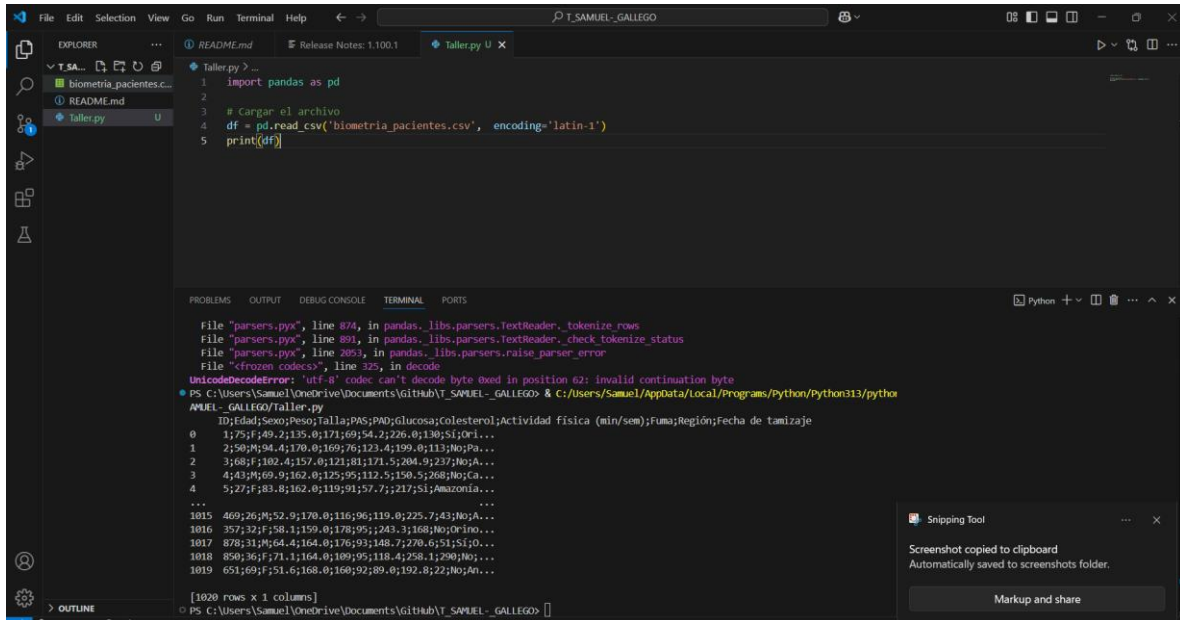


PDF Taller PROGRAMACION

TRANSFORMACIONES:



The screenshot shows a VS Code editor window with a Python file named `Taller.py` open. The file contains the following code:

```
1 import pandas as pd
2
3 # Cargar el archivo
4 df = pd.read_csv('biometria_pacientes.csv', encoding='latin-1')
5 print(df)
```

