Tabla de contenidos

# Visualización de Datos con Python: Técnicas y Ejemplos de Gráficos para Análisis de Datos

Elmer Achalma

Economía, Universidad Nacional de San Cristóbal de Huamanga

# Nota de Autores

Elmer Achalma Orcid ID Logo: A green circle with white letters ID <https://orcid.org/0000-0001-6996-3364>

Los autores no tienen conflictos de intereses que revelar. Agradezco a mis maestros por su orientación en el aprendizaje de la programación y visualización de datos, a mis padres por su apoyo constante durante mi formación, y a Dios por la salud y fortaleza para perseguir mis metas. Me comprometo a seguir mejorando mis habilidades en la universidad y en todos los ámbitos, siempre con respeto.

Los roles de autor se clasificaron utilizando la taxonomía de roles de colaborador (CRediT; https://credit.niso.org/) de la siguiente manera: *Elmer Achalma***:** Investigación, Programación, y Redacción

La correspondencia relativa a este artículo debe dirigirse a Elmer Achalma, Economía, Universidad Nacional de San Cristóbal de Huamanga, Email: [elmer.achalma.09@unsch.edu.pe](mailto:elmer.achalma.09@unsch.edu.pe)

# Abstract

Este documento presenta una colección de ejemplos prácticos de visualización de datos utilizando Python y la biblioteca Matplotlib. Se incluyen gráficos de líneas, barras, histogramas, circulares, de caja y combinados, cada uno acompañado de código comentado y optimizado. El objetivo es proporcionar una guía educativa para estudiantes y profesionales interesados en representar datos de manera clara y efectiva, con énfasis en buenas prácticas de diseño y presentación.

*Palabras Claves*: Visualización de datos, Python, Matplotlib, Gráficos estadísticos, Análisis de datos

# Visualización de Datos con Python: Técnicas y Ejemplos de Gráficos para Análisis de Datos

# Grafico de lineas

import numpy as np  
import matplotlib.pyplot as plt  
  
# Configurar el tamaño de la figura para mejor visualización  
plt.figure(figsize=(10, 8))  
  
# Datos de consumo de carne bovina (kg por habitante)  
consumo\_bovino = [22.1, 22.1, 23.1, 23.9, 24.6, 21.7, 23.5, 22.0, 22.5, 23.6, 21.7]  
anios\_bovino = [2001, 2002, 2003, 2004, 2005, 2006, 2007, 2008, 2009, 2010, 2011]  
  
# Datos de consumo de carne porcina (kg por habitante)  
consumo\_porcino = [17.9, 19.4, 19.1, 18.3, 19.3, 22.5, 23.5, 25.0, 24.0, 24.4, 25.6]  
anios\_porcino = [2001, 2002, 2003, 2004, 2005, 2006, 2007, 2008, 2009, 2010, 2011]  
  
# Graficar consumo de carne bovina con marcadores circulares y línea discontinua  
plt.plot(anios\_bovino, consumo\_bovino, marker='o', linestyle='--', color='red', label='Carne Bovina')  
  
# Graficar consumo de carne porcina con marcadores de diamante y línea discontinua  
plt.plot(anios\_porcino, consumo\_porcino, marker='d', linestyle='--', color='blue', label='Carne Porcina')  
  
# Etiquetas de los ejes y título con formato adecuado  
plt.xlabel('Año')  
plt.ylabel('Consumo (kg por habitante)')  
plt.title('Consumo Anual de Carne en Chile (2001-2011)')  
  
# Añadir leyenda en la esquina inferior derecha  
plt.legend(loc='lower right')  
  
# Configurar marcas en el eje x para mostrar cada año  
plt.xticks(anios\_bovino)  
  
# Añadir una cuadrícula para mejorar la legibilidad  
plt.grid(True, linestyle='--', alpha=0.7)  
  
# Ajustar el diseño para evitar recortes de etiquetas  
plt.tight\_layout()  
  
# Guardar la figura en un archivo (opcional, se puede descomentar para usar)  
# plt.savefig('consumo\_carne\_chile.png')  
  
