Funciones fundamentales para Estadística y Matemáticas

Carlos Andrés Pérez Guzmán

Cargado de librerías

```
import pandas as pd
import numpy as np
import os
```

Cargado de datos

```
mainpath = r"C:\Users\DELL\OneDrive\Formación\Python\Datasets"
filename = "Datos ventas.csv"
fullpath = os.path.join(mainpath,filename)
dataset = pd.read_csv(filepath_or_buffer = fullpath, sep = ";", encoding = "utf8")
type(dataset)
```

```
pandas.core.frame.DataFrame
```

Medidas de tendencia central

Media aritmética o promedio

El método mean permite decidir tomar en cuenta o no a los valores faltantes.

```
dataset['Ventas [unid]'].mean(skipna = False)
```

```
nan
```

```
np.nanmean(dataset['Ventas [unid]'])
```

```
4.968196819681968
```

```
# np.mean devolverá nan cuando tiene un array en su argumento
# Si se alimenta a no.mean con una serie de pandas,
# entonces no tamará en cuenta a los valores vacíos
np.mean(dataset['Ventas [unid]'])
```

```
4.968196819681968
```

Mediana

```
dataset2 = pd.DataFrame({'Name' : ['Tom', 'nick', 'krish', 'jack', 'Charlie'],
                         'Age'  : [1, 2, 5, np.nan,9]})
```

```
dataset2['Age'].median(skipna = True)
```

```
3.5
```

```
np.median(dataset2["Age"])
```

nan

```
np.nanmedian(dataset2["Age"])
```

3.5

Moda

La moda es el valor o categoría más frecuente dentro de un conjunto de datos. Dado que se considera como "popular" podría representar un estándar o valor representatitvo para el dataset.

```
Puntaje = [3, 15, 30, 42, 30, 10, 10, 12]

Name = ['Mike', 'Andy', 'Nicol', 'Jordan',
               'Jordan', 'Ronald', 'Ronald', 'Andy']
```

```
# Función para calcular la moda

def mode(dataset_mode):
    # Diccionario vacío
    frequency = {}

    # El método get(keyname,value) devuelve un valor de un diccionario
    for value in dataset_mode:

        # Para el elemento 'value' se extrae su recuento y se suma 1 (a modo de conteo)
        frequency[value] = frequency.get(value, 0) + 1

    # El método values() develve los valor de un diccionario como lista
    most_frequent = max(frequency.values())

    modes = [key for key, value in frequency.items() if value == most_frequent]

    return modes
```

```
mode(Name)
```

['Andy', 'Jordan', 'Ronald']

Medidas de tendencia no central

```
# Percentiles
```

Medidas de dispersión

Rango o alcance

```
# Utilizando las fómulas de mínimo y máximo
ata = {'Name':['Tom', 'nick', 'krish', 'jack', 'Charlie'],
        'Age':[1, 2, 5, np.NaN,9]}
data = pd.DataFrame(data)

print('Valor máximo', max(data["Age"]))
print('Valor mínimo',min(data["Age"]))
rango = max(data["Age"]) - min(data["Age"])
print(f"El rango es {rango}")
```

```
Valor máximo 9.0
Valor mínimo 1.0
El rango es 8.0
```

```
# Utilizando las fómulas de numpy
# Si existen valores en blanco se debe hacer una transformación adicional
# para no afectar los valores máximos y mínimos de una variable

data = {'Name':['Tom', 'nick', 'krish', 'jack', 'Charlie'],
        'Age':[1, 2, 5, np.NaN,9]}
data = pd.DataFrame(data)
data["Age"] = data["Age"].fillna(np.mean(data["Age"]))
np.ptp(data["Age"])
```

```
8.0
```