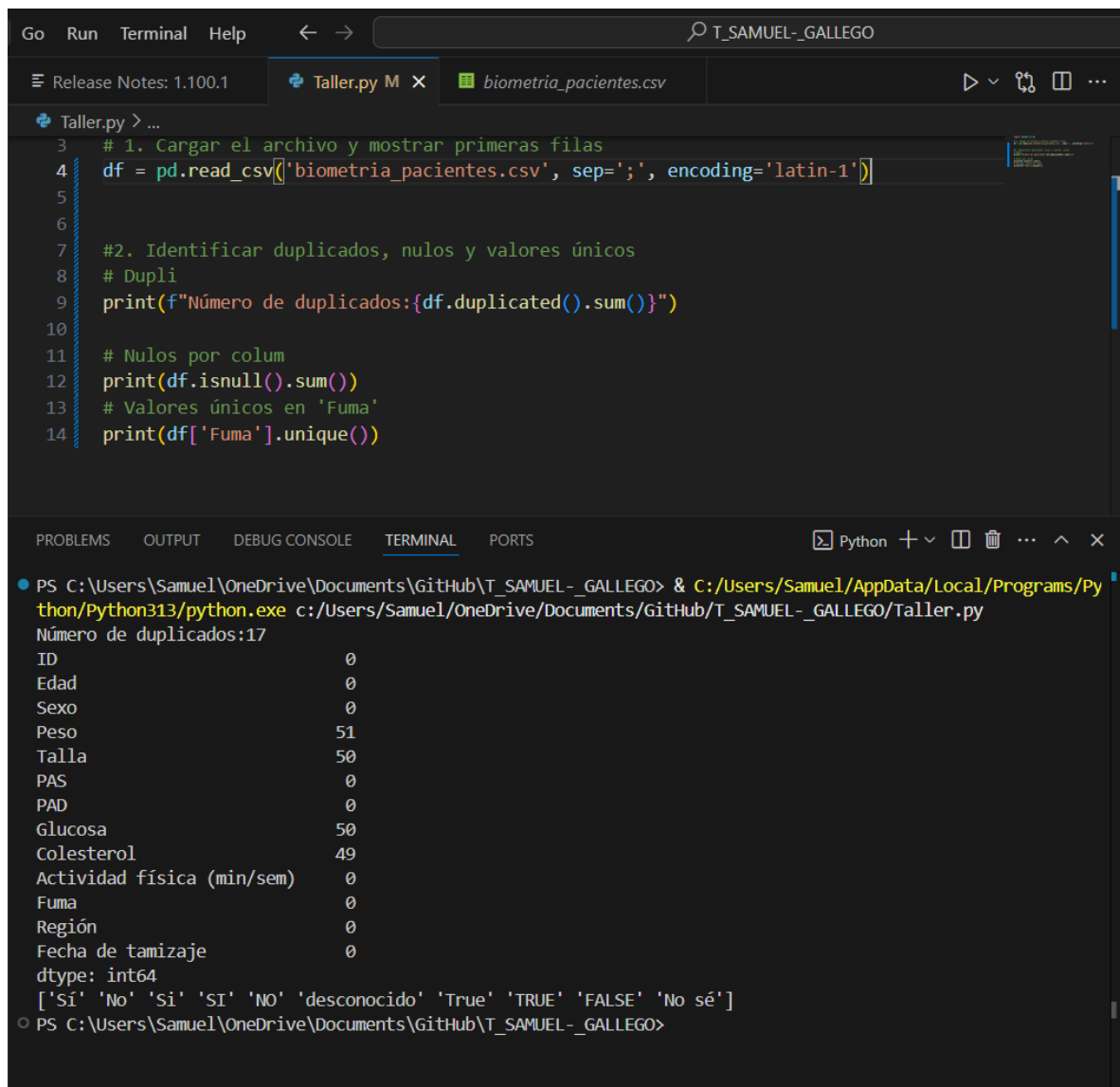
The terminal output shows the execution of the script, which results in a `UnicodeDecodeError` and a list of patient data rows. The error message is:

```
File "parsers.py", line 874, in pandas._libs.parsers.TextReader._tokenize_rows
File "parsers.py", line 891, in pandas._libs.parsers.TextReader._check_tokenize_status
File "parsers.py", line 2051, in pandas._libs.parsers.raise_parser_error
File "c:\frozen\codecs.py", line 325, in decode
UnicodeDecodeError: 'utf-8' codec can't decode byte 0xed in position 62: invalid continuation byte
```

The data rows are displayed as a list of tuples, with the first row being:

```
0 1;75;F;149.2;135.0;171.0;99.54.2;226.0;130;SI;Dr1...
```

The terminal output also shows the file path of the script and the location of the data file.



