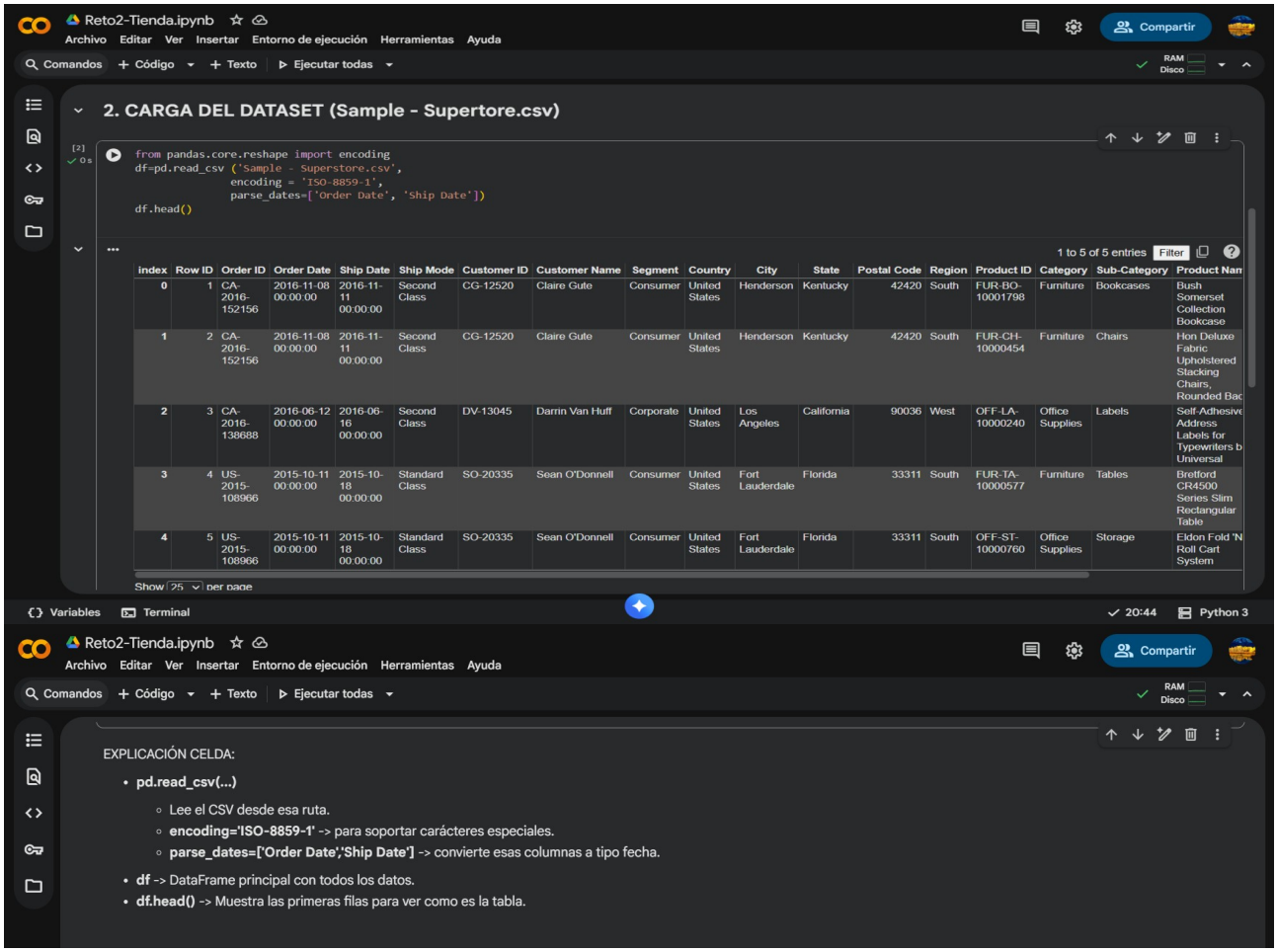
TIENDA - EDA + Ingeniería de Características

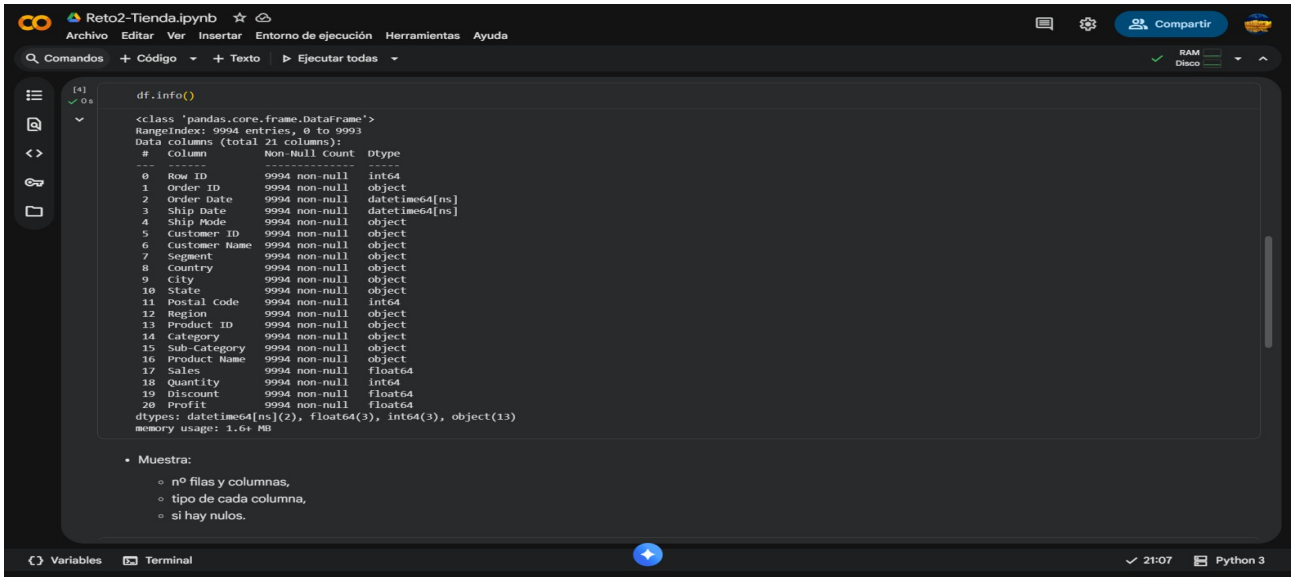
ÍNDICE DEL NOTEBOOK

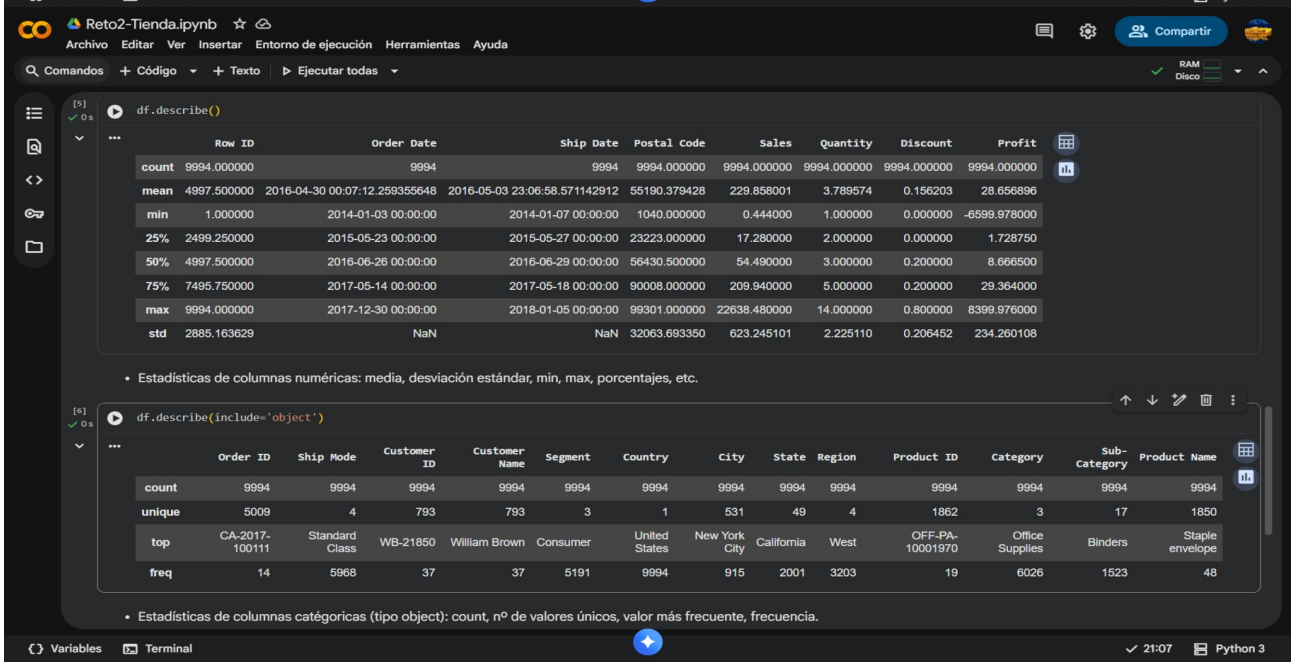
1. IMPORTACIONES Y CONFIGURACIÓN INICIAL