# Mostrar el gráfico  
plt.show()

import matplotlib.pyplot as plt  
import numpy as np  
  
x = np.linspace(0, 2 \* np.pi, 200)  
y = np.sin(x)  
  
fig, ax = plt.subplots()  
ax.plot(x, y)  
plt.show()

hol

Figura 1

import numpy as np  
import matplotlib.pyplot as plt  
  
r = np.arange(0, 2, 0.01)  
theta = 2 \* np.pi \* r  
fig, ax = plt.subplots(  
 subplot\_kw = {'projection': 'polar'}   
)  
ax.plot(theta, r)  
ax.set\_rticks([0.5, 1, 1.5, 2])  
ax.grid(True)  
plt.show()

# Gráfico de barras

## horizontal

import matplotlib.pyplot as plt  
  
# Configurar el tamaño de la figura para una mejor visualización  
plt.figure(figsize=(10, 8))  
  
# Tipos de legumbres y sus respectivos consumos en kg por habitante en 2001  
tipos\_legumbres = ["Poroto", "Lenteja", "Garbanzo", "Arveja"]  
consumo\_legumbres = [2.1, 1.0, 0.3, 0.5]  
  
# Crear un gráfico de barras con color personalizado y borde  
plt.bar(tipos\_legumbres, consumo\_legumbres, color='green', edgecolor='black', alpha=0.7)  
  
# Etiquetas de los ejes y título con formato adecuado  
plt.xlabel('Tipos de Legumbres')  
plt.ylabel('Consumo (kg por habitante)')  
plt.title('Consumo de Legumbres en Chile (2001)')  
  
# Añadir una cuadrícula en el eje y para facilitar la lectura  
plt.grid(True, axis='y', linestyle='--', alpha=0.7)  
  
# Ajustar el diseño para evitar recortes de etiquetas  
plt.tight\_layout()  
  
# Guardar la figura en un archivo (opcional, descomentar para usar)  
# plt.savefig('consumo\_legumbres\_2001.png')  
  
# Mostrar el gráfico  
plt.show()

## vertical

plt.figure(figsize=[10,8])  
#Consumo de legumbres en el 2001  
legumbres=["Poroto","Lenteja","Garbanzo","Arveja"]  
consumo=[2.1, 1.0, 0.3, 0.5]  
  
plt.barh(legumbres,consumo)  
  
plt.ylabel("Tipos de legumbre")  
plt.xlabel("Consumo (kg/hab)")  
plt.title("Consumo de Legumbres en el 2001")  
plt.show()

# Histograma

import matplotlib.pyplot as plt  
  
# Configurar el tamaño de la figura para una mejor visualización  
plt.figure(figsize=(10, 8))  
  
# Datos de niveles de glucosa (mg/dl)  
niveles\_glucosa = [52, 54, 55, 57, 56, 57, 54, 59, 60, 57, 52, 62, 64, 68, 64, 72, 77, 80,   
 76, 79, 81, 85, 88, 84, 89, 92, 85, 92, 94, 93, 92, 99, 100, 105, 106, 107, 109]  
  
# Bordes de los intervalos para el histograma (bins)  
intervalos = [50, 60, 70, 80, 90, 100, 110, 120, 130]  
  
# Crear el histograma con color personalizado y bordes  
plt.hist(niveles\_glucosa, bins=intervalos, color='skyblue', edgecolor='black', alpha=0.7)  
  
# Etiquetas de los ejes y título con formato adecuado  
plt.xlabel('Nivel de Glucosa (mg/dl)')  
plt.ylabel('Número de Pacientes')  
plt.title('Distribución de Niveles de Glucosa en Pacientes')  
  
# Añadir una cuadrícula en el eje y para facilitar la lectura  
plt.grid(True, axis='y', linestyle='--', alpha=0.7)  
  
# Ajustar el diseño para evitar recortes de etiquetas  
plt.tight\_layout()  
  
# Guardar la figura en un archivo (opcional, descomentar para usar)  
# plt.savefig('distribucion\_glucosa.png')  
  
# Mostrar el gráfico  
plt.show()

# Grafico circular

import matplotlib.pyplot as plt  
  
# Configurar el tamaño de la figura para una mejor visualización  
plt.figure(figsize=(10, 8))  
  
# Datos de marcas de autos y sus ventas (en alguna unidad, ej. miles de unidades)  
marcas\_autos = ["Kia", "Toyota", "Nissan", "Suzuki", "Audi"]  
ventas = [10.5, 15.3, 14.2, 16.1, 9.8]  
# Resaltar la primera marca (Kia) ligeramente  
resaltar = [0.1, 0, 0, 0, 0]  
  
