REPRESENTACIÓN GRÁFICA DE DATOS ESTADÍSTICOS EN PYTHON UTILIZANDO LA BIBLIOTECA MATPLOTLIB

Ing. Esther Viridiana Vázquez Carmona
Instituto Politécnico Nacional
Centro de Innovación y Desarrollo Tecnológico en Cómputo
evazquezc1801@alumno.ipn.mx
Ing. Rodrigo Vázquez López
Instituto Politécnico Nacional
Centro de Innovación y Desarrollo Tecnológico en Cómputo
rvazquezl1800@alumno.ipn.mx
Dr. Juan Carlos Herrera Lozada
Instituto Politécnico Nacional
Centro de Innovación y Desarrollo Tecnológico en Cómputo
jlozada@ipn.mx

Resumen

Hoy en día, en el análisis de la información se estila representar los datos en gráficos que tienen por objetivo servir de apoyo visual para transmitir de una manera sencilla y comprensible dicha información. Este documento tiene como finalidad exponer diferentes formas de representación gráfica tales como: diagramas de barras, histogramas, gráficos de sectores, gráfico de líneas y gráficas 3D, a través del lenguaje Python, leyendo los datos desde un archivo de valores separados por comas (CSV) que contiene información de sensores de humedad, presión y temperatura además de formar parte de un trabajo previo (Vázquez Carmona et al., 2019). Igualmente, se muestra el procedimiento para realizar la lectura de este tipo de archivos y posteriormente realizar las gráficas antes mencionadas. Adicionalmente se habla acerca de las diferentes bibliotecas que hay en Python para visualizar información como: Seaborn, Bokeh, Pygal y Plotly, concentrando el trabajo en la biblioteca Matplotlib que además de funcionar en scripts de Python, funciona también en aplicaciones web.

I. Introducción

Un gráfico es la representación visual para transmitir información de una serie de datos estadísticos que facilitan su comparación para deducir ciertos comportamientos y posteriormente tomar decisiones en base a lo obtenido. Actualmente existen diferentes herramientas para desarrollar este tipo de representaciones, sin embargo, el trabajo se centra en realizar el procedimiento a través del leguaje Python, ya que es un lenguaje fácil de aprender en comparación con otros, además de que el script puede ser reproducido en cualquier plataforma.

1.1. Lenguaje Python

Python es un lenguaje de programación que se implementó en 1989, es un lenguaje interpretado de alto nivel, es decir convierte el código a lenguaje máquina a medida que se ejecuta; cabe destacar que el lenguaje máquina es un conjunto de instrucciones que únicamente entiende el procesador. Python

permite múltiples paradigmas y estilos de programación tales como: orientado a objetos (herencia y polimorfismo), imperativo (con sentencias de bucle) y funcional (con módulos y funciones) (Lenguajes de programación, 2019), además de ser una plataforma libre y de código abierto.

1.2 Bibliotecas para gráficos en Python

Python se ha convertido en uno de los lenguajes más populares de hoy en día, ya que una de sus características es su flexibilidad. Cuenta con diferentes bibliotecas, las cuales son un conjunto de funciones que proporcionan un servicio o tarea específica al ser invocadas. Las bibliotecas encargadas de la visualización de datos son uno de los tantos servicios que brindan dichas bibliotecas. Existen diferentes tipos de bibliotecas para la creación de gráficos en Python, como Seaborn, Bokeh, Pygal, Plotly y Matplotlib (BBVA API_Market, 2015).

1.3 Tipos de representaciones gráficas

Cuando se requiere representar la información a través de una correlación estadística se hace mediante líneas, vectores o superficies, que están plasmados en gráficos, estos pueden ser de gráficas de barras, histogramas, gráficos de sectores, gráfico de líneas y gráficas 3D, entre otros. A continuación, se presenta a detalle la descripción de cada una de ellas.

Gráficas de barras: Estas gráficas representan a la variable a través de barras, de modo que la altura de cada una de ellas sea proporcional a la frecuencia o porcentaje de casos en cada caso, la orientación de estas puede ser horizontal o vertical (Euskadi.es, s.f., 2019).