The image shows a VS Code editor window with a Python script named `Taller.py` and a terminal window below it. The script is designed to load a CSV file, identify duplicates, null values, and unique values. The terminal output shows the execution of the script, displaying the number of duplicates and a list of unique values for the 'Fuma' column.

```
3 # 1. Cargar el archivo y mostrar primeras filas
4 df = pd.read_csv('biometria_pacientes.csv', sep=';', encoding='latin-1')
5
6
7 # 2. Identificar duplicados, nulos y valores únicos
8 # Dupli
9 print(f"Número de duplicados:{df.duplicated().sum()}")
10
11 # Nulos por colum
12 print(df.isnull().sum())
13 # Valores únicos en 'Fuma'
14 print(df['Fuma'].unique())
```

Terminal Output:

```
PS C:\Users\Samuel\OneDrive\Documents\GitHub\T_SAMUEL-_GALLEGO> & C:/Users/Samuel/AppData/Local/Programs/Python/Python313/python.exe c:/Users/Samuel/OneDrive/Documents/GitHub/T_SAMUEL-_GALLEGO/Taller.py
Número de duplicados:17
ID                0
Edad              0
Sexo              0
Peso             51
Talla            50
PAS              0
PAD              0
Glucosa          50
Colesterol        49
Actividad física (min/sem) 0
Fuma              0
Región           0
Fecha de tamizaje 0
dtype: int64
['Sí' 'No' 'Si' 'SI' 'NO' 'desconocido' 'True' 'TRUE' 'FALSE' 'No sé']
PS C:\Users\Samuel\OneDrive\Documents\GitHub\T_SAMUEL-_GALLEGO>
```

Aca corriji un error de la separación que tenia al cargar el archivo agregando `sep=";"`

File Edit Selection View Go Run Terminal Help T.SAMUEL_GALLEGO

biometria_pacientes.csv

```
14 print(df['Fuma'].unique())
15
16
17 # 3. eliminar duplicados por columnas criticas
18 df = df.drop_duplicates()
19
20 columnas_criticas = ['Peso', 'Talla', 'Glucosa', 'colesterol']
21 df = df.dropna(subset=columnas_criticas)
22
23 # columna 'Fuma'
24 df['Fuma'] = df['Fuma'].str.lower().replace({
25     'sí': True, 'si': True, 'no': False,
26     'desconocido': False, 'true': True, 'false': False
27 }).astype(bool)
28
29 # mes de la fecha
30 df['Mes'] = pd.to_datetime(df['Fecha de tamizaje'], format='%d/%m/%Y').dt.month
31
32 print(df)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

	ID	Edad	Sexo	Peso	PAS	PAD	Glucosa	Colesterol	Actividad física (min/seu)	Fuma	Región	Fecha de tamizaje	Mes	
0	1	75	F	49.2	135.0	171	69	54.2	226.0	130	True	Orinoquia	2/10/2024	10
1	2	50	M	94.4	170.0	169	76	123.4	199.0	113	False	Pacifica	4/03/2025	3
2	3	68	F	102.4	157.0	121	81	171.5	204.9	237	False	Andina	25/03/2025	3
3	4	43	M	69.0	162.0	125	95	112.5	150.5	268	False	Caribe	30/09/2024	9
5	6	35	M	67.0	156.0	93	67	110.2	162.3	204	False	Amazonia	30/11/2024	11
...
997	998	20	F	57.1	168.0	167	109	68.8	186.3	147	False	Orinoquia	24/11/2024	11
999	1000	76	M	56.1	173.0	166	71	88.7	213.3	248	True	Orinoquia	27/06/2024	6
1002	1008	24	M	70.0	157.0	128	109	128.5	188.5	286	False	Orinoquia	16/03/2025	3
1005	33	73	M	82.3	159.0	124	101	28.7	90.9	91	False	Orinoquia	29/03/2025	3
1014	613	58	M	22.0	150.0	154	103	86.8	268.7	173	False	Amazonia	10/10/2024	10

[817 rows x 14 columns]

PS C:\Users\Samuel\OneDrive\Documents\GitHub\T.SAMUEL_GALLEGO>

main 0:21 0

Configuración de energía y suspensión

747

14/05/2025

File Edit Selection View Go Run Terminal Help T.SAMUEL_GALLEGO

biometria_pacientes.csv