2. CARGA DEL DATASET (Store.csv)

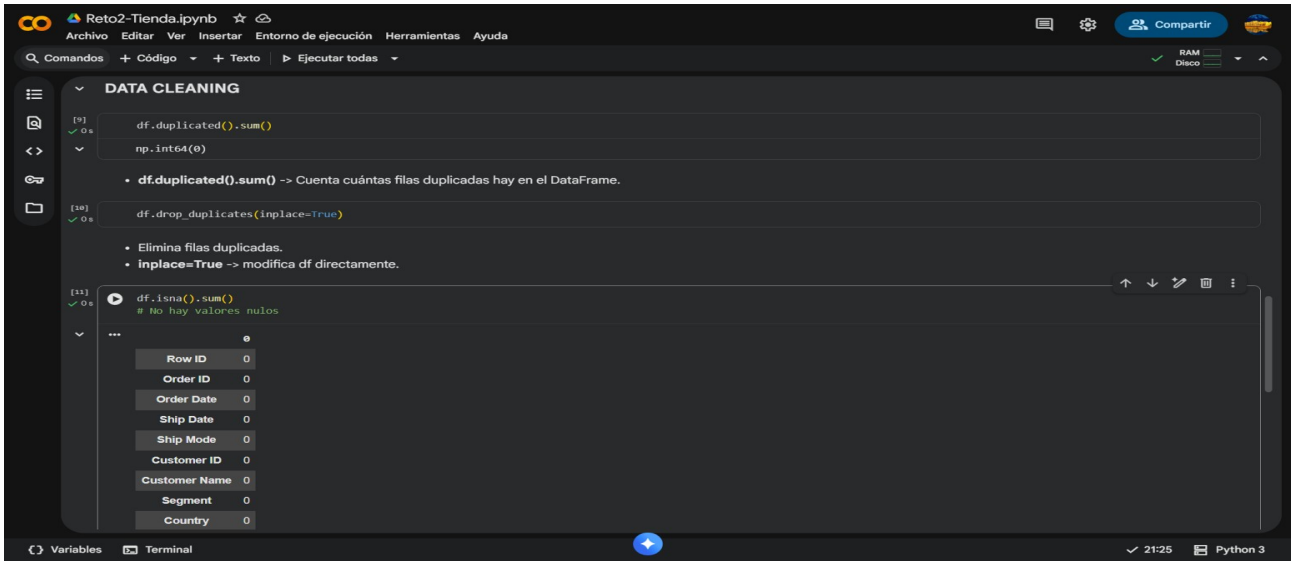


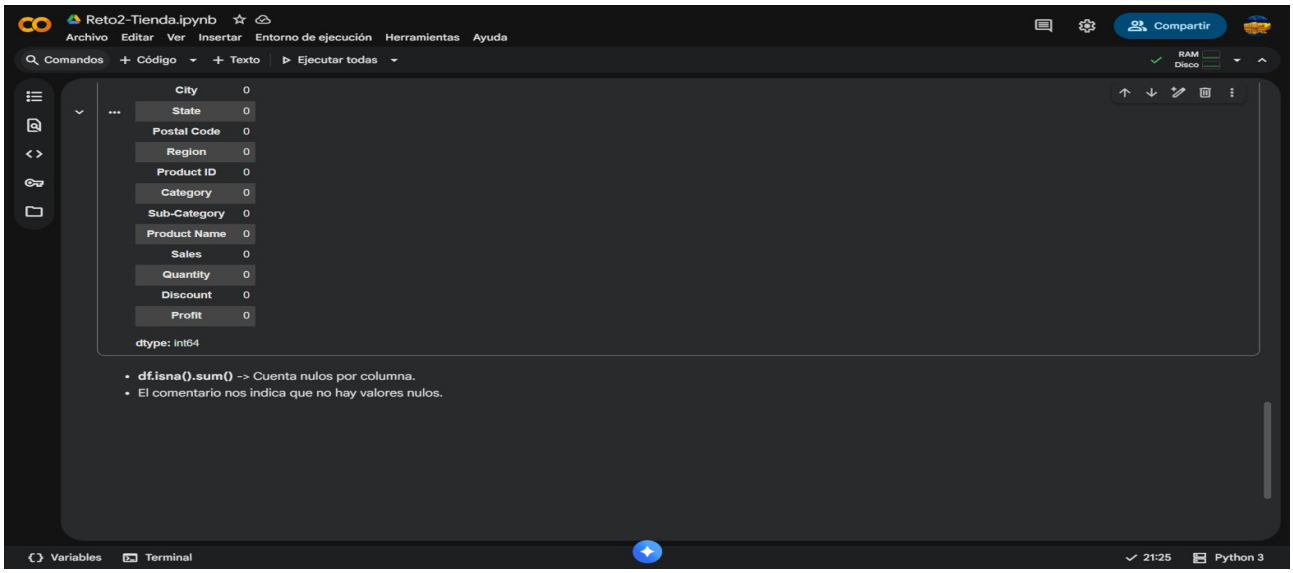
3. VISTA RÁPIDA DEL DATAFRAME



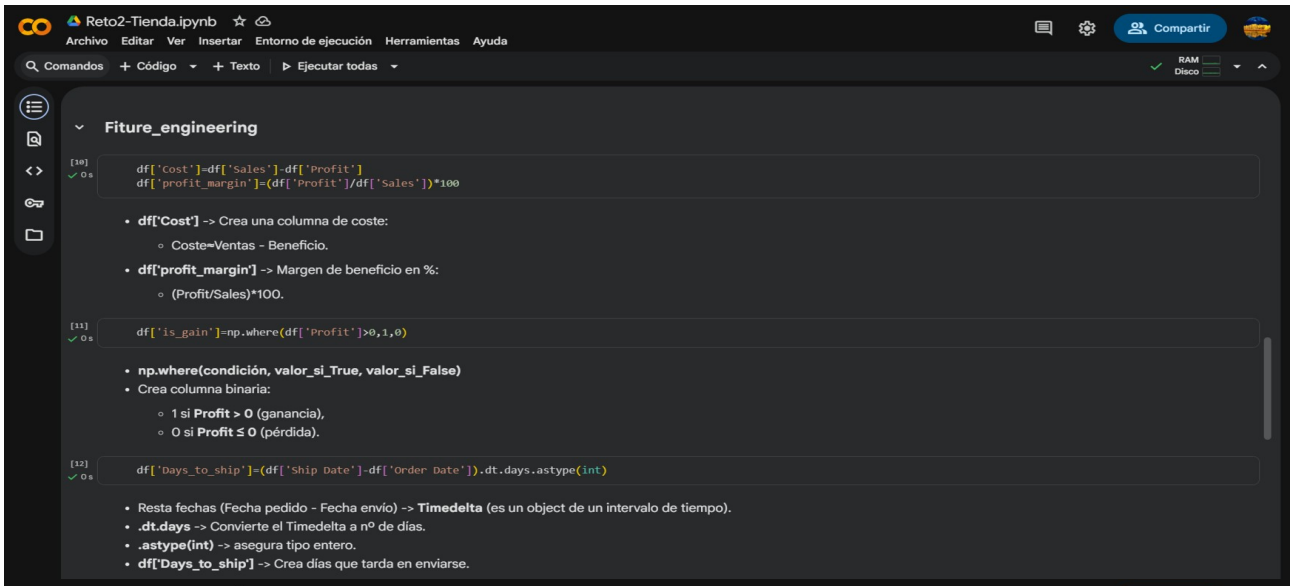


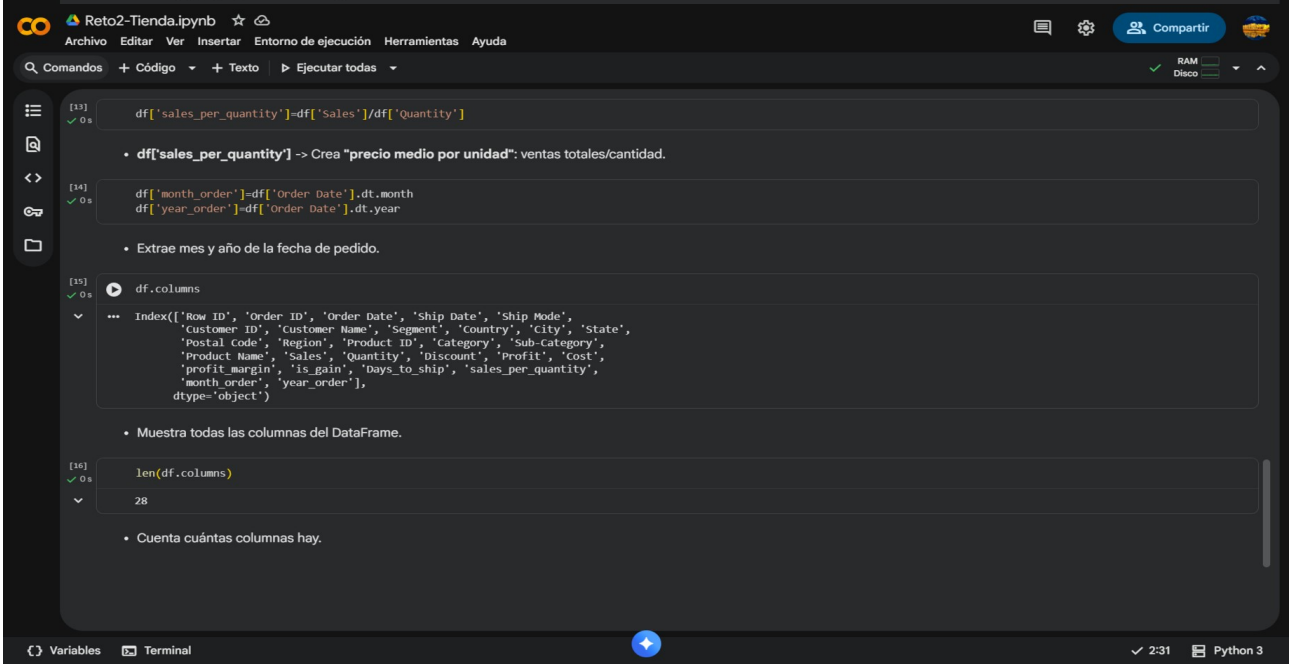
4. LIMPIEZA BÁSICA (DATA CLEANING)