Histogramas: Se usan para representar las frecuencias de una variable, en uno de los ejes se posicionan los intervalos y en el otro eje las frecuencias. Cabe destacar que no existen separación entre dichas barras (Euskadi.es, s.f., 2019).

Gráficas de sectores: Este tipo de gráficas circulares o de tarta, dividen un círculo en porciones proporcionales según el valor de las frecuencias relativas (Euskadi.es, s.f., 2019).

Gráficas de líneas: Un gráfico de líneas es una representación gráfica en un eje cartesiano de la relación que existe entre dos variables reflejando con claridad los cambios producidos (Fisterra.com, 2014).

Gráficas 3D: Se llaman gráficos 3D a todos los objetos que se pueden dibujar en un espacio R3: puntos, segmentos, curvas, superficies y varios cuerpos formados por caras poligonales (Gráficos en tercera dimensión (3D) con Python y Matplotlib", s.f.).

2. Desarrollo

Para la adquisición de datos de las variables de humedad, presión barométrica y temperatura se desarrolló previamente un prototipo de un sistema de monitoreo de variables climáticas que funciona a través de los sensores DHT- 22 y BMP-180, logrando mostrar los datos en el monitor serie proporcionado por el IDE de Arduino, estos datos a su vez se encuentran almacenados en un archivo de valores separados por comas (CSV) para hacer uso de ellos a continuación.

2.1 Diseño del software para la representación de datos en Python

Como primera etapa se realizó la codificación que se encarga de hacer la lectura de los datos a partir del archivo CSV que contiene los valores de cada uno de los sensores, el código para la lectura de estos se describe en la figura 1.

```
import pandas as pd #Biblioteca que contiene una estructura de datos tabular integrada por columnas y filas
import matplotlib.pyplot as plt #Biblioteca de trazado 2D para Python
from mpl_toolkits.mplot3d import axes3d  # Módulo para trazar gráficos en 3D

archivo='Datos_sensores.csv' # Asignar a una variable el nombre de archivo y extensión
datos=pd.read_csv(archivo) # Leer el archivo que tiene como parámetro la variable que contiene nombre + extension
df=pd.DataFrame(datos) # Llamar al constructor DataFrame en la lista

Columna1=df.loc[:,"Humedad"] # Asignar a una variable independiente a cada columna utilizando
Columna2=df.loc[:,"Temperatura"] # una etiqueta como referencia que indique el tipo de valor numérico.
Columna3=df.loc[:,"Presion"]# Cabe destacar que estas etiquetas deben estar también en el archivo .csv como encabezado
tam=Columna1.size #Tamaño de todas las series

muestra=[] # Arreglo que contiene el numero de muestras
humedad=[] # Arreglo que contiene los datos de la serie humedad
temperatura=[] #Arreglo que contiene los datos de la serie temperatura
presion=[] # #Arreglo que contiene los datos de la serie presión
contador=0 # Variable que sirve como el contador para recorrer cada serie
i=0 # Iterador para el ciclo for

for i in range (tam):
    muestra.append(contador) #Llenar un arreglo con los datos que corresponden a el numero de muestras
    humedad.append(Columna1) #Llenar un arreglo con los datos que corresponden a la serie humedad
    temperatura.append(Columna2) #Llenar un arreglo con los datos que corresponden a la serie temperatura
    presion.append(Columna3) #Llenar un arreglo con los datos que corresponden a la serie presion
    contador=contador+1 #Contador para recorrer cada elemento en las series

print(datos) #Imprimir el contenido del archivo
```

Figura 1. Algoritmo codificado en Python para la lectura de datos.