Varianza

```
# Varianza muestral
data = {'Name':['Tom', 'nick', 'krish', 'jack', 'Charlie'],
        'Age':[1, 2, 5, np.NaN,9]}
data = pd.DataFrame(data)

print(np.var(data["Age"], ddof = 1))
print(np.nanvar(data["Age"], ddof = 1))
print(pd.DataFrame.var(data["Age"],skipna = False, ddof = 1))
```

```
12.916666666666666
12.916666666666666
nan
```

```
# Varianza poblacional
data = {'Name':['Tom', 'nick', 'krish', 'jack', 'Charlie'],
        'Age':[1, 2, 5, np.NaN,9]}
data = pd.DataFrame(data)

print(np.var(data["Age"], ddof = 0))
```

```
print(np.nanvar(data["Age"], ddof = 0))
print(pd.DataFrame.var(data["Age"],skipna = False, ddof = 0))
```

```
9.6875
9.6875
nan
```

Desviación estándar

```
data = {'Name':['Tom', 'nick', 'krish', 'jack', 'Charlie'],
        'Age':[1, 2, 5, np.NaN,9]}
data = pd.DataFrame(data)

print(np.std(data["Age"], ddof = 1))
print(np.nanstd(data["Age"], ddof = 1))
print(pd.DataFrame.std(data["Age"],skipna = False, ddof = 1))
```

```
3.593976442141304
3.593976442141304
nan
```

Coeficiente de variación

Esta medida de variación expresa la desviación estándar como un porcentaje de la media. Permite comparar distribuciones.

```
data = {'Name':['Tom', 'nick', 'krish', 'jack', 'Charlie'],
        'Age':[1, 2, 5, np.NaN,9]}
data = pd.DataFrame(data)

CV = (np.std(data["Age"], ddof = 1)) / (np.mean(data["Age"]))

print(f"El coeficiente de variación de la variable edad es de : {round(CV*100,2)} %")
```

```
El coeficiente de variación de la variable edad es de : 84.56 %
```

Otras funciones

Reemplazar valores NaN con 0

```
# Utilizando Pandas para una columna
data['Age'] = data['Age'].fillna(0)
data
```

```
# Utilizando Pandas para todo el data frame
data.fillna(0)
```

Redondear números

```
np.round(3.14159265359,2)
```

Matemáticas generales: librería *math*

El módulo "math" de Python contiene funciones que permiten realizar operaciones típicas de las matemáticas. El conjunto completo de funciones puede consultarse de https://docs.python.org/3/library/math.html

```
import math
```

Algunas constantes matemáticas

Número PI

```
math.pi
```

```
3.141592653589793
```

Número e

```
math.e
```

```
2.718281828459045
```

Thau (Equivale a dos veces Pi)

```
math.tau
```

```
6.283185307179586
```

Infinito

```
math.inf
```

```
inf
```

```
math.inf == float('inf')
```

```
True
```

Valor NaN (not a number)

```
math.nan
```

nan

Funciones logarítmicas y exponenciales

Constante e elevado a cierto exponente La potencia figura en el argumento de la función

```
math.exp(1)
```

```
2.718281828459045
```

```
2**6
```

```
64
```

Potencia

```
math.pow(5,2)
```

```
25.0
```

```
# nan está ideado para operar según reglas matemáticas
# funcion para potencias
math.pow(math.nan,0)
```

```
1.0
```

Logaritmos Si la función posee un argumento, entonces se calculará el logaritmo natural (base e)

```
math.log(10000,10)
```

```
4.0
```

```
math.log(math.e)
```

```
1.0
```

Tipos de errores

Error en la introducción de un parámetro

```
math.sqrt(-1)
```

```
ValueError: math domain error
```

Error de rango Excesivas cifras de cálculo

```
math.exp(10000)
```

```
OverflowError: math range error
```

Representación numérica

Funciones de redondeo

```
# Redondeo al alza
math.ceil(3.4523)
```

```
4
```

```
# Redondeo a la baja
math.floor(3.4523)
```

```
3
```

```
# Truncar un número real a uno entero
math.trunc(3.952)
```

```
3
```

Copiar el signo

```
math.copysign(3,-2)
```

```
-3.0
```

Valor absoluto

```
math.fabs(5)
```

```
5.0
```

```
math.fabs(-23)
```

```
23.0
```