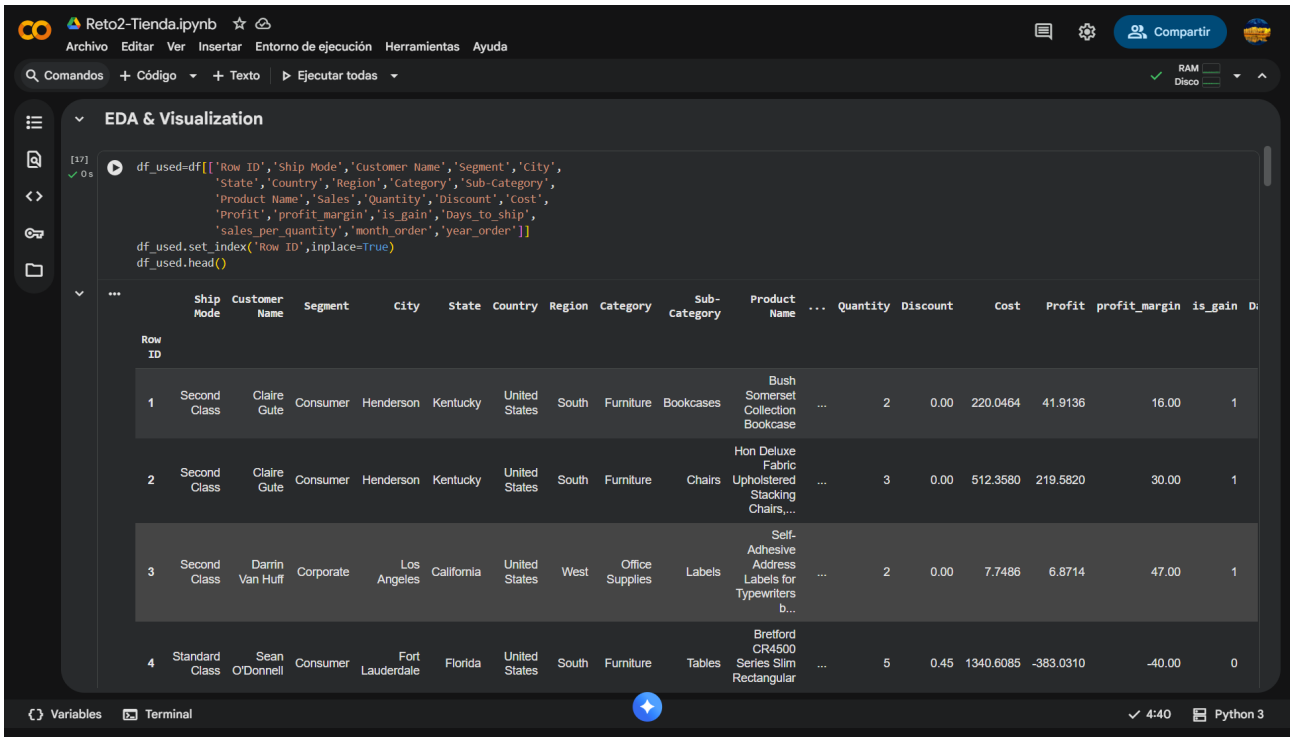


5. FEATURE ENGINEERING BÁSICO

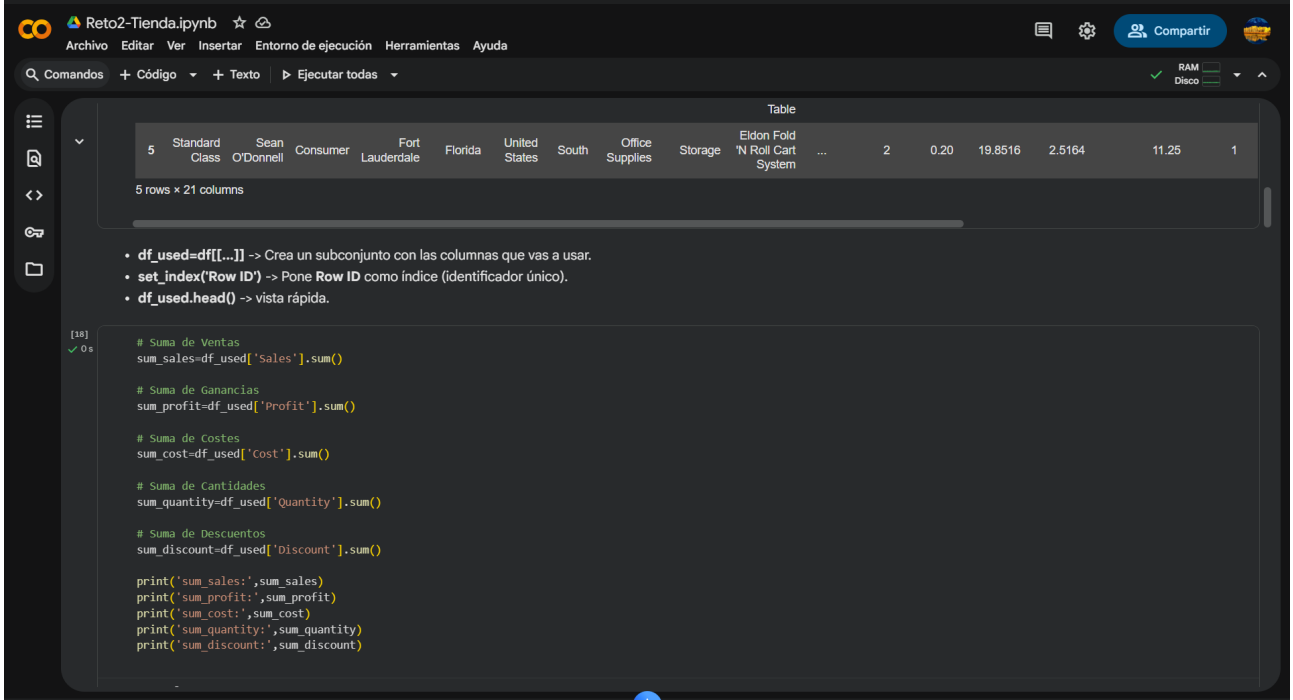




6. PREPARACIÓN Y PRIMEROS EDA



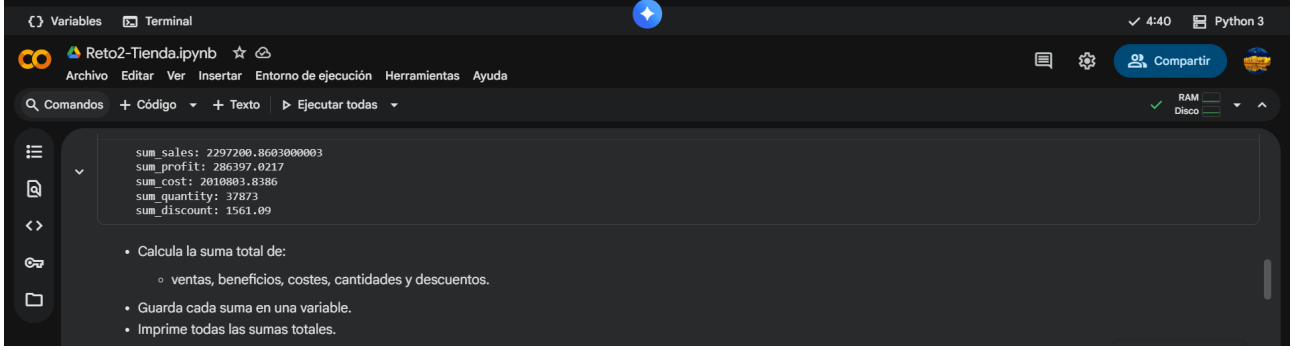
Row ID	Ship Mode	Customer Name	Segment	City	State	Country	Region	Category	Sub-Category	Product Name	...	Quantity	Discount	Cost	Profit	profit_margin	is_gain	D
1	Second Class	Claire Gute	Consumer	Henderson	Kentucky	United States	South	Furniture	Bookcases	Bush Somerset Collection Bookcase	...	2	0.00	220.0464	41.9136	16.00	1	
2	Second Class	Claire Gute	Consumer	Henderson	Kentucky	United States	South	Furniture	Chairs	Hon Deluxe Fabric Upholstered Stacking Chairs,...	...	3	0.00	512.3580	219.5820	30.00	1	
3	Second Class	Darrin Van Huff	Corporate	Los Angeles	California	United States	West	Office Supplies	Labels	Self-Adhesive Address Labels for Typewriters b...	...	2	0.00	7.7486	6.8714	47.00	1	
4	Standard Class	Sean O'Donnell	Consumer	Fort Lauderdale	Florida	United States	South	Furniture	Tables	Bretford CR4500 Series Slim Rectangular	...	5	0.45	1340.6085	-383.0310	-40.00	0	