```
31 df['Mes'] = pd.to_datetime(df['Fecha de tamizaje'], format='%d/%m/%Y').dt.month
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

	ID	Edad	Sexo	Peso	PAS	PAD	Glucosa	Colesterol	Actividad física (min/seu)	Fuma	Región	Fecha de tamizaje	Mes	BIC	Clasificación_BIC	Sedentario	Hipertenso	Metabólicamente_alterado	
0	1	75	F	49.2	135.0	171	69	54.2	226.0	130	True	Orinoquia	2/10/2024	10	26.999805	sobrepeso	False	True	False
1	2	50	M	94.4	170.0	169	76	123.4	199.0	113	False	Pacifica	4/03/2025	3	32.664946	obesidad	False	True	False
2	3	68	F	102.4	157.0	121	81	171.5	204.9	237	False	Andina	25/03/2025	3	41.543267	obesidad	False	False	True
3	4	43	M	69.0	162.0	125	95	112.5	150.5	268	False	Caribe	30/09/2024	9	36.434059	sobrepeso	False	True	False
5	6	35	M	67.0	156.0	93	67	110.2	162.3	204	False	Amazonia	30/11/2024	11	32.531229	sobrepeso	False	False	False
...
997	998	20	F	57.1	168.0	167	109	68.8	186.3	147	False	Orinoquia	24/11/2024	11	20.231880	Normal	False	True	False
999	1000	76	M	56.1	173.0	166	71	88.7	213.3	248	True	Orinoquia	27/06/2024	6	18.744362	Normal	False	True	False
1002	1008	24	M	70.0	157.0	128	109	128.5	188.5	286	False	Orinoquia	16/03/2025	3	26.290728	sobrepeso	False	True	False
1005	33	73	M	82.3	159.0	124	101	28.7	90.9	91	False	Orinoquia	29/03/2025	3	35.554802	obesidad	False	False	False
1014	613	58	M	22.0	150.0	154	103	86.8	268.7	173	False	Amazonia	10/10/2024	10	9.777778	baño peso	False	True	False

[817 rows x 19 columns]

PS C:\Users\Samuel\OneDrive\Documents\GitHub\T.SAMUEL_GALLEGO>

main 0:21 0

16:25

14/05/2025


```

66
67     """PARTE 3"""
68     # agrup por región
69     riesgo_region = df.groupby('Región')['Metabolicamente_alterado']
70     region_mas_riesgosa= riesgo_region.idxmax()
71     print("Región con mayor riesgo metabólico:")
72     print(region_mas_riesgosa.idxmax())
73
74     # agrup por mes
75     sedentarismo_mes = df.groupby('Mes')['Sedentario']
76     mes_mas_sedentario= sedentarismo_mes.idxmax()
77     print("Meses con mayor sedentarismo:")
78     print(mes_mas_sedentario.idxmax())
79

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

● PS C:\Users\Samuel\T_SAMUEL-_GALLEGO> & C:/Users/Samuel/AppData/Local/Programs/Python/Python38-32/Python.exe C:\Users\Samuel\T_SAMUEL-_GALLEGO\main.py

Número de duplicados:17

ID	0
Edad	0
Sexo	0
Peso	51
Talla	50
PAS	0
PAD	0
Glucosa	50
Colesterol	49
Actividad física (min/sem)	0
Fuma	0
Región	0
Fecha de tamizaje	0

dtype: int64

['Sí' 'No' 'Si' 'SI' 'NO' 'desconocido' 'True' 'TRUE' 'FALSE' 'No sé']

Región con mayor riesgo metabólico:
Orinoquía

Meses con mayor sedentarismo:
11

○ PS C:\Users\Samuel\T_SAMUEL-_GALLEGO>

HERRAMIENTAS:

Uso en general de esta librería para revisar los comandos y de las notas que tengo en mi cuaderno:

<https://aprendeconalf.es/docencia/python/manual/pandas/>

Usado en parte 1.1:



Usado en parte 1.3:

que comando puedo usar para extraer un mes de una columna de una fecha en pandas

Para extraer el mes de una columna de tipo fecha en Pandas, puedes usar el atributo `.dt.month`.

✅ Supongamos que tienes una columna de fechas:

