instrucción:

```
import pandas as pd
```

Tras importar la libreria de pandas, a nuestra variable “**datos**”, le asignaremos el DataFrame con la información de la columna “*sex*” (sexo) que importaremos de nuestro archivo “***possum.csv***”. Si el archivo lo tenemos en otra ubicación, debemos colocar la ruta (o *path*) de nuestro archivo de datos, junto con el nombre del arhcivo CSV.

```
datos = pd.read_csv( "possum.csv"
                     ,delimiter=','
                     ,usecols=['sex']
                    )
```

Antes de realizar nuestras gráficas, realizaremos la transformación de los valores dentro de nuestra columna de datos “*sex*”, en la que cambiaremos “**m**” por “**Macho**” y “**f**” por “**Hembra**”. Para ello, indicaremos que trabajaremos en la columna “*sex*” del *DataFrame*, e implementaremos el método ***replace***, el primer parámetro a utilizar será ***to_replace***, el cual está conformado de un diccionario, la llave corresponde al valor original y la definición el valor a remplazar, el segundo parámetro a utilizar será ***implace***, pasando como valor ***True***, para indicar que deseamos preservar los cambios en la columna de nuestro *DataFrame*.

```
datos["sex"].replace(to_replace={"m": "Macho", "f": "Hembra"}, inplace=True)
```

Como lo que deseamos graficar es la frecuencia de machos y hembras en el muestreo realizado de zarigüeyas, agruparemos los valores por sexo y contaremos la cantidad de machos y hembras que tenemos en los registros y lo asignaremos a la variable **datos_grafica**.

Para la agrupación utilizaremos el método ***groupby***, pasando como parámetro la columna “sex”, donde indicaremos que de esta agrupación trabajaremos sobre la columna “sex” y como queremos contar la cantidad de machos y hembras, utilizaremos el método ***count***, a este punto lo que obtendremos será un

tipo de dato Series de Pandas. Por lo que utilizaremos el método ***reset_index*** para crear los indices de nuestro agrupamiento de datos y tener una estructura de tipo *DataFrame*, pasando en el parámetro ***name*** el valor ***count***, el cual será utilizado como nombre de la columna donde se ubicará el conteo de machos y hembras.

Para realizar las graficas en Matplotlib y Plotly estaremos utilizando la variable **datos_grafica**, en la cual ya no se realizará ninguna manipulación de los datos. A lo largo de nuestra explicación, no se manipularán los datos contenidos en el *DataFrame*, por lo cual esta misma variable será utilizada para realizar la gráfica; tanto con Matplolib, como con Plotly.

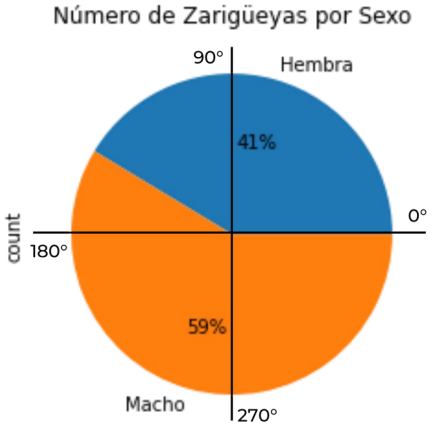
Gráfica con Pandas y Matplotlib

Teniendo instalada la librería de *Matplotlib*, importamos el paquete en nuestro archivo de *Jupyter Notebook* de la siguiente forma, colocándole como alias ***plt***:

```
import matplotlib.pyplot as plt
```

Haciendo uso de la variable **datos_grafica**, utilizaremos el método **plot** incluido dentro del tipo de estructura *DataFrame*, seleccionando la opción de **pie**. Pasaremos en el parámetro **title** un nombre apropiado y descriptivo para nuestro gráfico, con el parámetro **startangle** indicaremos en que grado del gráfico iniciará la división (imagen 3), donde nosotros indicaremos que iniciaremos en el ángulo de 0°.

Imagen 3: Distribución de ángulos para uso de *startangle*.



