Clase9_Intro_DS

May 9, 2022

1 Seminario de Lenguajes - Python

1.1 Cursaa 2022

1.1.1 Introducción al análisis de datos

2 Ciencia de datos:

En pocas palabras: se refiere al análisis de datos aplicando técnicas de programación.

Sacado de Ciencia de Datos para Gente Sociable, aunque trabaja con lenguaje R la introducción es un buen punto de partida.

3 Análisis de datos

- Algo muy básico hicimos cuando procesamos datasets en formato csv.
- Ahora veremos una de las librerías más utilizadas en Python: pandas
- Vamos a graficar resultados con matplotlib

4 pandas y matplotlib

- Se instalan con pip.
- Hay muchos ejemplos con Jupyter lab / Jupyter notebook.

5 Recordemos este ejemplo

[ ]:
```
import csv
import os

archivo_netflix = os.path.join(os.getcwd(), "archivos", "netflix_titles.csv")
data_set = open(archivo_netflix, encoding='utf-8')
```

[ ]:
```
reader = csv.reader(data_set, delimiter=',')
shows_ar = filter(lambda x:  x[5] == "Argentina" and x[1] == "TV Show", reader)

for elem in shows_ar:
    print(f"{elem[2]:<40} {elem[3]}")
```

6 Ahora vamos a usar pandas

[ ]:
```
import pandas as pd

data_set = pd.read_csv(archivo_netflix, encoding='utf-8')
```

[ ]:
```
data_set
```

7 ¿De qué tipo de datos es data_set?

[ ]:
```
type(data_set)
```

- Un **dataframe** es una de las estructuras más importantes con las que trabaja pandas.
- Es una **"estructura tabular bidimensional, de tamaño mutable y potencialmente heterogénea"**.
- Un dataframe contiene:
  - datos: organizados en filas y columnas
  - labels: asociados a las filas y a las columnas
- En nuestro ejemplo: ¿cuáles son los datos? ¿Y los labels?

8 Algunas operaciones sencillas

¿Cuántas filas?

[ ]:
```
len(data_set)
```

¿Cuántas filas y columnas?

[ ]:
```
data_set.shape
```

9 ¿Cuáles son las columnas?

[ ]:
```
data_set.columns
```

10 Queremos saber los tipos de contenidos que hay en el data set de Netflix

- Deberíamos filtrar por la columna **"type"**.

[ ]:
```
data_set["type"]
#data_set.type
```

11 ¿Y si queremos que no aparezcan valores repetidos?

[ ]:
```
data_set["type"].unique()
```

12 Podemos profundizar un poquito más, y ver cuántos contenidos hay de cada tipo:

[ ]:
```
data_set["type"].value_counts()
```

13 ¿De qué tipo son las columnas?

[ ]:
```
columna = data_set["type"]
```

[ ]:
```
type(columna)
```

14 Series

- Es la otra estructura de datos básica con las que trabaja pandas.
- **Poseen un índice que se define implícitamente**. Este índice implícito indica la **posición del elemento en la Serie**.

[ ]:
```
print(columna[0])
```

- También es posible definir un índice de otro tipo.

[ ]:
```
# Top 3 de TIOBE: https://www.tiobe.com/tiobe-index/
top_3 = pd.Series(
       [13.92, 12.71, 10.82],
       index=["Python", "C", "Java"]
       )
top_3
```

[ ]:
```
top_3["Python"]
```

15 Otras formas de crear un DataFrame

Fuente de los datos: https://es.wikipedia.org/wiki/Anexo:R%C3%A9cords_del_ATP_World_Tour

[ ]:
```
datos = {
       'tenista': ['Novak Djokovic', 'Rafael Nadal', 'Roger Federer', 'Ivan␣
 ↪Lendl', 'Pete Sampras', 'John McEnroe', 'Bjorn Borg'],
       'pais': ['Serbia', 'España', 'Suiza', 'República Checa','Estados␣
 ↪Unidos', 'Estados Unidos', 'Suecia'],
       'gran_slam': [20, 21, 20, 8, 14, 7, 11],
```

```
        'master_1000': [37, 36, 28, 22, 11, 19, 15],
        'otros': [5, 1, 6, 5, 5, 3, 2]
         }
labels_filas = ["H01", "H02", "H03", "H04", "H05", "H06", "H07"]
```

[ ]:
```
df = pd.DataFrame(data=datos, index=labels_filas)
```

[ ]:
```
df
```

[ ]:
```
tenistas = df["tenista"]
tenistas
```

16 Podemos acceder a una fila del dataframe

[ ]:
```
fila = df.loc["H03"]
fila
```

¿De qué tipo es fila?

