

LABORATORIO N°1

INTRO PYTHON FOR DS

Andrés Felipe Esquivel Ruiz; Código 12967.

Nathaly Daniela Mejía Meléndez; Código 84588.

Universidad ECCI

Seminario Big Data

Elías Buitrago Bolívar

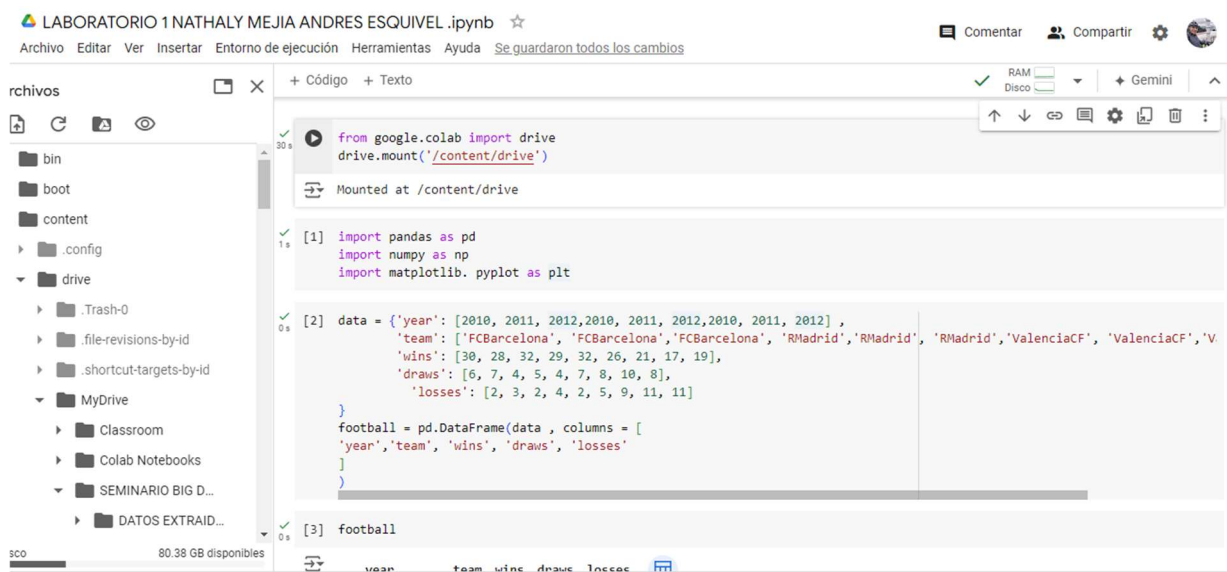
Julio de 2024

Descripción de la Actividad:

Comprender la introducción al lenguaje de programación en Python mediante el entorno Jupyter Notebook de acuerdo a la guía y la data compartida por el ingeniero Elías Buitrago Bolívar, A continuación, describimos los pasos realizados en Google Colab.

1. Para el inicio del laboratorio primeramente enlazamos el código junto con Google Drive para poder cargar y llamar el archivo de datos cargado y utilizarlo desde la nube, posteriormente definimos la librería de PANDAS as PD, la librería NUMPY as NP y MATPLOTLIB.PYPILOT as PLT.

Posteriormente describimos la estructura de datos en base a filas y columnas, esta estructura de datos está en base a equipos de futbol



```
LABORATORIO 1 NATHALY MEJIA ANDRES ESQUIVEL.ipynb
Archivo Editar Ver Insertar Entorno de ejecución Herramientas Ayuda Se guardaron todos los cambios

rchivos bin boot content .config drive .Trash-0 .file-revisions-by-id .shortcut-targets-by-id MyDrive Classroom Colab Notebooks SEMINARIO BIG D... DATOS EXTRAID...

+ Código + Texto

from google.colab import drive
drive.mount('/content/drive')
Mounted at /content/drive

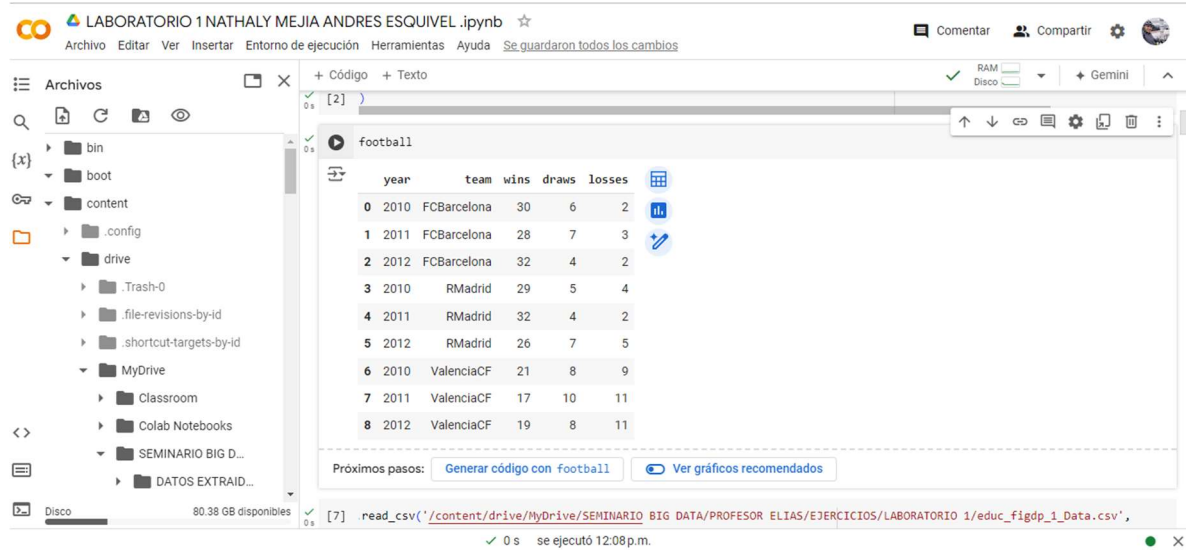
[1] import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

[2] data = {'year': [2010, 2011, 2012, 2010, 2011, 2012, 2010, 2011, 2012],
'team': ['FCBarcelona', 'FCBarcelona', 'FCBarcelona', 'RMadrid', 'RMadrid', 'RMadrid', 'ValenciaCF', 'ValenciaCF', 'V'],
'wins': [30, 28, 32, 29, 32, 26, 21, 17, 19],
'draws': [6, 7, 4, 5, 4, 7, 8, 10, 8],
'losses': [2, 3, 2, 4, 2, 5, 9, 11, 11]}

football = pd.DataFrame(data, columns = ['year', 'team', 'wins', 'draws', 'losses'])

[3] football
```