Como segunda etapa se realizó la codificación que se encarga de la visualización de cada gráfico. En la figura 2 se puede observar el código que corresponde a un diagrama de barras haciendo uso de los datos de los sensores.

```
plt.grid(True) #Activar malla de fondo

# Tipo de grafico, en este caso es una grafica de barras con parámetros que incluyen el color y la serie a graficar, puede
#contener otros parámetros.
plt.bar(muestra, Columna1, color='yellowgreen')
plt.legend(('Humedad',), loc = 'upper right') # Etiqueta que representa el contenido del gráfico
plt.ylabel("% Humedad Relativa", fontsize = 15) # Etiqueta que representa el eje de las y
plt.xlabel("Muestras", fontsize = 15) # Etiqueta que representa el eje de las x
plt.savefig("Humedad.jpg") # Guardar gráfico creado
plt.show() #Mostrar gráfico
```

Figura 2. Algoritmo codificado en Python para gráfica de barras.

En la figura 3 se puede observar el código que corresponde a un histograma.

```
plt.grid(True)
# Tipo de grafico, en este caso es una histograma con parámetros que incluyen
#el arreglo en el que se encuentra el dato a graficar y el número de particiones
plt.hist(temperatura, bins=30)
plt.legend(('Temperatura',), loc = 'upper right')
plt.ylabel("Temperatura °C", fontsize = 15)
plt.xlabel("Muestras", fontsize = 15)
plt.savefig("Temperatura.jpg")
plt.show()
```

Figura 3. Algoritmo codificado en Python para histograma.

En la figura 4 se puede observar el código que corresponde a un gráfico de sectores.

```
minimo_t=min(Columna2) # Obtiene el minimo de la serie temperatura
maximo_t=max(Columna2) # Obtiene el maximo de la serie temperatura

#Nótese que este paso no es necesario, se realizó para efectos de la gráfica
```

```
labels = 'Temperatura mínima', 'Temperatura máxima' # Etiquetas que van a representar cada una de las particiones
sizes = [minimo_t, maximo_t] #Valores que corresponden a cada partición
colors = ['lightcoral', 'lightskyblue'] # Colores para cada partición
explode = (0.1, 0,)  # Separación entre cada partición

# Tipo de grafico, en este caso es un gráfico de sectores con parámetros que incluyen
#el tamaño, la separación, etiquetas y colores, número de decimales a mostrar, sombreado, posición.
plt.pie(sizes, explode=explode, labels=labels, colors=colors,
autopct='%1.1f%%', shadow=True, startangle=140)
plt.axis('equal') #Forma que tomará el gráfico
plt.savefig("TemperaturaM.jpg")
plt.show()
```

Figura 4. Algoritmo codificado en Python para un gráfico de sectores.

En la figura 5 se puede observar el código que corresponde a un gráfico de líneas

```
plt.grid(True)

# Tipo de grafico, en este caso es un gráfico de líneas con parámetros que incluyen
#las series a graficar, color que puede ser en hexadecimal y transparencia.
plt.plot(muestra,Columna3,'#cd98f1',alpha=0.6)
plt.legend(('Presión', ), loc = 'upper right')
plt.xlabel("Muestras", fontsize = 15).
plt.ylabel("Presión mb", fontsize = 15)
plt.savefig("Presion.jpg")
plt.plot()
```

Figura 5. . Algoritmo codificado en Python para un gráfico de líneas.

```
#Establecer la figura para estos ejes.Cuenta con parámetros en los que se describe
#La posición de la subtrama que está dada por un índice en una cuadrícula con filas nrows y columnas ncols.
#El índice comienza en 1 en la esquina superior izquierda y aumenta a la derecha. Dicha posición es un número entero de tres
#dígitos, donde el primer dígito es el número de filas, el segundo el número de columnas y el tercero el índice de la subtrama.
fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
muestra, temperatura, presion = axes3d.get_test_data(0.07)
ax.plot_surface(muestra,temperatura,presion,cmap='viridis', edgecolor='none')
plt.savefig("Tp.jpg")
plt.show()
```

Figura 6. Algoritmo codificado en Python para un gráfico 3D.

3. Resultados

Los resultados que se obtuvieron al ejecutar cada uno de los bloques de código se presentan a continuación. En la figura 7 se observa la gráfica de barras.