# Aplicar un estilo visual predefinido (ggplot)  
plt.style.use("ggplot")  
  
# Crear el gráfico de pastel  
plt.pie(x=ventas, explode=resaltar, labels=marcas\_autos, autopct="%.2f%%", shadow=True, startangle=20)  
  
# Asegurar que el gráfico sea circular  
plt.axis=("equal")  
  
# Añadir un título descriptivo  
plt.title("Distribución de Ventas de Autos en EE.UU.")  
  
# Añadir una leyenda en la esquina superior izquierda  
plt.legend(marcas\_autos, loc="upper left")  
  
# Ajustar el diseño para evitar recortes  
plt.tight\_layout()  
  
# Guardar la figura en un archivo (opcional, descomentar para usar)  
# plt.savefig('ventas\_autos\_eeuu.png')  
  
# Mostrar el gráfico  
plt.show()

# Grafico de Donut

import matplotlib.pyplot as plt  
  
# Configurar el tamaño de la figura para una mejor visualización  
plt.figure(figsize=(10, 8))  
  
# Datos de marcas de autos y sus ventas (en alguna unidad, ej. miles de unidades)  
marcas\_autos = ["Kia", "Toyota", "Nissan", "Suzuki", "Audi"]  
ventas = [10.5, 15.3, 14.2, 16.1, 9.8]  
# Resaltar la primera marca (Kia) ligeramente  
resaltar = [0.1, 0, 0, 0, 0]  
  
# Aplicar un estilo visual predefinido (ggplot)  
plt.style.use("ggplot")  
  
# Crear el gráfico de pastel (donut chart)  
plt.pie(ventas, explode=resaltar, labels=marcas\_autos, autopct="%.2f%%", shadow=True, startangle=20)  
  
# Asegurar que el gráfico sea circular  
plt.axis=("equal")  
  
# Añadir un título descriptivo  
plt.title("Distribución de Ventas de Autos en EE.UU.")  
  
# Añadir una leyenda en la esquina superior izquierda  
plt.legend(marcas\_autos, loc="upper left")  
  
# Añadir un círculo central para crear el efecto de "donut chart"  
circulo\_central = plt.Circle(xy=(0, 0), radius=0.75, facecolor="white")  
plt.gca().add\_artist(circulo\_central)  
  
# Ajustar el diseño para evitar recortes  
plt.tight\_layout()  
  
# Guardar la figura en un archivo (opcional, descomentar para usar)  
# plt.savefig('ventas\_autos\_eeuu\_donut.png')  
  
# Mostrar el gráfico  
plt.show()

# Grafico de cajas

import matplotlib.pyplot as plt  
  
# Configurar el tamaño de la figura para una mejor visualización  
plt.figure(figsize=(10, 8))  
  
# Datos de las edades de los alumnos  
edades\_alumnos = [12, 13, 12, 17, 16, 15, 14, 15, 15, 16, 14, 12, 15, 16, 14, 12, 16, 17]  
  
# Crear el diagrama de caja con un estilo personalizado  
plt.boxplot(edades\_alumnos, vert=True, patch\_artist=True,   
 boxprops=dict(facecolor='lightblue', edgecolor='black'),  
 medianprops=dict(color='red'), whiskerprops=dict(color='black'),  
 capprops=dict(color='black'), flierprops=dict(marker='o', color='orange', markersize=8))  
  
# Etiquetas de los ejes y título con formato adecuado  
plt.ylabel('Edad (años)')  
plt.title('Distribución de Edades de Alumnos de Secundaria')  
  