2. Para el segundo punto visualizamos la información en modo de tabla para obtenerla de manera organizada y más entendible



The screenshot shows a Jupyter Notebook interface with a file explorer on the left and a code editor on the right. The file explorer shows a directory structure with folders like 'bin', 'boot', 'content', 'drive', and 'MyDrive'. The code editor shows a cell with the following code:

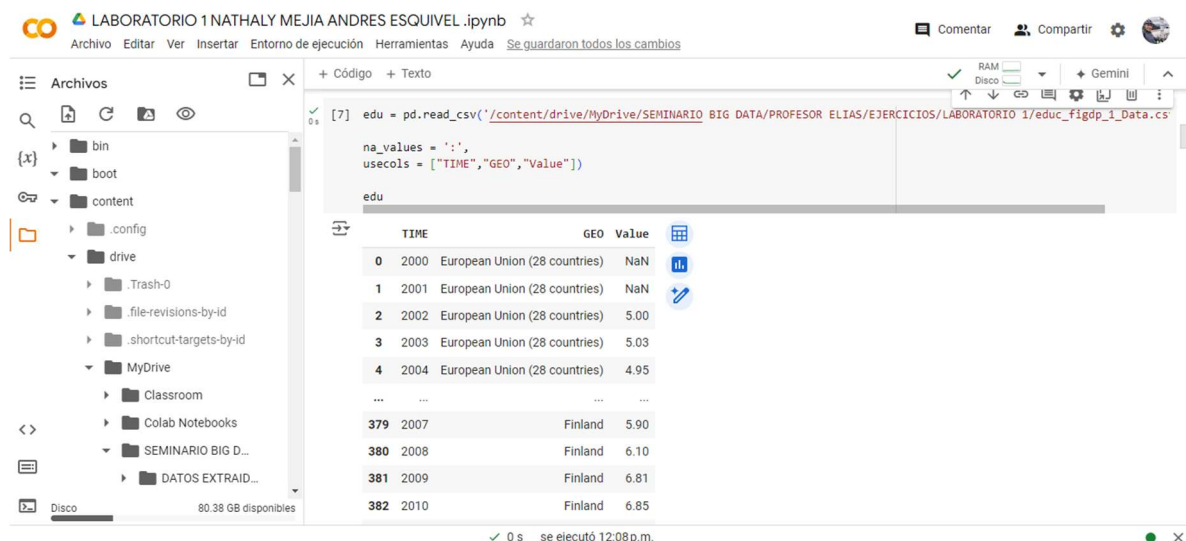
```
[2] 
```

The output of the code is a table with the following data:

| | year | team | wins | draws | losses |
|---|------|-------------|------|-------|--------|
| 0 | 2010 | FCBarcelona | 30 | 6 | 2 |
| 1 | 2011 | FCBarcelona | 28 | 7 | 3 |
| 2 | 2012 | FCBarcelona | 32 | 4 | 2 |
| 3 | 2010 | RMadrid | 29 | 5 | 4 |
| 4 | 2011 | RMadrid | 32 | 4 | 2 |
| 5 | 2012 | RMadrid | 26 | 7 | 5 |
| 6 | 2010 | ValenciaCF | 21 | 8 | 9 |
| 7 | 2011 | ValenciaCF | 17 | 10 | 11 |
| 8 | 2012 | ValenciaCF | 19 | 8 | 11 |

Below the table, there are two buttons: 'Generar código con football' and 'Ver gráficos recomendados'. The status bar at the bottom indicates '0 s se ejecutó 12:08 p.m.'.

3. En la siguiente línea de código utilizamos el comando `edu = pd.read_csv` comenzamos a leer los datos que descargamos y de esta manera poder crear una nueva ruta de dirección del archivo previamente cargado en nuestro Google drive.



The screenshot shows a Jupyter Notebook interface with a file explorer on the left and a code editor on the right. The file explorer shows a directory structure with folders like 'bin', 'boot', 'content', 'drive', and 'MyDrive'. The code editor shows a cell with the following code:

```
[7] edu = pd.read_csv('/content/drive/MyDrive/SEMINARIO BIG DATA/PROFESOR ELIAS/EJERCICIOS/LABORATORIO 1/educ_figdp_1_Data.csv',
na_values = ':',
usecols = ["TIME", "GEO", "Value"])

edu
```

The output of the code is a table with the following data:

| | TIME | GEO | Value |
|-----|------|-------------------------------|-------|
| 0 | 2000 | European Union (28 countries) | NaN |
| 1 | 2001 | European Union (28 countries) | NaN |
| 2 | 2002 | European Union (28 countries) | 5.00 |
| 3 | 2003 | European Union (28 countries) | 5.03 |
| 4 | 2004 | European Union (28 countries) | 4.95 |
| ... | ... | ... | ... |
| 379 | 2007 | Finland | 5.90 |
| 380 | 2008 | Finland | 6.10 |
| 381 | 2009 | Finland | 6.81 |
| 382 | 2010 | Finland | 6.85 |

The status bar at the bottom indicates '0 s se ejecutó 12:08 p.m.'.