```
python
df['fecha'] = pd.to_datetime(df['fecha']) # Asegúrate de que sea tipo datetime
df['mes'] = df['fecha'].dt.month
```

🌟 Otras opciones relacionadas:

• Mes como número (1-12):

```
python
df['mes'] = df['fecha'].dt.month
```

• Mes como nombre corto ("Jan", "Feb", etc.):

```
python
df['mes_nombre_corto'] = df['fecha'].dt.strftime('%b')
```

Pregunta lo que quieras

+ 🔍 Buscar 💡 Razona 🖼️ Crea una imagen ...

🗣️ 🎧

ChatGPT puede cometer errores. Considera verificar la información importante.

que comando puedo usar para extraer un mes de una columna de una fecha en pandas

Para extraer el mes de una columna de tipo fecha en Pandas, puedes usar el atributo `.dt.month`.

✅ Supongamos que tienes una columna de fechas:

```
python
df['fecha'] = pd.to_datetime(df['fecha']) # Asegúrate de que sea tipo datetime
df['mes'] = df['fecha'].dt.month
```

🌟 Otras opciones relacionadas:

• Mes como número (1-12):

```
python
df['mes'] = df['fecha'].dt.month
```

• Mes como nombre corto ("Jan", "Feb", etc.):

```
python
df['mes_nombre_corto'] = df['fecha'].dt.strftime('%b')
```

Pregunta lo que quieras

+ 🔍 Buscar 💡 Razona 🖼️ Crea una imagen ...

🗣️ 🎧

ChatGPT puede cometer errores. Considera verificar la información importante.

ChatGPT ▾

como puedo cambiar una columna de un tipo de dato a otro en py pandas



En Pandas, puedes cambiar el tipo de dato de una columna usando el método `.astype()`. Aquí te muestro cómo hacerlo paso a paso:

Usado en parte 2:

`df['Hipertenso'] = (df['PAS'] >= 140) or (df['PAD'] >= 90)` por que este fragmento de codigo podria generar un error ?

Ese fragmento de código:

```
python                                                                    Copiar Editar

df['Hipertenso'] = (df['PAS'] >= 140) or (df['PAD'] >= 90)
```

✱ Va a generar un error porque estás usando el operador lógico `or`, que no funciona con series de Pandas (Series booleanas) de esa manera.

✗ ¿Cuál es el problema?

`or` es un operador lógico para valores escalares (como `True` o `False`), no para vectores o columnas de datos. Cuando haces `df['PAS'] >= 140`, eso devuelve una serie de valores booleanos. Y `or` no sabe cómo comparar dos series.

El error típico que verás es:

```
vbrnet                                                                    Copiar Editar

ValueError: The truth value of a Series is ambiguous.
```

✓ ¿Cómo se soluciona?

Debes usar el operador elemento a elemento `|` (OR bit a bit), y agrupar las condiciones entre paréntesis:

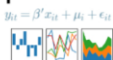
```
python                                                                    Copiar Editar