Factorial

```
math.factorial(4)
```

```
24
```

Resto de una divsión

```
math.fmod(7,3)
```

```
1.0
```

```
math.remainder(7,3)
```

```
1.0
```

Cociente de una división exclusivamente

```
7//3
```

```
2
```

División tradicional

```
7/3
```

```
2.3333333333333335
```

Separar la parte entera y la parte decimal

```
math.modf(4.25)
```

```
(0.25, 4.0)
```

Máximo común divisor

```
math.gcd(24,36)
```

```
12
```

Comprombar si un número es finito

```
math.isfinite(2.5)
```

```
True
```

```
math.isinf(4.7)
```

```
False
```

```
math.isnan(1.3)
```

```
False
```

Comprobar la similitud entre dos números

```
math.sqrt(2)**2 == 2
```

```
False
```

```
# Comprueba el nivel de cercanía entre dos números respecto a cierta tolerancia.
math.isclose(math.sqrt(2)**2, 2, rel_tol = 1e-09)
```

```
True
```

Operadores de decisión

```
x = int(input("Escribe un número:"))
```

```
Escribe un número: 5
```

```
if x == 5:
    print("Escribiste el número ",x)
```

```
Escribiste el número  5
```

```
x = int(input("Escribe un número:"))
if x < 8 :
    print("Escribiste un número menor que 8")
elif x < 10:
    print("Escribiste un número menor que 10 y mayor o igual que 8")
else:
    print("El número es mayor o igual a 10")
```

```
Escribe un número:4
Escribiste un número menor que 8
```

```
x = int(input("Escribe un número:"))
if x > 0 and x < 10:
    print("Escribiste un número comprendido en el rango de 0 a 10")
else:
    print("Rango no válido")
```

```
Escribe un número: -5
```

```
Rango no válido
```

```
x = int(input("Escribe un número:"))
if x < 0 or x > 10:
    print("Escribiste un número que no está comprendido en el rango de 0 a 10")
```

```
Escribe un número: 11
```

```
Escribiste un número que no está comprendido en el rango de 0 a 10
```

Funciones matemáticas

Función exponencial

```
# Esta función es más precisa que utilizar e**3
math.exp(3)
```

```
20.085536923187668
```

Potencias

```
math.pow(5,3)
```

```
125.0
```

```
math.expm1(1)
```

```
1.718281828459045
```

```
# Se pierde precisión...
math.exp(1) - 1
```

```
1.718281828459045
```

```
math.exp(1e-05) - 1
```

```
1.0000050000069649e-05
```

```
math.expm1(1e-05)
```

```
1.0000050000166667e-05
```

```
math.log(12)
```

```
2.4849066497880004
```

```
math.log(12,2)
```

```
3.5849625007211565
```

```
math.log1p(1e-05)
```

```
9.99995000033333e-06
```

```
math.log2(32)
```

```
5.0
```

```
math.sqrt(64)
```

```
8.0
```

Funciones trigonométricas

```
# El argumento de la función debe ser en radianes
math.sin(180)
```

```
-0.8011526357338304
```

```
math.cos(math.pi)
```

```
-1.0
```

```
math.tan(math.pi/2)
```

```
1.633123935319537e+16
```

```
math.asin(1)
```

```
1.5707963267948966
```

```
math.acos(0)
```

```
1.5707963267948966
```

```
# Los resultados de la función están en radianes
math.atan(1)
```

```
0.7853981633974483
```

```
# Trnasformación de radianes a grados
math.degrees(0.7853981633974483)
```

```
45.0
```

```
math.cos(math.radians(60))
```

```
0.5000000000000001
```

```
# Calcular el módulo de un vector
# Los parámetros son coordenadas x,y
math.hypot(3,4)
```

```
5.0
```

```
# Hallar el ángulo del vector con el eje horizontal
math.degrees(math.atan2(4,3))
```

```
53.13010235415598
```

```
math.erf(math.pi)
```

```
0.9999911238536323
```

```
math.erfc(math.pi)
```

```
8.876146367641612e-06
```