4. Mediante la siguiente línea de código utilizamos el comando `edu.head()` mediante la cual realizamos la organización de modo encabezado de tal manera que se puedan organizar los datos ciusalizando unicamente las 5 primeras filas.

LABORATORIO 1 NATHALY MEJIA ANDRES ESQUIVEL .ipynb

Archivos

bin
boot
content
drive
MyDrive
Classroom
Colab Notebooks
SEMINARIO BIG D...
DATOS EXTRAID...

Disco 80.38 GB disponibles

+ Código + Texto

edu.head()

| | TIME | GEO | Value |
|---|------|-------------------------------|-------|
| 0 | 2000 | European Union (28 countries) | NaN |
| 1 | 2001 | European Union (28 countries) | NaN |
| 2 | 2002 | European Union (28 countries) | 5.00 |
| 3 | 2003 | European Union (28 countries) | 5.03 |
| 4 | 2004 | European Union (28 countries) | 4.95 |

Próximos pasos: [Generar código con edu](#) [Ver gráficos recomendados](#)

[9] edu.tail()

| | TIME | GEO | Value |
|-----|------|---------|-------|
| 379 | 2007 | Finland | 5.90 |
| 380 | 2008 | Finland | 6.10 |
| 381 | 2009 | Finland | 6.81 |
| 382 | 2010 | Finland | 6.85 |

se ejecutó 12:08 p.m.

5. Para la siguiente línea de Código utilizamos el comando `edu.describe()` para poder únicamente tener la información estadística rápida de las columnas numéricas, y el comando `edu['Value']` para seleccionar únicamente el subconjunto de datos de data frame.

LABORATORIO 1 NATHALY MEJIA ANDRES ESQUIVEL .ipynb

Archivos

bin
boot
content
drive
MyDrive
Classroom
Colab Notebooks
SEMINARIO BIG D...
DATOS EXTRAID...

Disco 80.38 GB disponibles

+ Código + Texto

edu.describe()

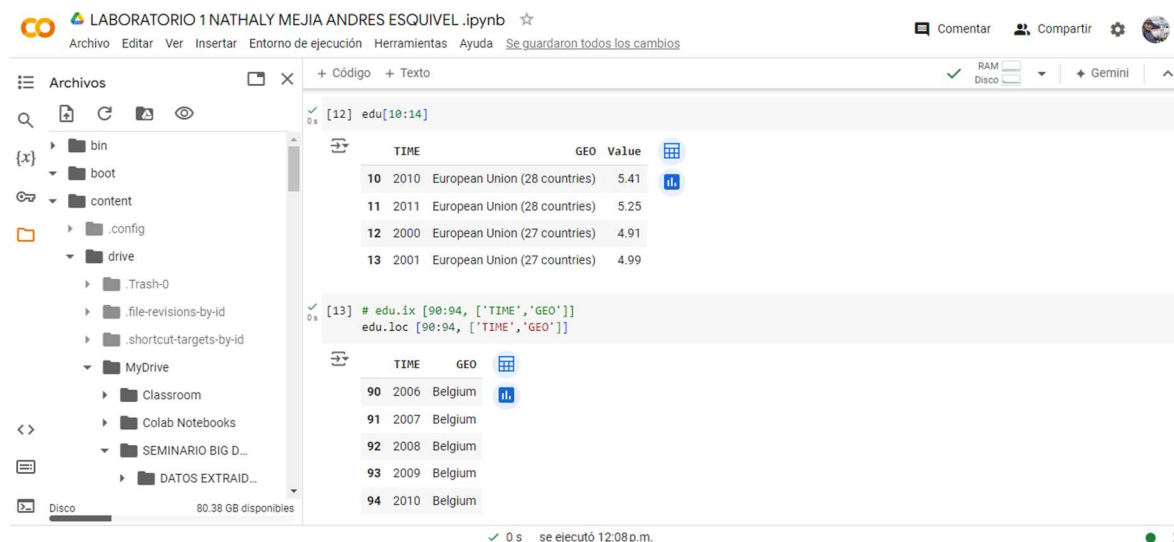
| | TIME | Value |
|-------|-------------|------------|
| count | 384.000000 | 361.000000 |
| mean | 2005.500000 | 5.203989 |
| std | 3.456556 | 1.021694 |
| min | 2000.000000 | 2.880000 |
| 25% | 2002.750000 | 4.620000 |
| 50% | 2005.500000 | 5.060000 |
| 75% | 2008.250000 | 5.660000 |
| max | 2011.000000 | 8.810000 |

[11] edu['Value']

| | Value |
|---|-------|
| 0 | NaN |
| 1 | NaN |
| 2 | 5.00 |
| 3 | 5.03 |
| 4 | 4.95 |

se ejecutó 12:08 p.m.

6. En la siguiente línea del código utilizamos un subconjunto de filas del data Frame mediante el uso del comando edu. [10:14]



LABORATORIO 1 NATHALY MEJIA ANDRES ESQUIVEL .ipynb

Archivos

bin

boot

content

.config

drive

.Trash-0

.file-revisions-by-id

.shortcut-targets-by-id

MyDrive

Classroom

Colab Notebooks

SEMINARIO BIG D...

DATOS EXTRAID...