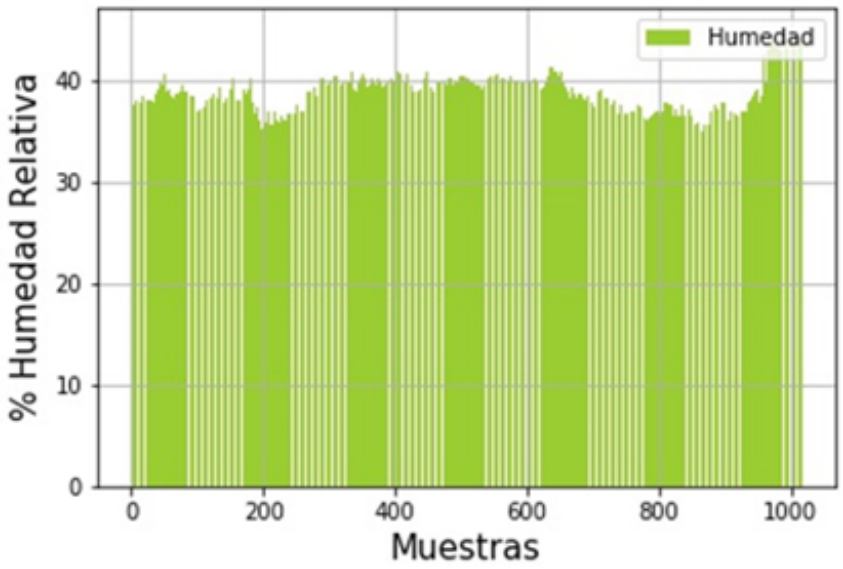

Figura 7. Gráfica de barras de la variable humedad.

En la figura 8 se observa el gráfico que corresponde a un histograma.

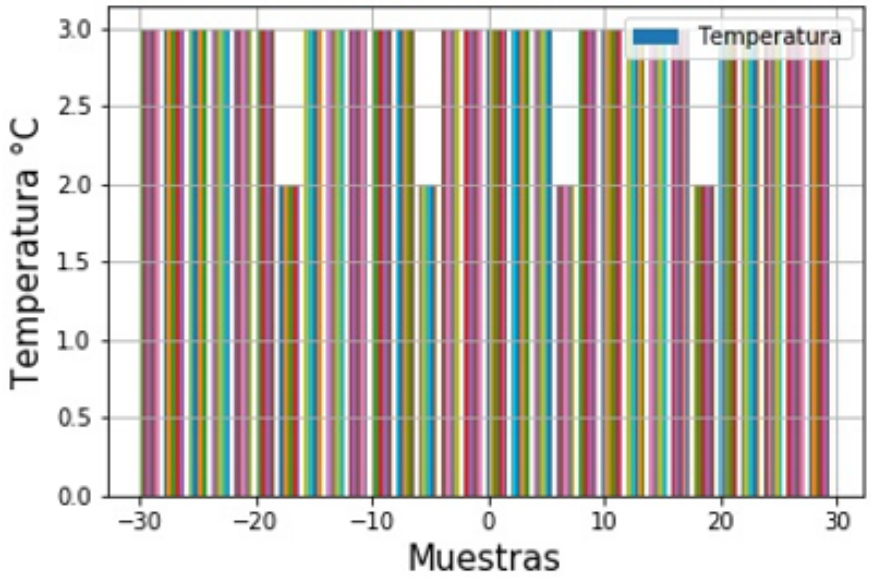

Figura 8. Histograma de la variable temperatura.

En la figura 9 se observa el gráfico que corresponde a un gráfico por sectores.