df['Hipertenso'] = (df['PAS'] >= 140) | (df['PAD'] >= 90)
```

Esto evaluará fila por fila si al menos una de las condiciones es verdadera, y guardará `True` o `False` en la nueva columna `'Hipertenso'`.

Consultado para en parte 3: "la verdad no lo use no entendí bien para que sirve"

pandas



Presentación

- Introducción a las estructuras de datos de pandas
- Creación de series y dataframes
- Inspección de series y dataframes

El método Series.agg

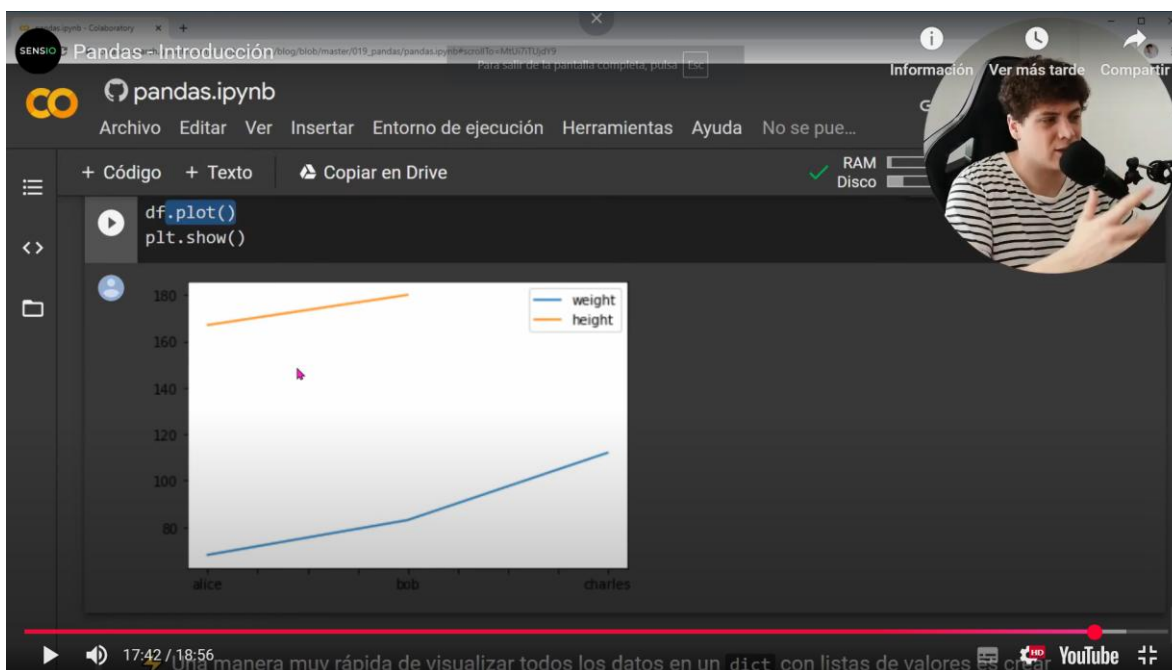
El método de las series pandas `Series.agg` aplica una o más funciones de agregación a los valores de la serie. La función a aplicar deberá pasarse como primer argumento y tendrá que ser el nombre de una función Python, una cadena de texto representando una función válida o un diccionario. En el caso de querer aplicar más de una función de agregación, éstas deberán pasarse como una lista.

Por ejemplo, consideremos la siguiente serie conteniendo información sobre las ventas realizadas en diferentes meses:

```
ventas = pd.Series([1, 3, 0, 2, 5, 4], index = ["Ene", "Mar", "Ene", "Feb", "Feb", "Ene"])
ventas
```

Ene	1
Mar	3
Ene	0
Feb	2
Feb	5
Ene	4

Usado en parte 4:



que es
`df.plot()`
`plt.show()` y como puedo usarlas en mi código por que, no lo reconoce

Parece que estás intentando usar `df.plot()` y `plt.show()`, pero tu código no las reconoce. Vamos a explicarlas y a ver por qué no funcionan, y cómo corregirlo.

Le pedí que resumiera todo lo que explico y me dijo esto:



Claro, aquí tienes un **hiperresumen**:

- `df.plot()` → genera una gráfica desde un DataFrame de **pandas**.
- `plt.show()` → muestra la gráfica en pantalla (**matplotlib**).
- Error común: escribiste `ptl.show()` en vez de `plt.show()`.
- Solución:

```
python

import pandas as pd
import matplotlib.pyplot as plt



df.plot()
plt.show()
```

 Copiar  Editar

- Instala las librerías si no las tienes:

```
bash

pip install pandas matplotlib
```

 Copiar  Editar