Disco 80.38 GB disponibles

+ Código + Texto

[12] edu[10:14]

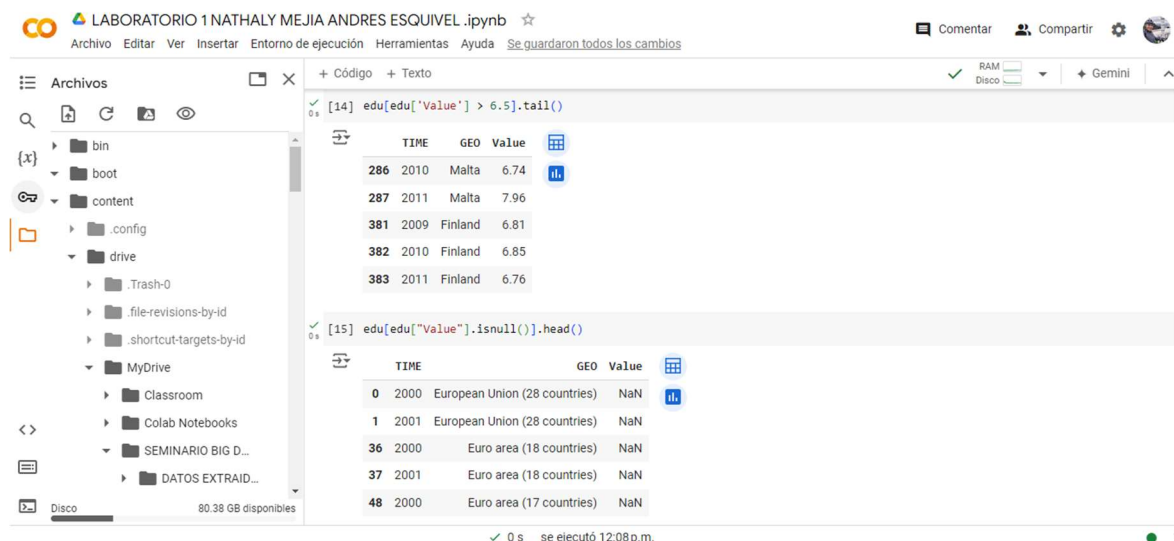
| | TIME | GEO | Value |
|----|------|-------------------------------|-------|
| 10 | 2010 | European Union (28 countries) | 5.41 |
| 11 | 2011 | European Union (28 countries) | 5.25 |
| 12 | 2000 | European Union (27 countries) | 4.91 |
| 13 | 2001 | European Union (27 countries) | 4.99 |

[13] # edu.ix [90:94, ['TIME', 'GEO']]
edu.loc [90:94, ['TIME', 'GEO']]

| | TIME | GEO |
|----|------|---------|
| 90 | 2006 | Belgium |
| 91 | 2007 | Belgium |
| 92 | 2008 | Belgium |
| 93 | 2009 | Belgium |
| 94 | 2010 | Belgium |

0 s se ejecutó 12:08 p.m.

7. En la siguiente línea de código realizamos la otra manera de seleccionar un subconjunto de datos la cual es aplicando la indexación booleana para seleccionar valores ejemplo menores a 6.5



LABORATORIO 1 NATHALY MEJIA ANDRES ESQUIVEL .ipynb

Archivos

bin

boot

content

.config

drive

.Trash-0

.file-revisions-by-id

.shortcut-targets-by-id

MyDrive

Classroom

Colab Notebooks

SEMINARIO BIG D...

DATOS EXTRAID...

Disco 80.38 GB disponibles

+ Código + Texto

[14] edu[edu['Value'] > 6.5].tail()

| | TIME | GEO | Value |
|-----|------|---------|-------|
| 286 | 2010 | Malta | 6.74 |
| 287 | 2011 | Malta | 7.96 |
| 381 | 2009 | Finland | 6.81 |
| 382 | 2010 | Finland | 6.85 |
| 383 | 2011 | Finland | 6.76 |

[15] edu[edu["Value"].isnull()].head()

| | TIME | GEO | Value |
|----|------|-------------------------------|-------|
| 0 | 2000 | European Union (28 countries) | NaN |
| 1 | 2001 | European Union (28 countries) | NaN |
| 36 | 2000 | Euro area (18 countries) | NaN |
| 37 | 2001 | Euro area (18 countries) | NaN |
| 48 | 2000 | Euro area (17 countries) | NaN |

0 s se ejecutó 12:08 p.m.

8. En la siguiente línea de código experimentamos con seleccionar los datos deseados para poder manipularlos según la guía una de las maneras mas fáciles de hacerla es operar columnas o filas

usando funciones de agregación.



The screenshot shows a Jupyter Notebook titled "LABORATORIO 1 NATHALY MEJIA ANDRES ESQUIVEL.ipynb". The left sidebar displays a file explorer with a tree view containing folders like bin, boot, content, drive, and MyDrive. The main area shows three code cells:

```
edu.max()
```

| | TIME | 2011 |
|--------|--------|------|
| GEO | Spain | |
| Value | 8.81 | |
| dtype: | object | |

```
[17] print ("Pandas max function:", edu['Value']. max ())  
      print ("Python max function:", max(edu['Value']))
```

Pandas max function: 8.81
Python max function: nan

```
[18] s = edu["Value" ]/100
```