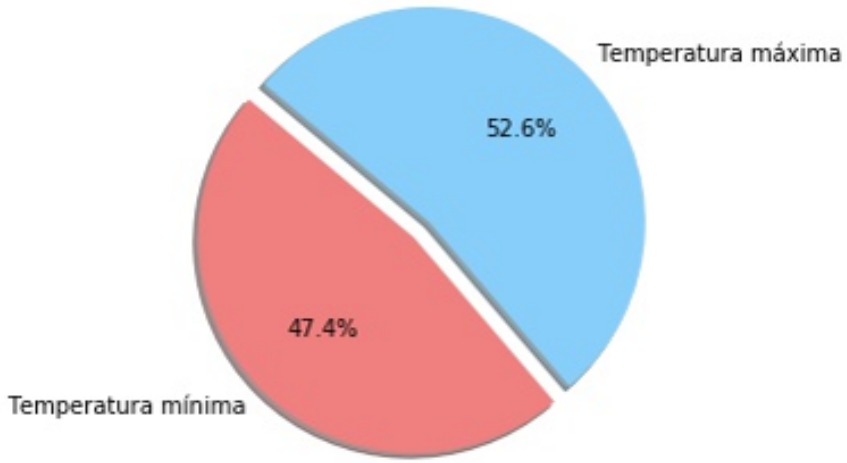

Figura 9. . Gráfico por sectores que representa el máximo y el mínimo de la variable temperatura.

En la figura 10 se observa el gráfico que corresponde a una gráfica de líneas.

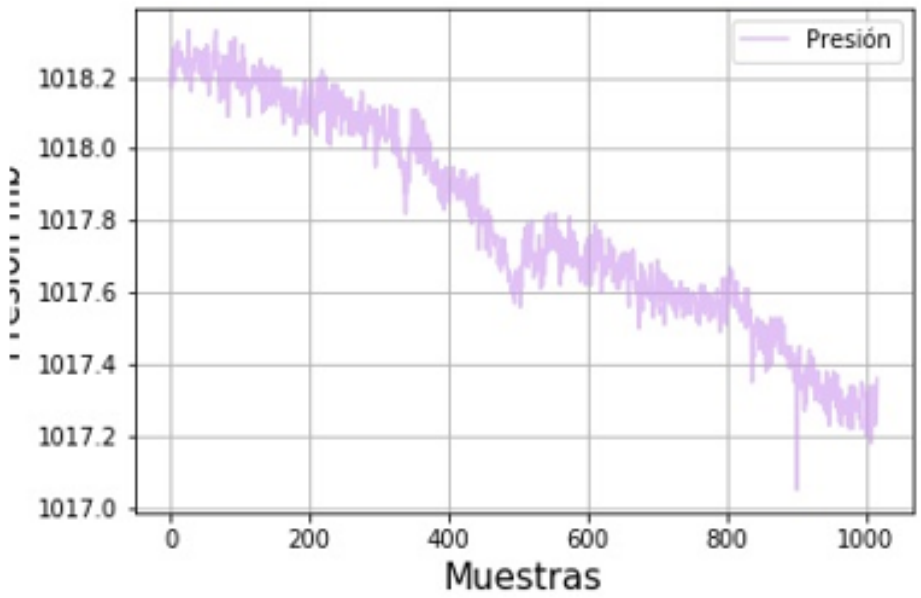

Figura 10. Gráfica de líneas que representa la variable presión.

En la figura 11 se observa el gráfico que corresponde a una gráfica 3D.

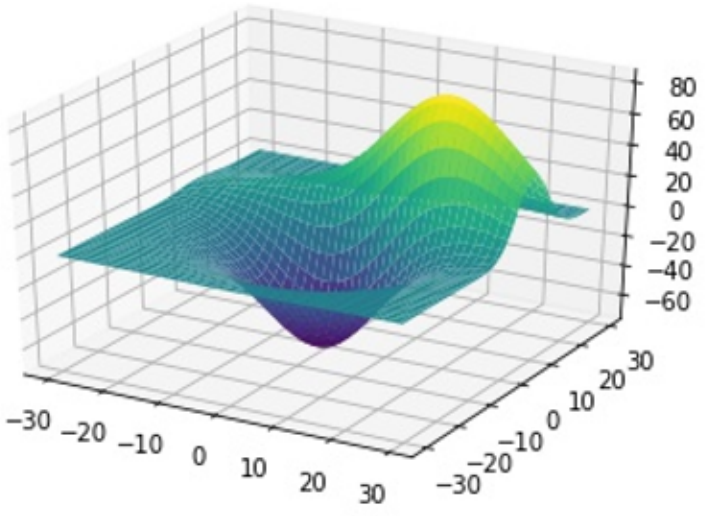

Figura . ....