[ ]:
```
type(fila)
```

[ ]:
```
print(fila["tenista"])
```

17 Queremos ver los datos de la primera fila:

[ ]:
```
df.iloc[0]
#x = df.iloc[0]
#x["tenista"]
```

18 O a un conjunto:

[ ]:
```
# Por filas
df.iloc[2:4]
#df.loc["H03":"H06"]
```

[ ]:
```
# Por columnas
df[["tenista","master_1000"]]
```

19 O a un dato específico:

[ ]:
```
el_mejor = df.at["H03","tenista"]
el_mejor
```

20 Probar en casa...

```
[ ]: # Por filas y columnas
df.iloc[[0, 2],[2, 3]]
#df.iloc[:,[2, 3]]
#df.iloc[[x for x in range(0,len(df)) if x % 2 == 0], [0,1]]
#df.iloc[:,[True, True, False, False, True]]
#df.iloc[lambda x: x.index == "H02", [0,2]]
```

- Más info en la documentación oficial

21 Queremos saber qué tenistas ganaron más de 20 Gran Slam

```
[ ]: df[df["gran_slam"] >= 20]
```

21.1 ¿Y si queremos sólo a Roger?

```
[ ]: df[(df["gran_slam"] >= 20) & (df["pais"] == "Suiza")]
```

22 Retomemos el trabajo con nuestro dataset de Netflix

- ¿Cómo obtenemos el resultado que buscábamos?
  - "Todos los TV Shows de Argentina"

```
[ ]: # Recordemos la estructura del dataset
data_set.columns
```

23 Empecemos por filtrar los programas catalogados como "TV Show"

```
[ ]: tv_shows = data_set[data_set["type"] == "TV Show"]
tv_shows
```

24 Ahora agregamos la otra condición: que sean de Argentina

```
[ ]: tv_shows_ar = tv_shows[tv_shows["country"] == "Argentina"]
tv_shows_ar["title"]
```

25 O podemos hacerlo todo junto

```
[ ]: tv_shows_ar = data_set[(data_set["type"] == "TV Show") &  (data_set["country"]␣
      ↪== "Argentina")]
     tv_shows_ar["title"]
```

26 Podemos guardar el dataframe en archivos

```
[ ]: # En formato csv
     tv_shows_ar.to_csv("ShowsAR.csv")
```

```
[ ]: # En formato json
     tv_shows_ar.to_json("ShowsAR.json")
```

27 Algunas cosas más interesantes

- Queremos saber cuántos títulos hay por tipo y año:

```
[ ]: data_set.groupby(["type", "release_year"])["title"].count()
```

O también:

```
[ ]: data_set.groupby(["type", "release_year"]).size()
```

28 ¿Cuántos de tipo Movie?

```
[ ]: grupos = data_set.groupby(["type", "release_year"]).size()
     len(grupos["Movie"])
```

29 Y si queremos saber los 10 años con más títulos en el dataset?

29.1 Primero vamos a agrupar por año:

```
[ ]: cantidad_por_año = data_set.groupby(["release_year"]).size()
```

```
[ ]: cantidad_por_año
```

¿Cómo obtengo la cantidad de títulos de 2021?

```
[ ]: cantidad_por_año[2021]
```

30 Y, ¿cómo podemos obtener los 10 años con más títulos publicados?

[ ]:
```
cantidad_por_año.sort_values(ascending=False).head(10)
```

[ ]:
```
cantidad_por_año.sort_values().tail(10)
```

31 Y ahora graficamos

[ ]:
```
top10 = cantidad_por_año.sort_values(ascending=False).head(10)
top10
```