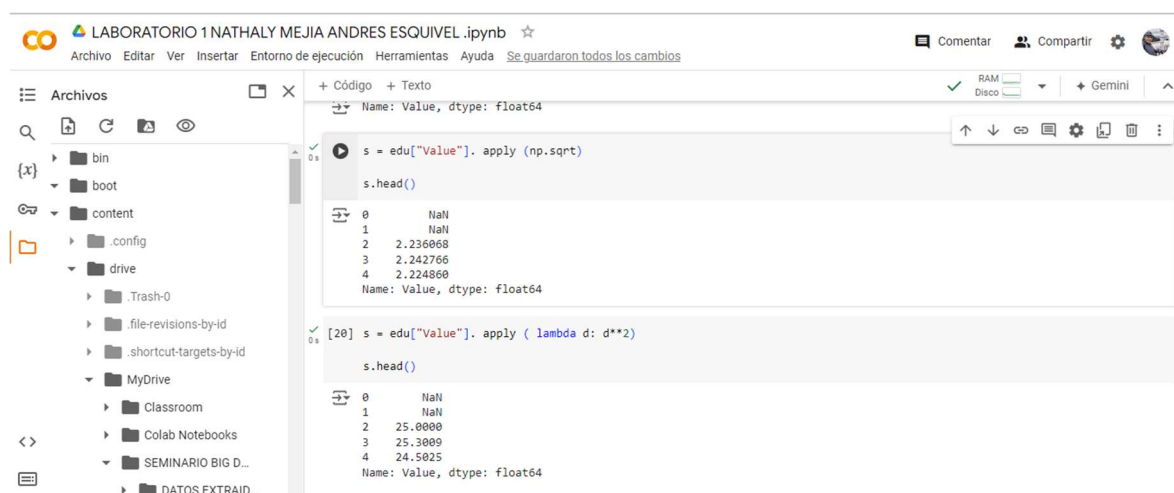
```
s.head()
```

| | 0 | NaN |
|---|--------|-----|
| 1 | NaN | |
| 2 | 0.0500 | |
| 3 | 0.0503 | |
| 4 | 0.0495 | |

Name: Value, dtype: float64

At the bottom, it says "0 s se ejecutó 12:08 p.m."

9. En la siguiente línea de código especifica que si necesitamos diseñar una función específica en línea podemos usar la función λ -función. Λ -función



The screenshot shows the same Jupyter Notebook interface. The code cells are:

```
Name: Value, dtype: float64
```

```
s = edu["Value"]. apply (np.sqrt)
```

```
s.head()
```

| | 0 | NaN |
|---|----------|-----|
| 1 | NaN | |
| 2 | 2.236068 | |
| 3 | 2.242766 | |
| 4 | 2.224860 | |

Name: Value, dtype: float64

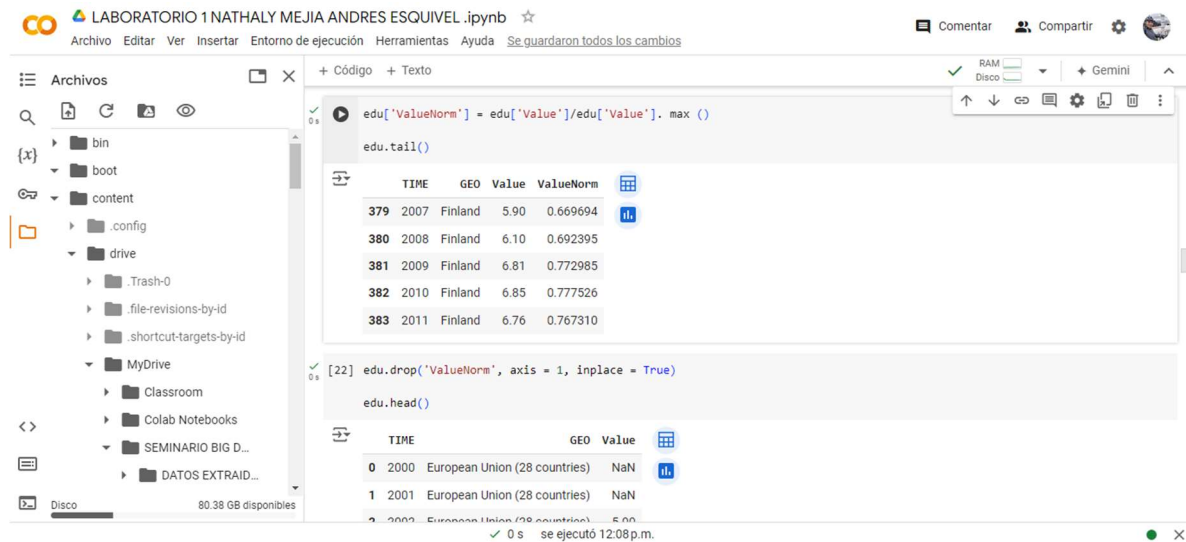
```
[20] s = edu["Value"]. apply ( lambda d: d**2)
```

```
s.head()
```

| | 0 | NaN |
|---|---------|-----|
| 1 | NaN | |
| 2 | 25.0000 | |
| 3 | 25.3009 | |
| 4 | 24.5025 | |

Name: Value, dtype: float64

10. En la siguiente línea de código realizamos una operación básica mediante el comando en la imagen, para mediante el operador (=) para la manipulación.



```
edu['ValueNorm'] = edu['Value']/edu['Value'].max ()

edu.tail()
```

| | TIME | GEO | Value | ValueNorm |
|-----|------|---------|-------|-----------|
| 379 | 2007 | Finland | 5.90 | 0.669694 |
| 380 | 2008 | Finland | 6.10 | 0.692395 |
| 381 | 2009 | Finland | 6.81 | 0.772985 |
| 382 | 2010 | Finland | 6.85 | 0.777526 |
| 383 | 2011 | Finland | 6.76 | 0.767310 |

```
[22] edu.drop('ValueNorm', axis = 1, inplace = True)

edu.head()
```

| | TIME | GEO | Value |
|---|------|-------------------------------|-------|
| 0 | 2000 | European Union (28 countries) | NaN |
| 1 | 2001 | European Union (28 countries) | NaN |
| 2 | 2002 | European Union (28 countries) | 5.00 |

11. En la siguiente línea de código utilizamos la función Drop para eliminar fila y poder colocar el eje a 0.



```
[23] # edu = edu.append({"TIME": 2000,"Value": 5.00,"GEO": 'a'},ignore_index = True)

# edu.tail()
edu.loc[edu.shape[0]]= [2000,5.00,'a']

[24] edu.tail()
```