5	Standard Class	Sean O'Donnell	Consumer	Fort Lauderdale	Florida	United States	South	Office Supplies	Storage	Eldon Fold 'N Roll Cart System	...	2	0.20	19.8516	2.5164	11.25	1	

5 rows × 21 columns

- **df_used=df[[...]]** -> Crea un subconjunto con las columnas que vas a usar.
- **set_index('Row ID')** -> Pone **Row ID** como índice (identificador único).
- **df_used.head()** -> vista rápida.



- Calcula la suma total de:
  - ventas, beneficios, costes, cantidades y descuentos.
- Guarda cada suma en una variable.
- Imprime todas las sumas totales.

Reto2-Tienda.ipynb

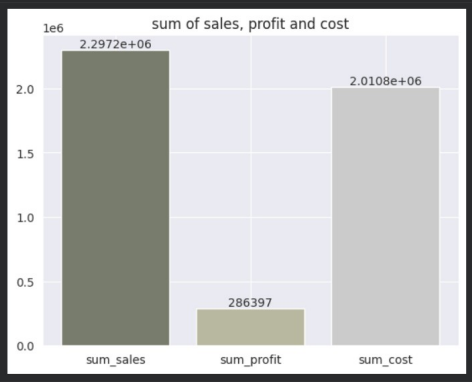
Archivo Editar Ver Insertar Entorno de ejecución Herramientas Ayuda

Comandos + Código + Texto ▷ Ejecutar todas

RAM Disco

Compartir

```
[20]
bars=plt.bar(['sum_sales','sum_profit','sum_cost'],
             [sum_sales,sum_profit,sum_cost],
             color=['#777C6D','#B7B89F','#CBCBCB'])
plt.title('sum of sales, profit and cost')
ax=plt.gca()
ax.spines['top'].set_visible(False)
ax.spines['right'].set_visible(False)
plt.bar_label(bars)
plt.show()
```



Reto2-Tienda.ipynb

Archivo Editar Ver Insertar Entorno de ejecución Herramientas Ayuda

Comandos + Código + Texto ▷ Ejecutar todas

RAM Disco

Compartir

- **plt.bar(...)** -> Gráfico de barras con las tres sumas totales.
- **plt.title(...)** -> Título del gráfico.
- **plt.gca()** -> Obtiene los ejes actuales.
- **ax.spines[...]** -> Oculta bordes superior y derecho para estética.
- **plt.bar_label(bars)** -> Pone etiquetas con los valores encima de cada barra.
- **plt.show()** -> Muestra el gráfico.

```
[21]
total_profit_margin=(sum_profit/sum_sales)*100
total_profit_margin
```

```
np.float64(12.467217240315604)
```

- Calcula el margen de beneficio total en %.
- **total_profit_margin** -> Muestra el valor.

Reto2-Tienda.ipynb

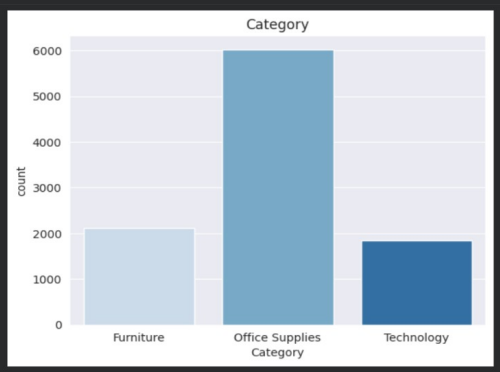
Archivo Editar Ver Insertar Entorno de ejecución Herramientas Ayuda

Comandos + Código + Texto ▷ Ejecutar todas

RAM Disco

Compartir

```
[38]
sns.countplot(data=df_used,x= 'Category' ,hue= 'Category' , palette= 'Blues' )
plt.title('Category')
plt.show()
```



- **sns.countplot** -> Cuenta cuántos registros hay por categoría.
- Eje X: **Category**.
- **palette='Blues'** -> Colores azules.
- Título del gráfico.
- **show** -> Muestra que **Office Supplies** es la más frecuente.

Variables Terminal

4:40 Python 3

Reto2-Tienda.ipynb

Archivo Editar Ver Insertar Entorno de ejecución Herramientas Ayuda

Comandos + Código + Texto Ejecutar todas

```
[23]
sales_furnitur=df.loc[df['Category']=='Furniture','Sales'].sum()
sales_office=df.loc[df['Category']=='Office Supplies','Sales'].sum()
sales_technology=df.loc[df['Category']=='Technology','Sales'].sum()

print('sales_furnitur:',sales_furnitur)
print('sales_office:',sales_office)
print('sales_technology:',sales_technology)

profit_furnitur=df.loc[df['Category']=='Furniture','Profit'].sum()
profit_office=df.loc[df['Category']=='Office Supplies','Profit'].sum()
profit_technology=df.loc[df['Category']=='Technology','Profit'].sum()

print(50*'-')

print('profit_furnitur:',profit_furnitur)
print('profit_office:',profit_office)
print('profit_technology:',profit_technology)
```

```
sales_furnitur: 741999.7953
sales_office: 719047.032
sales_technology: 836154.033
--------------------------------------------------
profit_furnitur: 18451.272800000006
profit_office: 122490.8008
profit_technology: 145454.9481
```

- **df['Category']=='...'** -> Crea un filtro booleano por categoría.
- **df.loc[filtro, 'Sales']** -> Selecciona las filas donde **Category = '...'**
- **.sum()** -> Suma esas **'Sales'** de la selección **'...'**
- Imprime las ventas totales de los tres resultados.
- Mismo estructura para **'Profit'**.

Reto2-Tienda.ipynb

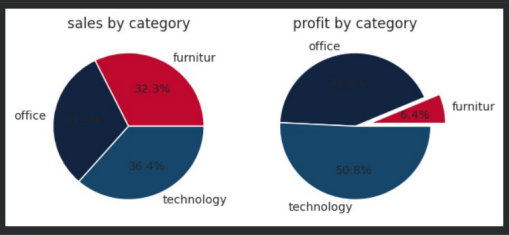
Archivo Editar Ver Insertar Entorno de ejecución Herramientas Ayuda

Comandos + Código + Texto Ejecutar todas

```
[26]
fig,ax=plt.subplots(1,2)
ax[0].pie([sales_furnitur,sales_office,sales_technology],
          labels=['furnitur','office','technology'],
          colors=['#BF092F','#132440','#16476A'],
          autopct='%1.1f%%')
ax[0].set_title('sales by category')

ax[1].pie([profit_furnitur,profit_office,profit_technology],
          labels=['furnitur','office','technology'],
          colors=['#BF092F','#132440','#16476A'],
          autopct='%1.1f%%',
          explode=[0.2,0,0])
ax[1].set_title('profit by category')

plt.show()
```



- **fig.ax=plt.subplots(1,2)** -> Crea una figura con dos subgráficos en una fila.
- Primer gráfico **(ax[0])**: pastel de ventas por categoría.
- Segundo **(ax[1])**: pastel de beneficios por categoría.
- **explode[0.2,0,0]** -> Resalta la primera porción (Furniture).

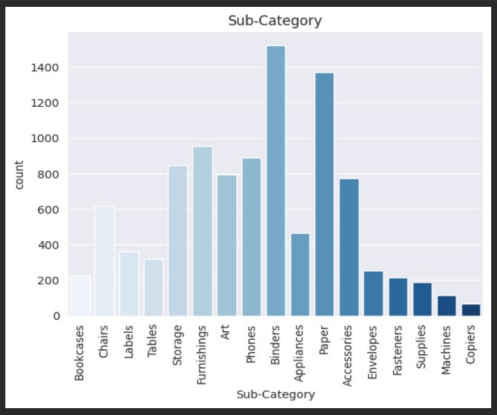
Variables Terminal 4:40 Python 3

Reto2-Tienda.ipynb

Archivo Editar Ver Insertar Entorno de ejecución Herramientas Ayuda

Comandos + Código + Texto Ejecutar todas

```
[37]
sns.countplot(data=df_used,x='Sub-Category',hue='Sub-Category', palette='Blues',legend=False)
plt.title('Sub-Category')
plt.xticks(rotation=90)
plt.show()
```



- Countplot por subcategoría.
- **plt.xticks(rotation=90)** -> Rota etiquetas 90º para que quepan.