[ ]:
```
top10.plot(kind="bar")
```

32 Usamos matplotlib

[ ]:
```
from matplotlib import pyplot as plt
%matplotlib inline
```

33 Un poco más detallado

[ ]:
```
titulos_x_tipo = data_set.groupby(["type"]).size()
#titulos_x_tipo
```

[ ]:
```
etiquetas = ["Movies", "TV Shows"]
cantidad_peliculas = titulos_x_tipo["Movie"]
cantidad_shows = titulos_x_tipo["TV Show"]

data_dibujo = [cantidad_peliculas, cantidad_shows]
explode = (0.1, 0)

plt.pie(data_dibujo, explode=explode, labels=etiquetas, autopct='%1.2f%%',
        shadow=True, startangle=90, labeldistance= 1.1)
plt.axis('equal')
plt.legend(etiquetas)

plt.title("Cantidad de contenidos por año")
plt.show()
```

34 ¿Cómo podemos guardar el gráfico en un png?

[ ]:
```
plt.savefig('grafico.png', format="png")
```

- Revisemos: ¿se grabó bien?

[ ]:
```
plt.pie(data_dibujo, explode=explode, labels=etiquetas, autopct='%1.1f%%',
        shadow=True, startangle=90, labeldistance= 1.1)
plt.axis('equal')
plt.legend(etiquetas)

plt.title("Cantidad de contenidos por año")
plt.savefig('grafico.png', format="png")

plt.show()
```

35 Resumimos hasta acá

- pandas me permite acceder y procesar datasets.
- Las dos estructuras más importantes: Series y DataFrame.
- Si bien podemos realizar algunos gráficos con pandas, matplotlib nos permite visualizar gráficos más complejos.

36 Podemos obtener el dataframe desde tablas de HTML

Vamos a analizar los mundiales de futbol.

[ ]:
```
url = "https://es.wikipedia.org/wiki/Copa_Mundial_de_F%C3%BAtbol"
result = pd.read_html(url)
```

La función **read_html** lee las tablas HTML y retorna una list de dataframes.

Para leer documentos HTML hay varias opciones: - Con pandas, con la función read_html, - También hay otras librerías tales como BeautifulSoup

En todos los casos, se requiere instalar un parser HTML. En nuestro caso estoy usando lxml que se instala con pip.

[ ]:
```
type(result)
```

Podemos explorar los dataframes leidos. La tabla general con la información de los torneos se encuentra en **result[1]**.

36.0.1 Observemos cómo está compuesto el dataframe. ¿Cuáles son los nombres de las columnas? ¿Todas tienen valores válidos?

[ ]:
```
result[1]
```

37 ¿Cómo sabemos cuántas tablas se leyeron?

[ ]:
```
len(result)
```

38 Buscamos la tabla con el resumen de las mundiales

[ ]:
```
ediciones = result[1]
ediciones.columns
```

39 Una primera depuración

- Si analizamos los datos, vemos que hay columnas con valor **NaN**.
- Con este valor indicamos que el valor no está o es nulo.
- Vamos a eliminar estas columnas: ¿cuáles son las columnas que deberíamos eliminar?

[ ]:
```
ediciones = ediciones.drop([2, 6], axis='columns')
```

[ ]:
```
ediciones
```

40 Obervemos los nombres de las columnas

¿dónde se encuentran realmente los nombres de las columnas?

[ ]:
```
ediciones.iloc[0]
```

41 Renombremos entonces las columnas

[ ]:
```
ediciones.columns = ediciones.iloc[0]
```

[ ]:
```
ediciones
```

42 Ahora queremos eliminar esa primera fila

[ ]:
```
ediciones = ediciones[1:]
```

[ ]:
```
ediciones
```

43 Obervemos nuevamente los datos

Hay algunas filas que deberíamos eliminar también.

[ ]:
```
ediciones = ediciones.drop([4, 5, 24, 25], axis="rows")
ediciones
```

44 Ahora si, vamos a realizar algunas consultas

44.1 Desafío 1: países organizadores que salieron campeones

[ ]:
```
campeones_organizadores = ediciones[ediciones["Sede"] == ediciones["Campeón"]]
campeones_organizadores
```

44.2 Desafío 2: los 5 países que más veces salieron campeones

[ ]:
```
ediciones.groupby("Campeón").size().sort_values(ascending=False).head(5)
```

45 Tarea para el hogar...

45.1 Desafío 3: ¿cómo puedo procesar la tabla de goleadores? Quiero saber qué goleadores hicieron más de 5 goles.

46 Seguimos el próximo martes