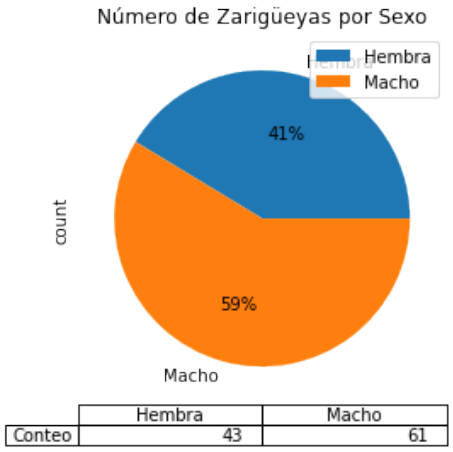
El parámetro **labels**, pasaremos en forma de lista el nombre de los valores que estamos representando en la gráfica para mostrarlo en la leyenda, siendo “Hembra” y “Macho” para este ejemplo. En el parámetro **autopct**, pasaremos el valor “*%.0f%%*”, el cual mostrara los valores enteros con porcentaje en cada sección de la gráfica circular, y para este caso haremos uso del parámetro **table** con el valor **True**, para que nos muestre una tabla con la cantidad de machos y hembras que se contaron dentro de nuestro set de datos.

Manipularemos la tabla que agregamos a la gráfica, por medio del objeto **ax**, utilizaremos el método **table**, enviando en el parámetro **cellText**, el valor del conteo contenido en la variable datos_grafica. En el parámetro **colLabels** en forma de lista, el nombre de las columnas (hembra y macho). Con el parámetro **rowLabels**, enviaremos como lista el nombre la fila, en nuestro caso le pondremos "**Conteo**".

```
ax = datos_grafica.plot.pie( y='count'
                            ,title="Número de Zarigüeyas por Sexo"
                            ,startangle = 0
                            ,labels=['Hembra','Macho']
                            ,table=True
                            ,autopct='%.0f%%'
                           )

ax.table( cellText=[datos_grafica["count"]]
         ,colLabels=['Hembra','Macho']
         ,rowLabels=["Conteo"]
        )
```

Al ejecutar el fragmento del código obtendremos el siguiente gráfico:



	Hembra	Macho
Conteo	43	61

Gráfica con Plotly

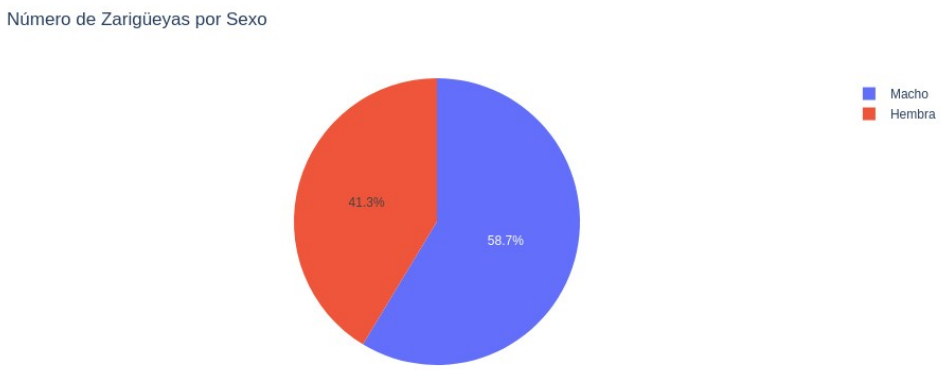
Importaremos la librería de plotly con el alias px.

```
import plotly.express as px
```

Utilizaremos el método **pie** de Plotly para realizar la gráfica, donde el primer parámetro del método corresponde a la variable que contiene el *DataFrame*. En el parámetro **values** pasaremos el nombre de la columna "*count*" que contiene los valores a graficar. Utilizaremos el parámetro **names** para enviar el nombre de la columna "*sex*" para identificar los valores que estaremos graficando. Con el parámetro **title**, le brindaremos un titulo descriptivo a nuestro diagrama y con el parámetro labels enviaremos en forma de diccionario el nombre con los que deseamos mostrar la identificación de la información que estaremos representando en el gráfico, en este caso estaremos cambiando "**sex**" por "**Sexo**" y "**count**" por "**Cantidad**".

```
px.pie(datos_grafica
        ,values='count'
        ,names='sex'
        ,title="Número de Zarigüeyas por Sexo"
        ,labels={
                  'sex':'Sexo'
                 ,'count':'Cantidad'
                 }
        )
```

Al ejecutar este fragmento de código obtendremos la siguiente imagen:



Referencias

Anderson, D., Sweeney, D., Williams, T., Camm, J. y Cochran, J. (2016). *Estadística para negocios y economía*. 12va edición. Ciudad de México, México. CENGAGE Learning. 35 – 38 pp.

Nussbaumer, C. (2015). Storytelling with data: a data visualization guide for business professionals. New Jersy, USA: WILEY. 61 – 65 pp.