Variables Terminal 4:40 Python 3

Reto2-Tienda.ipynb

Archivo Editar Ver Insertar Entorno de ejecución Herramientas Ayuda

Comandos + Código + Texto ▷ Ejecutar todas

RAM Disco

Compartir

[28] ✓ 0 s

```
order1=df_used.groupby('Sub-Category')['Sales'].sum().sort_values(ascending=False)
order2=df_used.groupby('Sub-Category')['Profit'].sum().sort_values(ascending=False)
```

- **df_used.groupby('Sub-Category')** -> Agrupar las filas por subcategoría:
  - **order1='Sales'.sum()** -> Por cada grupo de subcategoría suma ventas.
  - **order2='Profit'.sum()** -> Por cada grupo de subcategoría suma ganancias.
  - **.sort_values(ascending=False)** -> Ordena de mayor a menor.
- se guardan en las variables para usarlo en gráficos posteriores.

Reto2-Tienda.ipynb

Archivo Editar Ver Insertar Entorno de ejecución Herramientas Ayuda

Comandos + Código + Texto ▷ Ejecutar todas

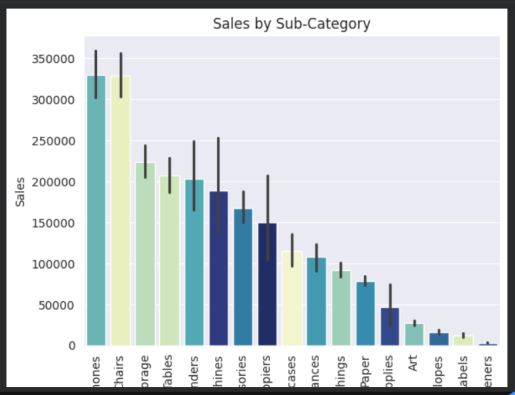
RAM Disco

Compartir

[35] ✓ 1 s

```
plt.figure(1)
sns.barplot(data=df_used,x='Sub-Category',y='Sales',hue='Sub-Category',
            palette='YlGnBu',legend=False,order=order1.index,estimator='sum')
plt.title('Sales by Sub-Category')
plt.xticks(rotation=90)

plt.figure(2)
sns.barplot(data=df_used,x='Sub-Category',y='Profit',hue='Sub-Category',
            palette='YlGnBu',legend=False,order=order2.index,estimator='sum')
plt.title('Profit by Sub-Category')
plt.xticks(rotation=90)
plt.show()
```



Variables Terminal ✓ 4:40 Python 3

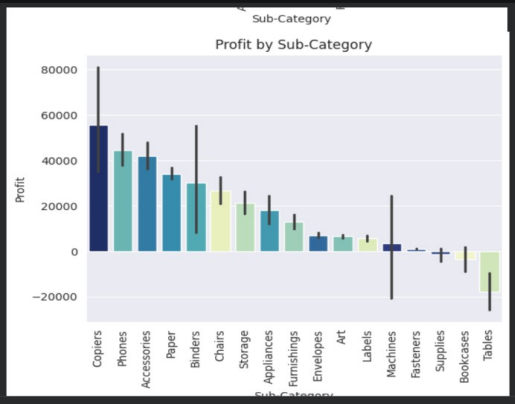
Reto2-Tienda.ipynb

Archivo Editar Ver Insertar Entorno de ejecución Herramientas Ayuda

Comandos + Código + Texto ▷ Ejecutar todas

RAM Disco

Compartir



- Creamos dos gráficos:
  - Primero: barra de ventas por subcategoría.
  - Segundo: barra de ganancias por subcategoría.
  - **order=order1.index/order2.index** -> Ordena de mayor a menor.
  - **estimator='sum'** -> Usa la suma como medida.

Variables Terminal ✓ 4:40 Python 3

Reto2-Tienda.ipynb

Archivo Editar Ver Insertar Entorno de ejecución Herramientas Ayuda

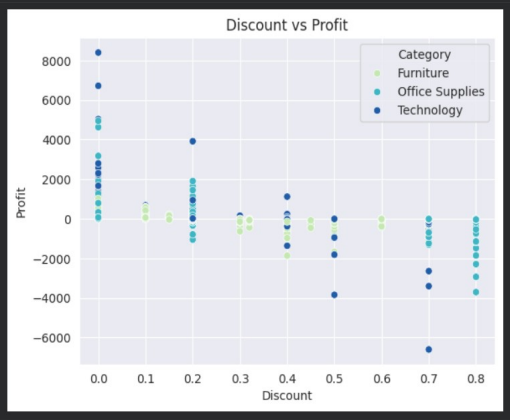
Comandos + Código + Texto Ejecutar todas

RAM Disco

Compartir

[34] 0 s

```
sns.scatterplot(data=df_used,x='Discount',y='Profit',hue='Category',palette='YlGnBu')
plt.title('Discount vs Profit')
plt.show()
```



- Dispersión descuento vs beneficio.

Reto2-Tienda.ipynb

Archivo Editar Ver Insertar Entorno de ejecución Herramientas Ayuda

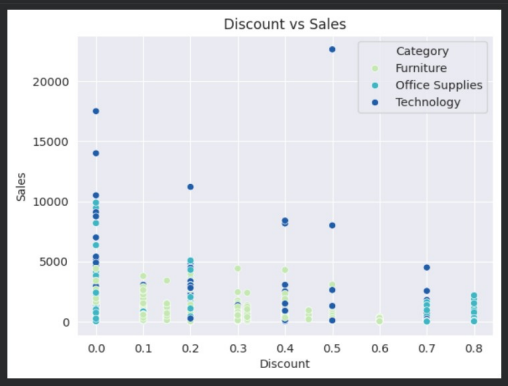
Comandos + Código + Texto Ejecutar todas

RAM Disco

Compartir

[33] 0 s

```
sns.scatterplot(data=df_used,x='Discount',y='Sales',hue='Category',palette='YlGnBu')
plt.title('Discount vs Sales')
plt.show()
```