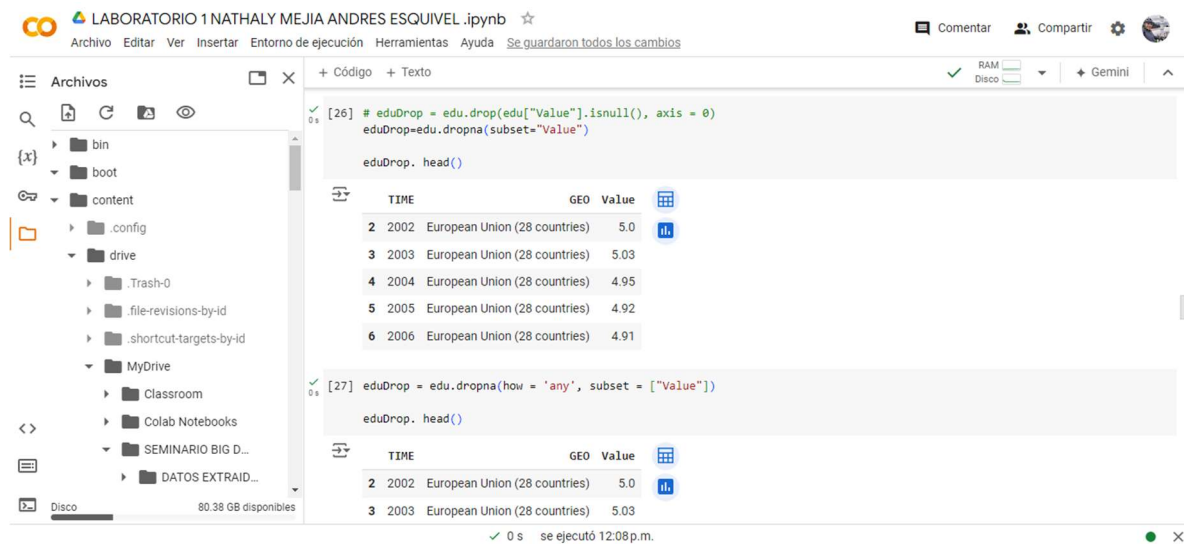
| | TIME | GEO | Value |
|-----|------|---------|-------|
| 380 | 2008 | Finland | 6.1 |
| 381 | 2009 | Finland | 6.81 |
| 382 | 2010 | Finland | 6.85 |
| 383 | 2011 | Finland | 6.76 |
| 384 | 2000 | 5.0 | a |

```
[25] edu.drop(max(edu.index), axis = 0, inplace = True)

edu.tail()
```

| | TIME | GEO | Value |
|--|------|-----|-------|
|--|------|-----|-------|

12. Para eliminar los calores NaN en lugar de usar la función drop utilizamos la siguiente función específica.



The screenshot shows a Jupyter Notebook with two code cells. The first cell uses `edu.dropna(subset="Value")` to remove rows with NaN values in the 'Value' column. The second cell uses `edu.dropna(how='any', subset=["Value"])` to remove rows with any NaN values in the specified subset. Both cells show the resulting dataset's head.

```
[26] # eduDrop = edu.drop(edu["Value"].isnull(), axis = 0)
eduDrop=edu.dropna(subset="Value")

eduDrop.head()
```

| | TIME | GEO | Value |
|---|------|-------------------------------|-------|
| 2 | 2002 | European Union (28 countries) | 5.0 |
| 3 | 2003 | European Union (28 countries) | 5.03 |
| 4 | 2004 | European Union (28 countries) | 4.95 |
| 5 | 2005 | European Union (28 countries) | 4.92 |
| 6 | 2006 | European Union (28 countries) | 4.91 |

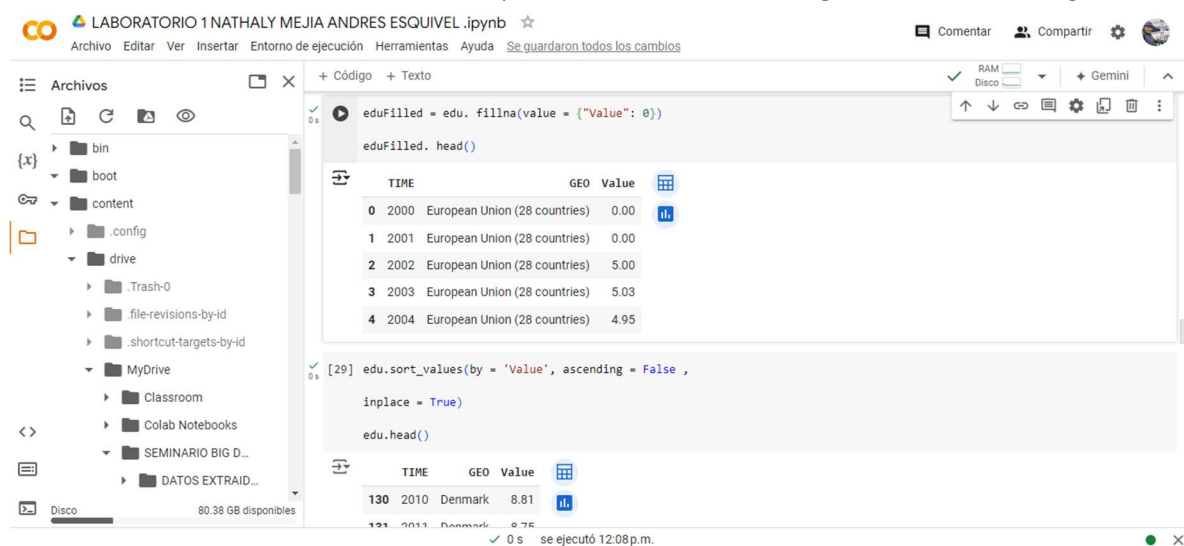
```
[27] eduDrop = edu.dropna(how = 'any', subset = ["Value"])

eduDrop.head()
```

| | TIME | GEO | Value |
|---|------|-------------------------------|-------|
| 2 | 2002 | European Union (28 countries) | 5.0 |
| 3 | 2003 | European Union (28 countries) | 5.03 |

0 s se ejecutó 12:08 p.m.