Conclusiones

Existe una gran variedad de bibliotecas que permiten la creación de gráficos en Python, sin embargo, Matplotlib es un paquete completo, ya que permite la inserción de otros módulos como Pyplot, Pylab y Basemap que admiten trabajar con figuras en 3D, mapas y operaciones como en MATLAB, con la diferencia de que es libre.

Referencias

1.

2. Shaw, Z. A. (**2017**). *Aprenda a programar con Python 3.(Ed. rev).*

3. BBVA API_Market. (**2015, 17 julio**) *Cinco librerías en Python para científicos de datos: cómo visualizar información. Recuperado 4 octubre, 2019, de* https://bbvaopen4u.com/es/actualidad/cinco-librerias-en-python-para-cientificos-de-datos-como-visualizar-informacion

4. Euskadi.eus.(**s.f.-b**) *Representación gráfica de datos estadísticos. Recuperado 2 octubre, 2019, de*https://www.hiru.eus/es/matematicas/representacion-grafica-de-datos-estadisticos.

5. Fisterra.com.(**2014, 6 abril**) *Representación gráfica en el Análisis de Datos. Recuperado 4 octubre, 2019, de*https://www.fisterra.com/mbe/investiga/graficos/graficos.asp

6. Gráficos en tercera dimensión (3D) con Python y Matplotlib.(**s.f.**) *Recuperado 4 octubre, 2019, de http://www.pythondiario.com/2018/08/graficos-en-tercera-dimension-3d-con.html*

7. Lenguajes de programación. (**2019, 25 septiembre**). *Lenguaje de programación Python características y ejemplos. Recuperado 4 octubre, 2019, de* https://lenguajesdeprogramacion.net/python/

8. Matplotlib. (**2019, 26 agosto**). *Matplotlib: Python plotting — Matplotlib 3.1.1 documentation. Recuperado 4 octubre, 2019, de* https://matplotlib.org

9. Matplotlib. (**2019b, 26 agosto**) *Pyplot tutorial — Matplotlib 3.1.1 documentation. Recuperado 4 octubre, 2019, de* https://matplotlib.org/3.1.1/tutorials/introductory/pyplot.html

10. Montoro, A. F. (**2012**)
    Python 3 al descubierto. RC Libros.

11. Programación en Castellano, S.L. (**s.f.**) *Introducción a pandas. Recuperado 4 octubre, 2019, de*https://programacion.net/articulo/introduccion_a_pandas_1632

12. Queirozf.com. (**2019, 31 agosto**) *Matplotlib, Pyplot, Pylab etc: What's the difference between these and when to use each? Recuperado 4 octubre, 2019, de* http://queirozf.com/entries/matplotlib-pylab-pyplot-etc-what-s-the-different-between-these

13. Vázquez Carmona, E., Vázquez López, R., & Herrera Lozada, J.(**2019**). *ARDUINO: MONITOREO DE VARIABLES CLIMÁTICAS HACIENDO USO DE LOS SENSORES BMP-180, DHT-22 Y NEO-6M. Boletín UPIITA, 74, 1–6. Recuperado de* http://www.boletin.upiita.ipn.mx/index.php/ciencia/831-cyt-numero-74/1739-arduino-monitoreo-de-variables-climaticas-haciendo-uso-de-los-sensores-bmp-180-dht-22-y-neo-6m

Apéndice

Datos para lectura de sensores (nótese que éste archivo es más pequeño que el que se utilizó para la prueba, el archivo de prueba contiene 1017 valores, sin bordes y está almacenado con la extensión .csv)