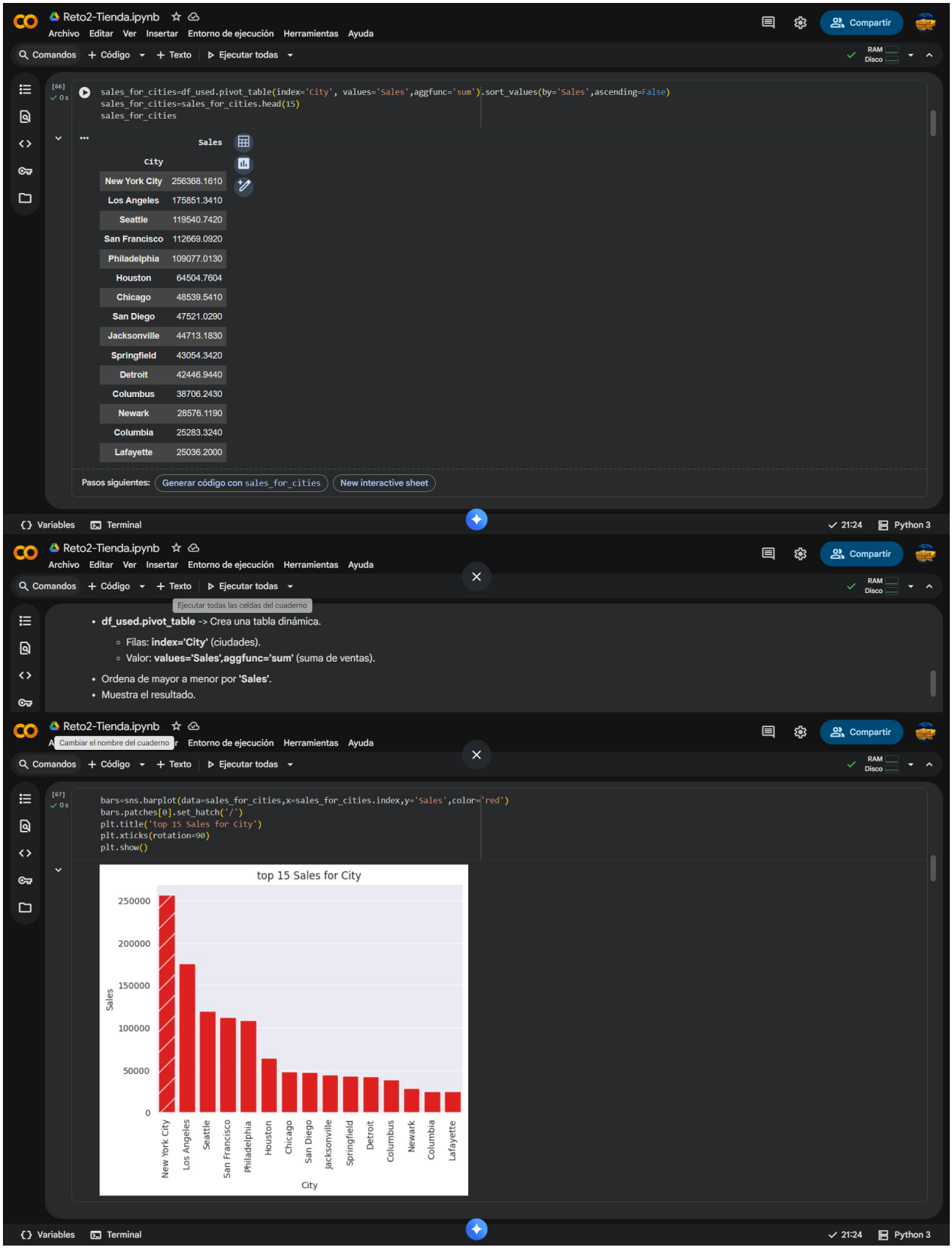


- Dispersión descuento vs ventas

Variables Terminal

4:40 Python 3

7. EDA POR CIUDAD, ESTADO, REGIÓN Y TIEMPO



Reto2-Tienda.ipynb

Archivo Editar Ver Insertar Entorno de ejecución Herramientas Ayuda

Comandos + Código + Texto Ejecutar todas

- Gráfico de barras con las 15 ciudades con más ventas.
- **bars.patches[0].set_hatch('/')** -> Pone la primera barra "rayada" para destacarla.

[68]
```
sales_for_state=df_used.pivot_table(index='State',values='Sales',aggfunc='sum').sort_values(by='Sales',ascending=False)
sales_for_state=sales_for_state.head(15)
sales_for_state
```

State	Sales
California	457687.6315
New York	310876.2710
Texas	170188.0458
Washington	138641.2700
Pennsylvania	116511.9140
Florida	89473.7080
Illinois	80166.1010
Ohio	78258.1360
Michigan	76269.6140
Virginia	70636.7200
North Carolina	55603.1640
Indiana	53555.3600
Georgia	49095.8400
Kentucky	36591.7500
New Jersey	35764.3120

Variables Terminal 21:24 Python 3

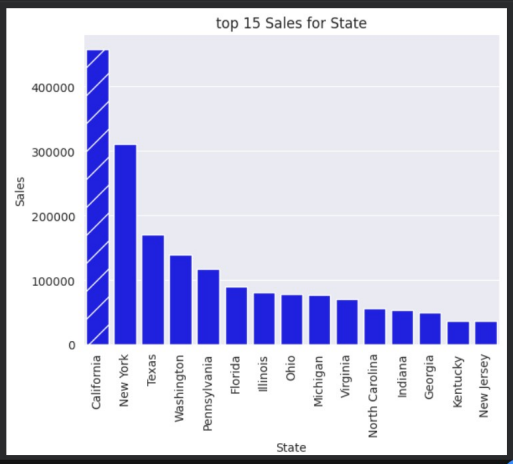
Reto2-Tienda.ipynb

Archivo Editar Ver Insertar Entorno de ejecución Herramientas Ayuda

Comandos + Código + Texto Ejecutar todas

- Igual pero con estado (state).

[69]
```
bars=sns.barplot(data=sales_for_state,x=sales_for_state.index,y='Sales',color='blue')
bars.patches[0].set_hatch('/')
plt.title('top 15 Sales for State')
plt.xticks(rotation=90)
plt.show()
```



Variables Terminal 21:24 Python 3

Reto2-Tienda.ipynb

Archivo Editar Ver Insertar Entorno de ejecución Herramientas Ayuda

Comandos + Código + Texto Ejecutar todas

- Gráfico de barras de venta por estado(top 15).

[70]
```
sales_for_region=df_used.pivot_table(index='Region',values='Sales',aggfunc='sum').sort_values(by='Sales',ascending=False)
sales_for_region
```

Region	Sales
West	725457.8245
East	678781.2400
Central	501239.8908
South	391721.9050

Pasos siguientes: Generar código con sales_for_region | New interactive sheet

- Ventas totales por región.

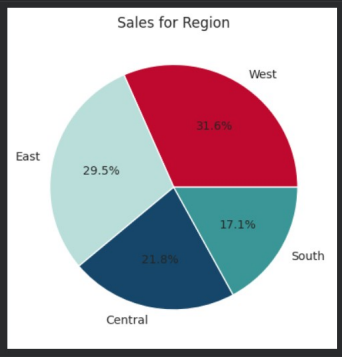
Reto2-Tienda.ipynb

Archivo Editar Ver Insertar Entorno de ejecución Herramientas Ayuda

Comandos + Código + Texto Ejecutar todas

- Ventas totales por región.

```
[71]
plt.pie(sales_for_region['Sales'],labels=sales_for_region.index,
        autopct='%1.1f%%',
        colors=['#BF092F','#BADFDB','#16476A','#3B9797'])
plt.title('Sales for Region')
plt.show()
```



- Gráfico de pastel de ventas por región.

Variables Terminal 21:24 Python 3

Reto2-Tienda.ipynb

Archivo Editar Ver Insertar Entorno de ejecución Herramientas Ayuda

Comandos + Código + Texto Ejecutar todas

```
[72]
top_20_customer=df_used.pivot_table(index='Customer Name',values='Sales',aggfunc='sum').sort_values(by='Sales',ascending=False)
top_20_customer=top_20_customer.head(20)
top_20_customer
```

Customer Name	Sales
Sean Miller	25043.050
Tamara Chand	19052.218
Raymond Buch	15117.339
Tom Ashbrook	14595.620
Adrian Barton	14473.571
Ken Lonsdale	14175.229
Sanjit Chand	14142.334
Hunter Lopez	12873.298
Sanjit Engle	12209.438
Christopher Conant	12129.072
Todd Sumrall	11891.751
Greg Tran	11820.120
Becky Martin	11789.630
Seth Vernon	11470.950
Caroline Jumper	11164.974
Clay Ludtke	10880.546
Maria Etezadi	10663.728

Variables Terminal 21:24 Python 3

Reto2-Tienda.ipynb

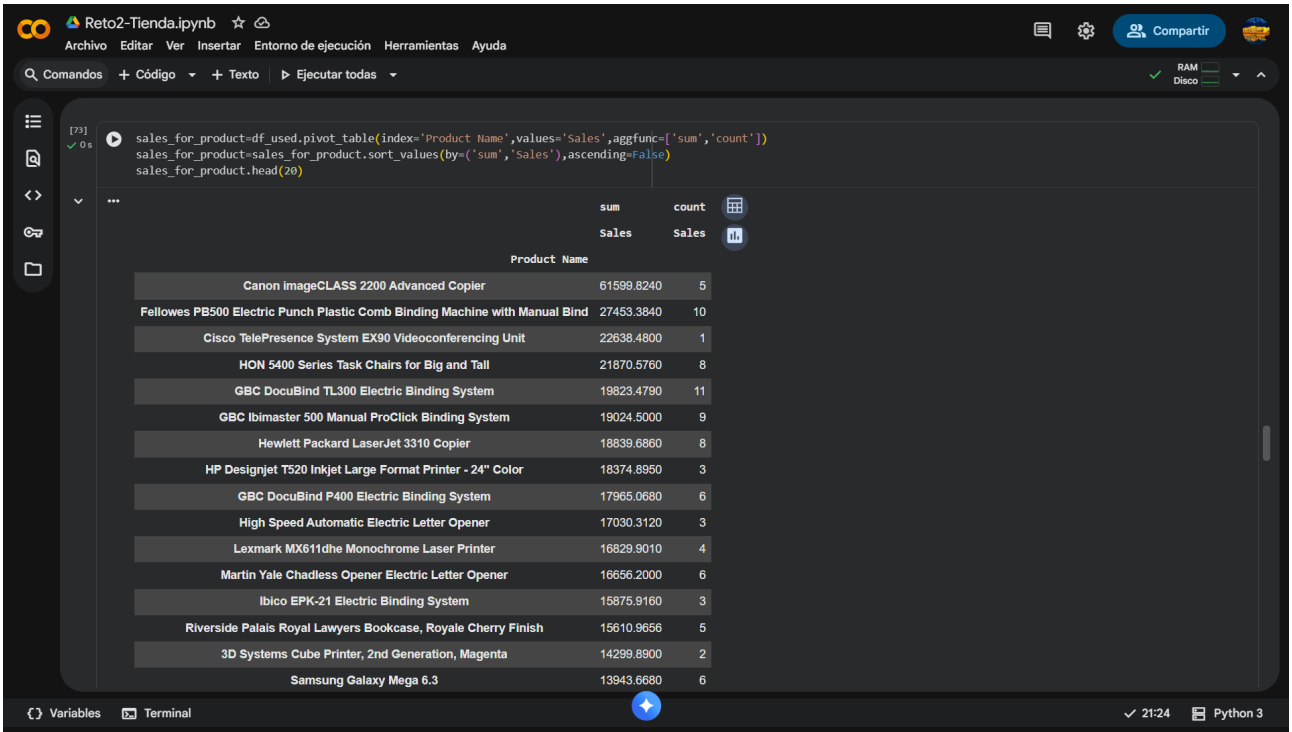
Archivo Editar Ver Insertar Entorno de ejecución Herramientas Ayuda