13. En la siguiente línea de código tenemos la posibilidad de en vez de eliminar los valores que contienen NaN rellenarlas con otro valor para lo cual utilizamos la siguiente línea de código.



The screenshot shows a Jupyter Notebook with two code cells. The first cell uses `eduFilled = edu.fillna(value = {"Value": 0})` to fill NaN values in the 'Value' column with 0. The second cell uses `edu.sort_values(by='Value', ascending=False, inplace=True)` to sort the dataset by the 'Value' column in descending order. Both cells show the resulting dataset's head.

```
eduFilled = edu.fillna(value = {"Value": 0})

eduFilled.head()
```

| | TIME | GEO | Value |
|---|------|-------------------------------|-------|
| 0 | 2000 | European Union (28 countries) | 0.00 |
| 1 | 2001 | European Union (28 countries) | 0.00 |
| 2 | 2002 | European Union (28 countries) | 5.00 |
| 3 | 2003 | European Union (28 countries) | 5.03 |
| 4 | 2004 | European Union (28 countries) | 4.95 |

```
[29] edu.sort_values(by = 'Value', ascending = False ,
inplace = True)

edu.head()
```

| | TIME | GEO | Value |
|-----|------|---------|-------|
| 130 | 2010 | Denmark | 8.81 |
| 131 | 2011 | Denmark | 8.75 |

0 s se ejecutó 12:08 p.m.

14. En la siguiente línea de código necesitamos inspeccionar los datos y ordenar el data frame por cualquier columna y utilizamos la función sort para poder organizar el data frame en columnas.

The screenshot shows a Jupyter Notebook with the following code and output:

```
[30] edu.sort_index( axis = 0, ascending = True , inplace = True)

edu.head()
```

| | TIME | GEO | Value |
|---|------|-------------------------------|-------|
| 0 | 2000 | European Union (28 countries) | NaN |
| 1 | 2001 | European Union (28 countries) | NaN |
| 2 | 2002 | European Union (28 countries) | 5.0 |
| 3 | 2003 | European Union (28 countries) | 5.03 |
| 4 | 2004 | European Union (28 countries) | 4.95 |

```
Próximos pasos: Generar código con edu Ver gráficos recomendados
```

```
[31] group = edu[["GEO", "Value"]].groupby('GEO').mean()

group.head()
```

| | Value |
|-----|-------|
| GEO | |

se ejecutó 12:08 p.m.

15. Ya teniendo la numeración en filas, evidenciamos que la organización no tiene mucho sentido y lo que hacemos en la línea a continuación es reorganizar los datos para los nuevos valores y columnas.

The first screenshot shows the following code and output:

```
filtered_data = edu[edu["TIME"] > 2005] # FILTRADO DE DATOS

pivedu = pd.pivot_table( filtered_data, values = 'Value',
index = ['GEO'],
columns = ['TIME'])

pivedu.head()
```

| | TIME | 2006 | 2007 | 2008 | 2009 | 2010 | 2011 |
|----------------|------|------|------|------|------|------|------|
| Austria | 5.4 | 5.33 | 5.47 | 5.98 | 5.91 | 5.8 | |
| Belgium | 5.98 | 6.0 | 6.43 | 6.57 | 6.58 | 6.55 | |
| Bulgaria | 4.04 | 3.88 | 4.44 | 4.58 | 4.1 | 3.82 | |
| Cyprus | 7.02 | 6.95 | 7.45 | 7.98 | 7.92 | 7.87 | |
| Czech Republic | 4.42 | 4.05 | 3.92 | 4.36 | 4.25 | 4.51 | |

se ejecutó 12:08 p.m.

The second screenshot shows the following code and output:

```
[33] # pivedu.loc[['Spain','Portugal'], [2006,2011]]
pivedu.loc[['Spain','Portugal'], [2006,2011]]
```

| | TIME | 2006 | 2011 |
|----------|------|------|------|
| Spain | 4.26 | 4.82 | |
| Portugal | 5.07 | 5.27 | |

```
[34] pivedu = pivedu.drop([
'Euro area (13 countries)',
'Euro area (15 countries)',
'Euro area (17 countries)',
'Euro area (18 countries)',
'Euro area (20 countries)',
'Euro area (27 countries)',
'Euro area (28 countries)'
],
axis = 0)

pivedu = pivedu.rename(index = {'Germany (until 1990 former territory of the FRG)': 'Germany'})
pivedu = pivedu.dropna()
pivedu.rank(ascending = False, method = 'first').head()
```

se ejecutó 12:08 p.m.

LABORATORIO 1 NATHALY MEJIA ANDRES ESQUIVEL .ipynb

Comentar

Compartir

Archivo

Editar

Ver

Insertar

Entorno de ejecución

Herramientas

Ayuda

Se guardaron todos los cambios

Archivos

bin

boot

content

.config

drive

.Trash-0

file-revisions-by-id

shortcut-targets-by-id

MyDrive

Código

Texto

✓

05

[35] totalSum = pivedu. sum(axis = 1)

totalSum. rank(ascending = False , method = 'dense').sort_values(). head()