# Añadir una cuadrícula en el eje y para facilitar la lectura  
plt.grid(True, axis='y', linestyle='--', alpha=0.7)  
  
# Configurar las etiquetas del eje x (opcional, ya que solo hay un grupo)  
plt.xticks([1], ['Alumnos'])  
  
# Ajustar el diseño para evitar recortes  
plt.tight\_layout()  
  
# Guardar la figura en un archivo (opcional, descomentar para usar)  
# plt.savefig('edades\_alumnos\_secundaria.png')  
  
# Mostrar el gráfico  
plt.show()

# Grafico de barras combinadas

import matplotlib.pyplot as plt  
  
# Configurar el tamaño de la figura para una mejor visualización  
plt.figure(figsize=(10, 8))  
  
# Datos de ventas de autos por día (en unidades)  
ventas\_toyota = [10, 15, 19, 14, 9]  
ventas\_audi = [15, 25, 27, 24, 28]  
dias = [0, 1, 2, 3, 4] # Posiciones para los días  
ancho\_barras = 0.5 # Grosor de las barras  
  
# Crear el gráfico de barras apiladas  
barras\_toyota = plt.bar(dias, ventas\_toyota, ancho\_barras, label='Toyota', color='skyblue', edgecolor='black')  
barras\_audi = plt.bar(dias, ventas\_audi, ancho\_barras, bottom=ventas\_toyota, label='Audi', color='salmon', edgecolor='black')  
  
# Configurar las etiquetas del eje x con los días  
plt.xticks(dias, ['Día 1', 'Día 2', 'Día 3', 'Día 4', 'Día 5'])  
  
# Etiquetas de los ejes y título con formato adecuado  
plt.xlabel('Días')  
plt.ylabel('Unidades Vendidas')  
plt.title('Ventas de Autos por Día (Toyota vs. Audi)')  
  
# Configurar las marcas del eje y para mayor claridad  
plt.yticks([0, 5, 10, 15, 20, 25, 30, 35, 40, 45, 50])  
  
# Añadir una cuadrícula en el eje y para facilitar la lectura  
plt.grid(True, axis='y', linestyle='--', alpha=0.7)  
  
# Añadir una leyenda para identificar las marcas  
plt.legend()  
  
# Ajustar el diseño para evitar recortes  
plt.tight\_layout()  
  
# Guardar la figura en un archivo (opcional, descomentar para usar)  
# plt.savefig('ventas\_autos\_toyota\_audi.png')  
  
# Mostrar el gráfico  
plt.show()

# Graficos combinados

import matplotlib.pyplot as plt  
  
# Configurar el tamaño de la figura para una mejor visualización  
plt.figure(figsize=(10, 8))  
  
# Datos de vacunación proyectada (en número de pacientes)  
vacunacion\_proyectada = [250, 120, 270, 560, 450, 280, 550]  
# Datos de vacunación real (en número de pacientes)  
vacunacion\_real = [150, 300, 120, 550, 500, 240, 600]  
meses = ["Enero", "Febrero", "Marzo", "Abril", "Mayo", "Junio", "Julio"]  
  
# Graficar la vacunación proyectada como una línea con marcadores  
plt.plot(meses, vacunacion\_proyectada, marker='d', linestyle='--', color='red', label='Vacunación Proyectada')  
  
# Graficar la vacunación real como barras  
plt.bar(meses, vacunacion\_real, color='skyblue', edgecolor='black', alpha=0.7, label='Vacunación Real')  
  
# Etiquetas de los ejes y título con formato adecuado  
plt.xlabel('Meses')  
plt.ylabel('Número de Pacientes Vacunados')  
plt.title('Vacunación Real vs. Proyectada (Enero - Julio)')  
  
# Añadir una leyenda  
plt.legend()  
  
# Añadir una cuadrícula en el eje y para facilitar la lectura  
plt.grid(True, axis='y', linestyle='--', alpha=0.7)  
  
# Ajustar el diseño para evitar recortes  
plt.tight\_layout()  
  
# Guardar la figura en un archivo (opcional, descomentar para usar)  
# plt.savefig('vacunacion\_real\_vs\_proyectada.png')  
  
# Mostrar el gráfico  
plt.show()