Comandos + Código + Texto Ejecutar todas

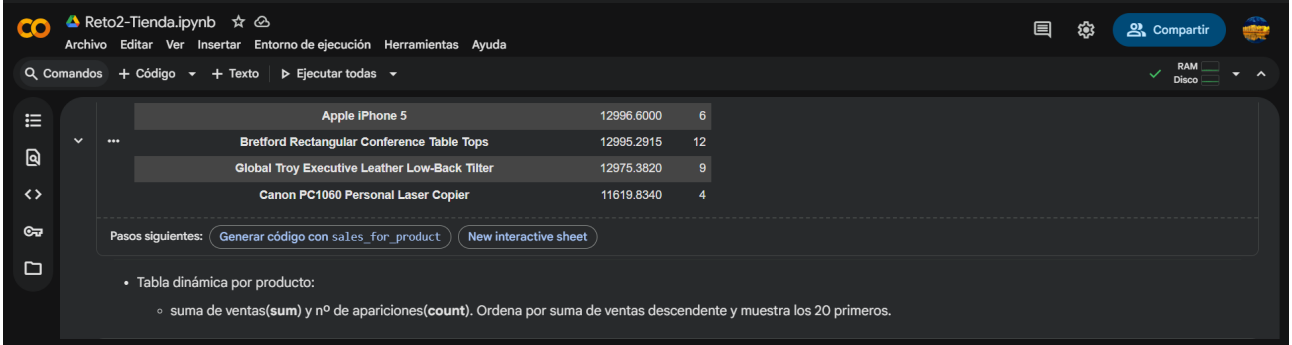
Karen Ferguson	10604.266
Bill Shonely	10501.653
Edward Hooks	10310.880

Pasos siguientes: Generar código con top_20_customer New interactive sheet

- Clientes ordenados de mayor a menor por ventas, top 20.

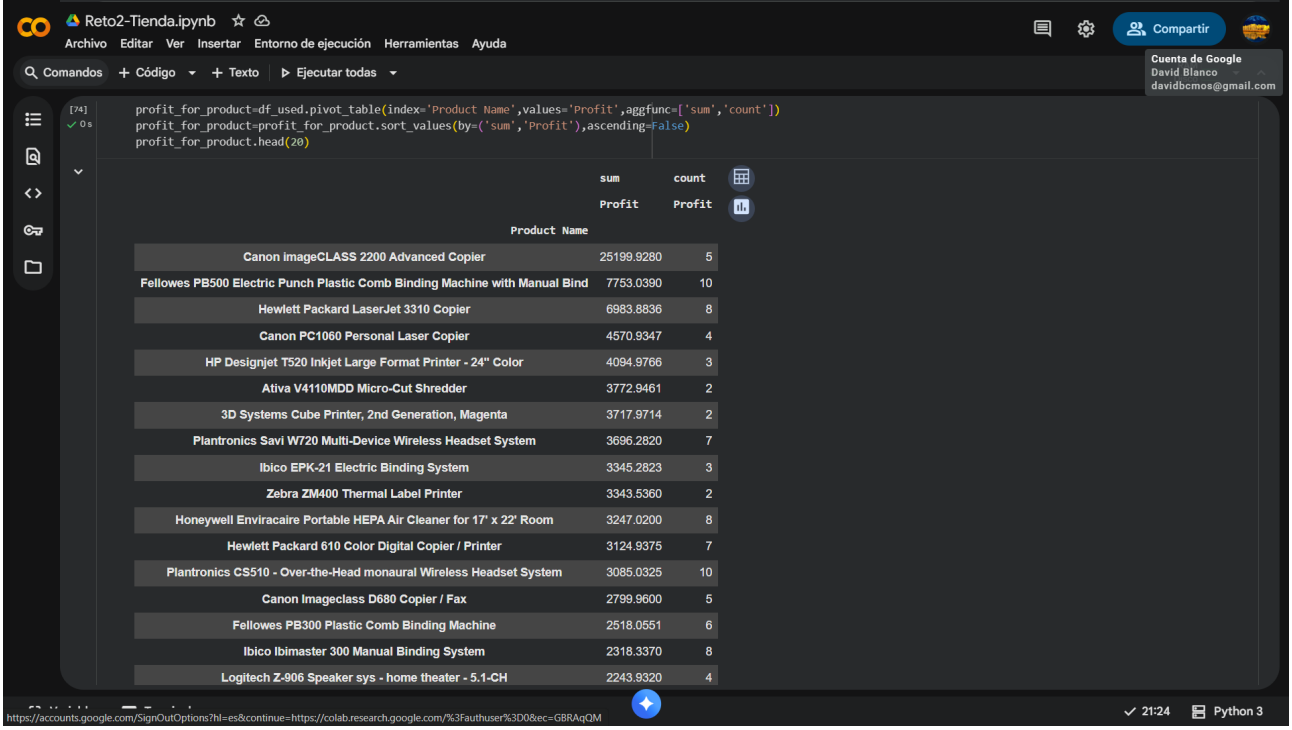


Product Name	sum Sales	count Sales
Canon imageCLASS 2200 Advanced Copier	61599.8240	5
Fellowes PB500 Electric Punch Plastic Comb Binding Machine with Manual Bind	27453.3840	10
Cisco TelePresence System EX90 Videoconferencing Unit	22638.4800	1
HON 5400 Series Task Chairs for Big and Tall	21870.5760	8
GBC DocuBind TL300 Electric Binding System	19823.4790	11
GBC Ibimaster 500 Manual ProClick Binding System	19024.5000	9
Hewlett Packard LaserJet 3310 Copier	18839.6860	8
HP Designjet T520 Inkjet Large Format Printer - 24" Color	18374.8950	3
GBC DocuBind P400 Electric Binding System	17965.0680	6
High Speed Automatic Electric Letter Opener	17030.3120	3
Lexmark MX611dhe Monochrome Laser Printer	16829.9010	4
Martin Yale Chadless Opener Electric Letter Opener	16656.2000	6
Ibico EPK-21 Electric Binding System	15875.9160	3
Riverside Palais Royal Lawyers Bookcase, Royale Cherry Finish	15610.9656	5
3D Systems Cube Printer, 2nd Generation, Magenta	14299.8900	2
Samsung Galaxy Mega 6.3	13943.6680	6