GEO

Denmark 1.0

Cyprus 2.0

Finland 3.0

Malta 4.0

Belgium 5.0

dtype: float64

✓

15

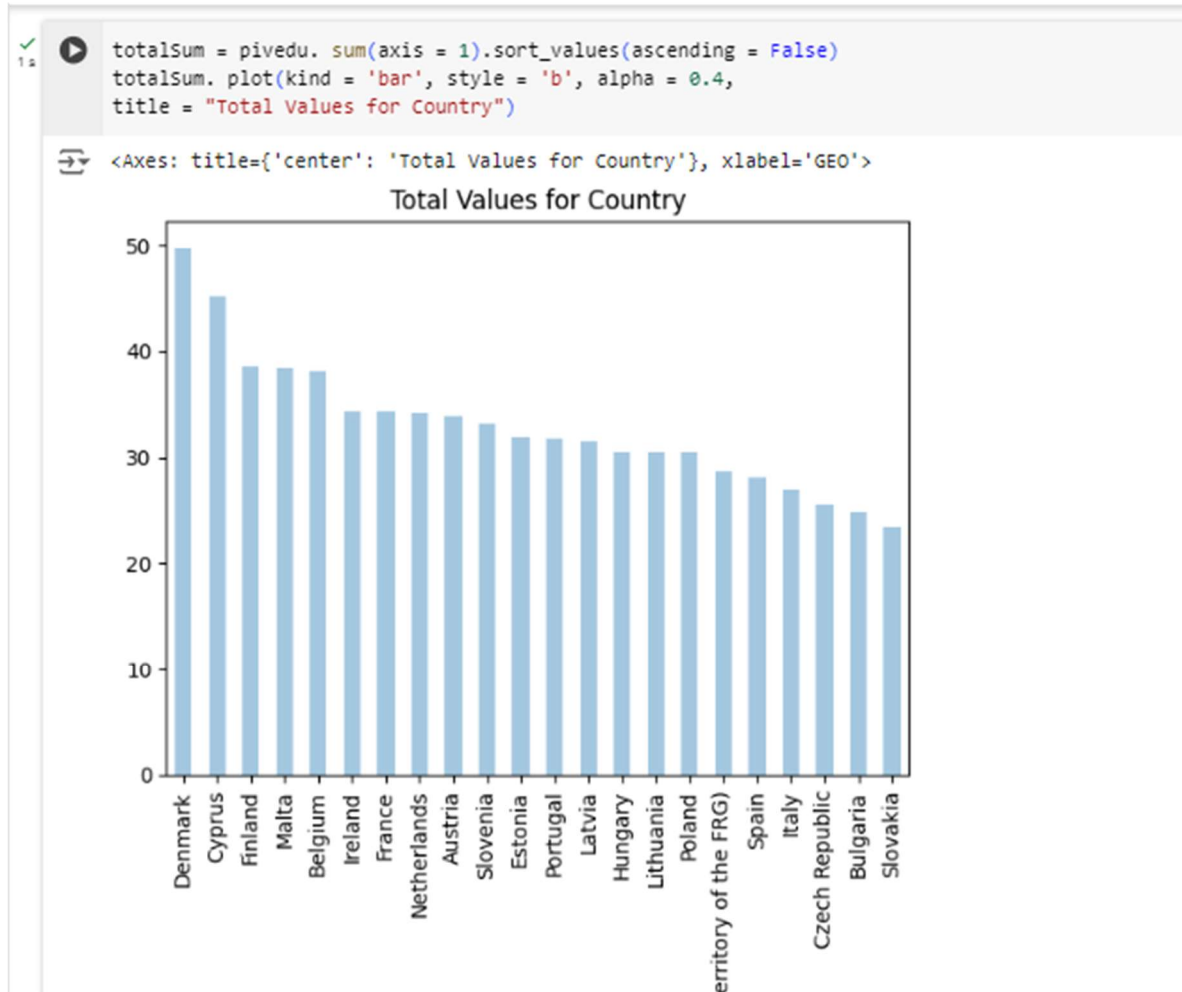
[36] totalSum = pivedu. sum(axis = 1).sort_values(ascending = False)

totalSum. plot(kind = 'bar', style = 'b', alpha = 0.4,

title = "Total Values for Country")

<Axes: title={'center': 'Total Values for Country'}, xlabel='GEO'>

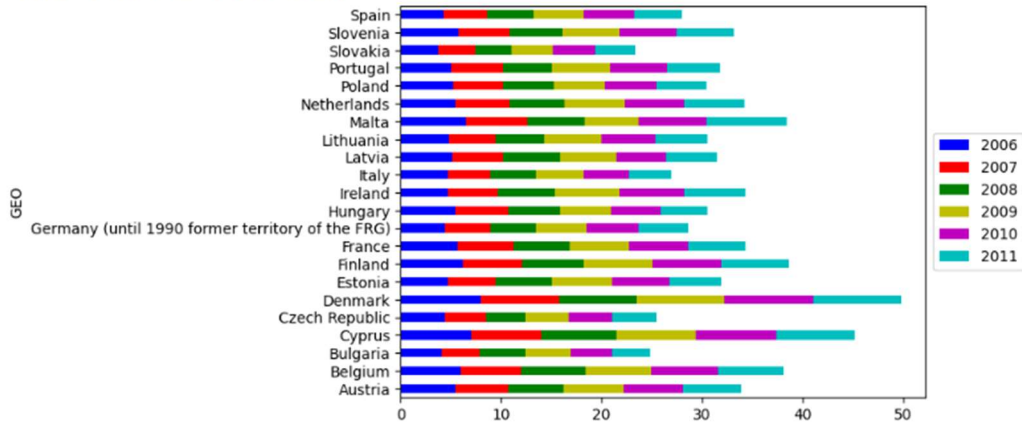
18. Por último lo que hacemos es graficar los datos en base a la librería Matplotlib. Con los datos colocados de cada país en los últimos 6 años.



19. Y también podemos realizar el diagrama de las barras más ordenadas del valor más alto al más bajo por cada uno de los países de manera mas organizada

```
[37] my_colors = ['b', 'r', 'g', 'y', 'm', 'c']  
ax = pivedu.plot(kind = 'barh',  
                stacked = True ,  
                color = my_colors)  
  
ax.legend(loc = 'center left', bbox_to_anchor = (1, .5))
```

<matplotlib.legend.Legend at 0x7da701027310>



```
[38] help(edu)
```

✓ 0 s se ejecutó 12:08 p.m.

CONCLUSION: