Diplomatura Universitaria en Ciencia de Datos

https://exa.unne.edu.ar/diplomatura/

Módulo 3. Análisis Exploratorio de Datos

Filtrar datos

- Permite identificar y analizar un conjunto de datos relevantes sin tener que examinar la totalidad
  - Consiste en elegir o NO elegir ciertos datos de un dataset, según un conjunto de criterios.
- Filtro, se aplica a todo tipo de datos: números, categorías, texto y datos complejos de series temporales.

Finalidad, dado un OBJETIVO:

- Encontrar datos importantes,
- Eliminar información innecesaria

  Mejorar la calidad general de los datos y sus resultados

Filtrar datos

Función loc

- accede con los nombres de las etiquetas de las filas y las series

Función iloc

- accede con los índices.

```
# Filtrado con loc
# Seleccionar las filas si 'island' es un
valor y mostrar las columnas 'species' y
'sex'
df_loc = df.loc[df['island'] ==
'Torgersen', ['species', 'sex']]
print("Filtrado con loc:")
print(df_loc.head())

# Filtrado con iloc
# Seleccionar las primeras 10 filas y las
columnas en las posiciones 0 y 1
df_iloc = df.iloc[:10, [0, 1]]
print("\nFiltrado con iloc:")
print(df_iloc)
```

```
Filtrado con loc:
  species     sex
0  Adelie    Male
1  Adelie  Female
2  Adelie  Female
3  Adelie     NaN
4  Adelie  Female

Filtrado con iloc:
  species     island
0  Adelie  Torgersen
1  Adelie  Torgersen
2  Adelie  Torgersen
3  Adelie  Torgersen
4  Adelie  Torgersen
5  Adelie  Torgersen
6  Adelie  Torgersen
7  Adelie  Torgersen
8  Adelie  Torgersen
9  Adelie  Torgersen
```

Filtrar datos

- Siendo sex una columna y *Male* el valor buscado:

```
df.head()
# leer el archivo y cargar en el dataframe df
# contar la cantidad de filas que hay en el
dataframe
df['sex'].count()
print('filas totales ')
print(df['sex'].count())

# excluir las filas si la columna sex = Male
# crear un nuevo dataframe df_filtered

df_filtered = df[df['sex'] !='Male']



# contar la cantidad de filas del nuevo dataframe
df_filtered['sex'].count()
print('filas sexo no es male')
print(df_filtered['sex'].count())
```

```
filas totales
333

filas sexo no es male
165
```

Filtrar datos nulos

Dado un dataset, identificar los registros con valores nulos.

En Pandas métodos isnull() y notnull(), se aplican a una columna para determinar:

- Isnull(), devuelve verdadero en los registros nulos
- Notnull(), devuelve verdadero en los registros no son nulos.

Ej, determinar aquellos en los que no se conoce la masa con el siguiente ejemplo.

```
import seaborn as sb
pengui = sb.load_dataset('penguins')
pengui.head()

pengui_null = pengui[pengui.sex.isnull()]
pengui_null.head()

pengui_not_null = pengui[pengui.sex.notnull()]
pengui_not_null.head()
```

Filtrar datos en un rango

Dado un dataset, identificar los registros con valores en un rango.

En Pandas método isin(), se aplican a una columna para determinar:

```
df_pe[df_pe.bill_length_mm.isin([39.1, 40.4])]
```

	species	island	bill_length_mm	bill_depth_mm	flipper_length_mm	body_mass_g	sex
0	Adelie	Torgersen	39.1	18.7	181.0	3750.0	Male

método between()

```
df_pe[df_pe.bill_length_mm.between(38.1,40.5)]
```

35	Adelie	Dream	39.2	21.1	196.0	4150.0	Male
36	Adelie	Dream	38.8	20.0	190.0	3950.0	Male
39	Adelie	Dream	39.8	19.1	184.0	4650.0	Male
45	Adelie	Dream	39.6	18.8	190.0	4600.0	Male
50	Adelie	Biscoe	39.6	17.7	186.0	3500.0	Female
51	Adelie	Biscoe	40.1	18.9	188.0	4300.0	Male
56	Adelie	Biscoe	39.0	17.5	186.0	3550.0	Female
71	Adelie	Torgersen	39.7	18.4	190.0	3900.0	Male
72	Adelie	Torgersen	39.6	17.2	196.0	3550.0	Female
88	Adelie	Dream	38.3	19.2	189.0	3950.0	Male
89	Adelie	Dream	38.9	18.8	190.0	3600.0	Female

Filtrar datos condicionales

Los filtros con múltiples condiciones, deben incluirse entre corchetes:

df[(condicion1) & (condicion2) | (condicion3)]

Operadores lógicos & (y) y | (or).

Unir datos // Dataframes

función pandas.concat()

une distintos DataFrames, si las filas de ambos se corresponden, localizadas en el mismo orden

```
pd.concat([df_1, df_2, df_3], axis=1)
```

axis=1, unión por filas
axis=0, unión por columnas

función pandas.merge()

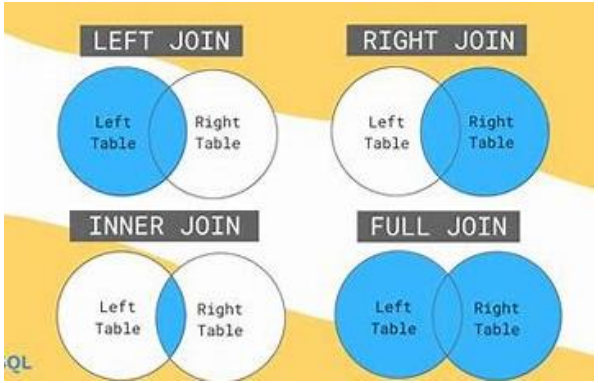
```
df_1.merge(df_2, on="id", how="left")
```

Parámetros:
on= indica nombre de la columna de unión
how= tipo de unión, similar a SQL: inner, left, right

Manejar base de datos // Dataframes

Manejo de bases de datos en Pandas...

Alternativas al uso de funciones de la librería



SQL

INNER JOIN:

- devuelve registros que tienen valores coincidentes en ambas tablas.

LEFT (OUTER) JOIN:

- devuelve todos los registros de la tabla de la izquierda y los registros coincidentes de la tabla de la derecha.
- Sin coincidencia, el resultado es NULL en el lado derecho.

RIGHT (OUTER) JOIN:

- devuelve todos los registros de la tabla de la derecha y los registros coincidentes de la tabla de la izquierda.
- Sin coincidencia, el resultado es NULL en el lado izquierdo.

FULL UNIÓN COMPLETA (EXTERNA):

- combina los resultados de las uniones externas izquierda y derecha.
- devuelve todos los registros cuando hay una coincidencia en la tabla izquierda o derecha.

SQL

SQL, o Lenguaje de Consulta Estructurado,

- Lenguaje de programación utilizado para manipular datos en Base de Datos (BD) relacionales: accede, extrae, maneja y explora,
- Pandassql, proporciona facilidades para manipular datos sin conexión a un servidor SQL
- Utiliza sintaxis SQLite

Ejemplo

```
#instalar librería
!pip install pandasql

#importar paquetes
from pandasql import sqldf
import pandas as pd
df_pe = sns.load_dataset("penguins")
print(sqldf('''SELECT distinct species FROM
df_pe ORDER BY species'''))
```

```
  species
0  Adelie
1  Chinstrap
2  Gentoo
```

```
(df_pe['species'].unique())
```

```
array(['Adelie', 'Chinstrap', 'Gentoo'], dtype=object)
```

Funciones SQL

#instalar librería

!pip install pandasql

#importar paquetes

from pandasql import sqldf

import pandas as pd

```
df_pe = sns.load_dataset("penguins")
```

 # ordenar datos

print(sqldf('''SELECT body_mass_g FROM df_pe ORDER BY body_mass_g DESC LIMIT 5'''))

```
print(sqldf('''SELECT body_mass_g FROM df_pe ORDER BY body_mass_g DESC LIMIT 5'''))
```

```
   body_mass_g
0       6300.0
1       6050.0
2       6000.0
3       6000.0
4       5950.0
```

print(df_pe['body_mass_g'].sort_values(ascending=False, ignore_index=True).head())

```
print(df_pe['body_mass_g'].sort_values(ascending=False, ignore_index=True).head())
```

```
0    6300.0
1    6050.0
2    6000.0
3    6000.0
4    5950.0
Name: body_mass_g, dtype: float64
```

Filtrar datos

filtro de datos distintos de una bd que cumple ciertas condiciones.

Ej. Las distintas especies que cumplen las condiciones (Where….)

```
sqldf('''SELECT DISTINCT species FROM df_pe WHERE sex = 'Male' AND
flipper_length_mm > 210''')
```

	species
0	Chinstrap
1	Gentoo