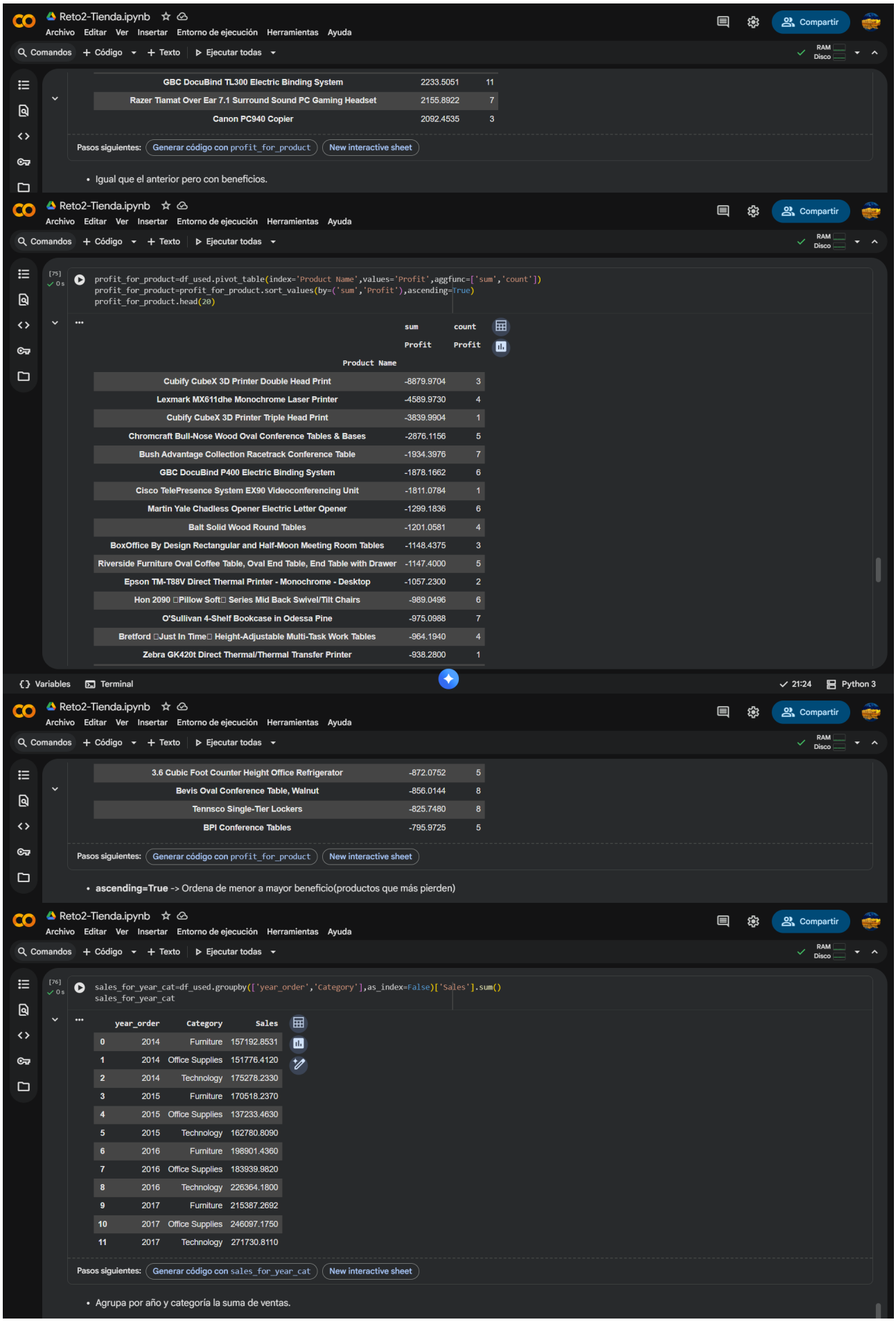


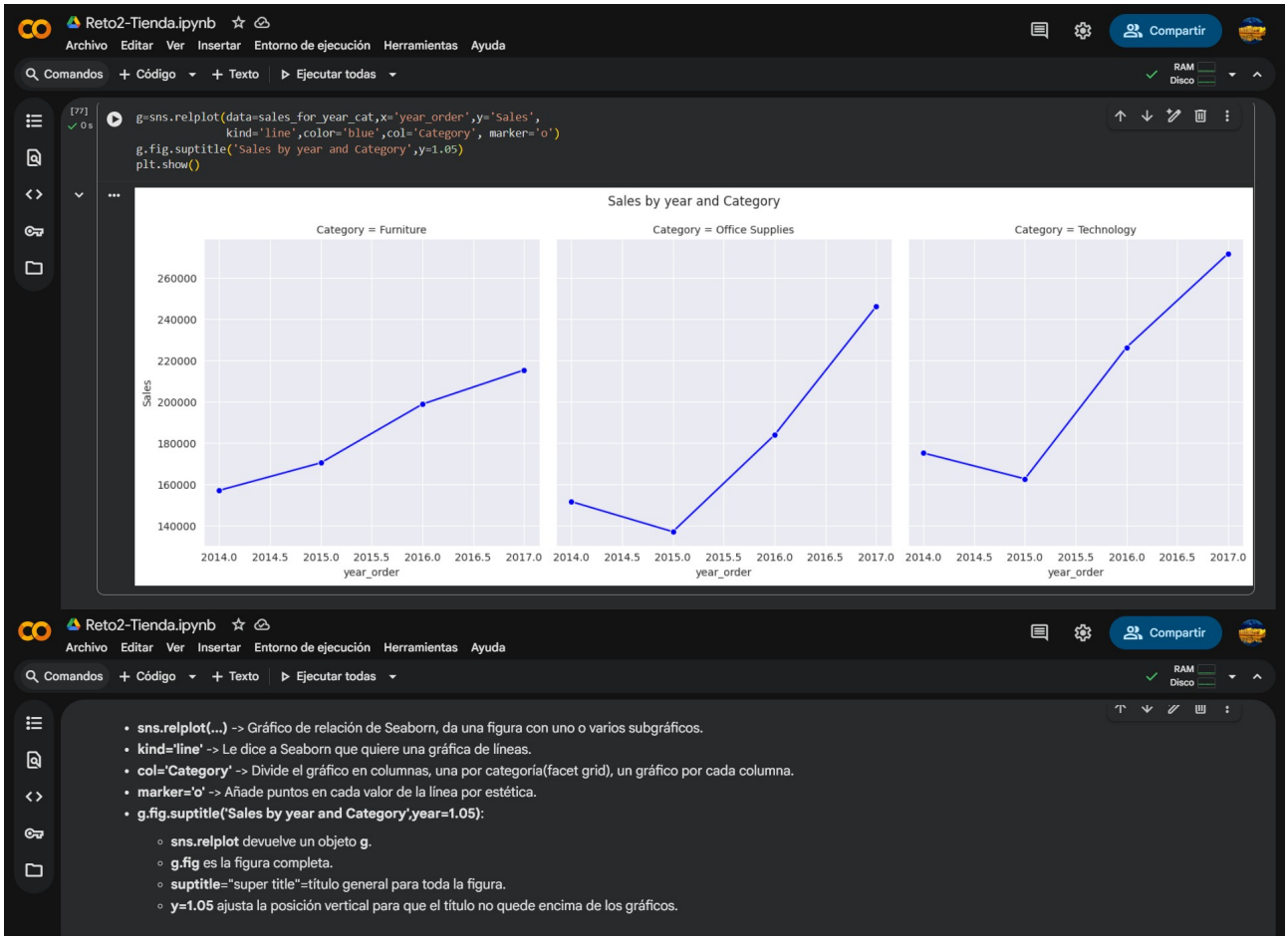
Apple iPhone 5	12996.6000	6
Bretford Rectangular Conference Table Tops	12995.2915	12
Global Troy Executive Leather Low-Back Tilter	12975.3820	9
Canon PC1060 Personal Laser Copier	11619.8340	4

- Tabla dinámica por producto:
  - suma de ventas(**sum**) y nº de apariciones(**count**). Ordena por suma de ventas descendente y muestra los 20 primeros.

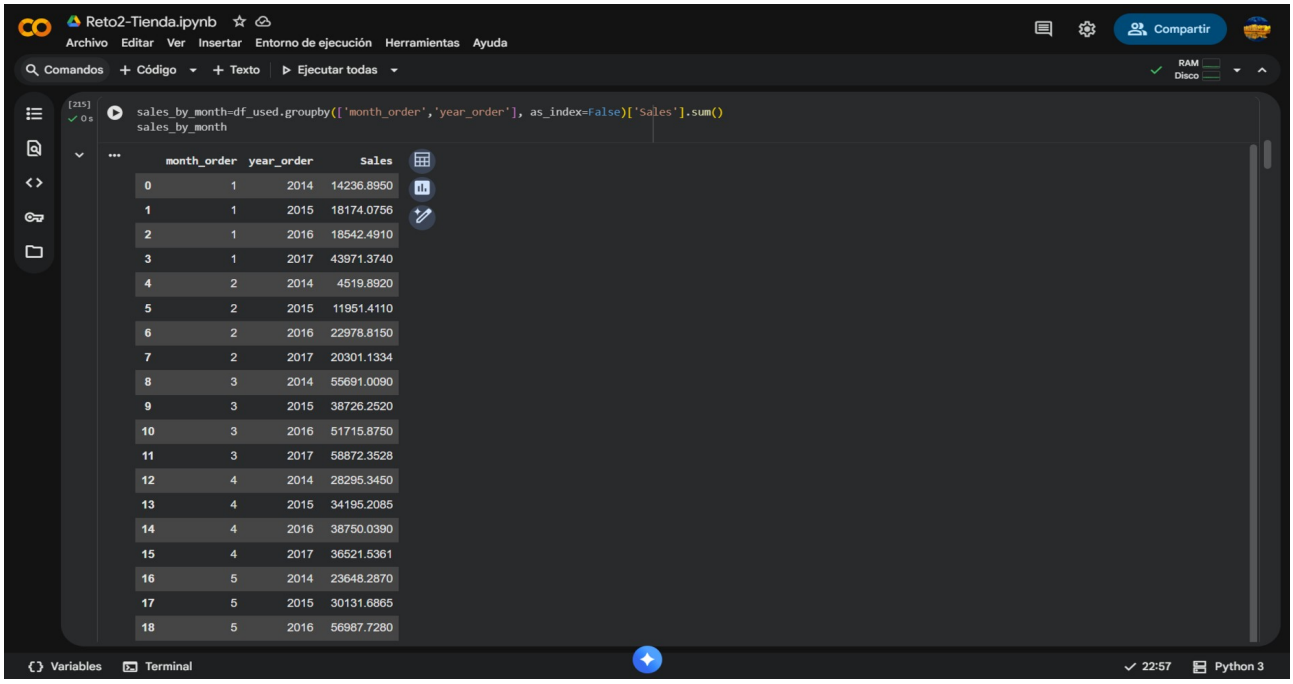


Product Name	sum Profit	count Profit
Canon imageCLASS 2200 Advanced Copier	25199.9280	5
Fellowes PB500 Electric Punch Plastic Comb Binding Machine with Manual Bind	7753.0390	10
Hewlett Packard LaserJet 3310 Copier	6983.8836	8
Canon PC1060 Personal Laser Copier	4570.9347	4
HP Designjet T520 Inkjet Large Format Printer - 24" Color	4094.9766	3
Ativa V4110MDD Micro-Cut Shredder	3772.9461	2
3D Systems Cube Printer, 2nd Generation, Magenta	3717.9714	2
Plantronics Savi W720 Multi-Device Wireless Headset System	3696.2820	7
Ibico EPK-21 Electric Binding System	3345.2823	3
Zebra ZM400 Thermal Label Printer	3343.5360	2
Honeywell Enviracaire Portable HEPA Air Cleaner for 17' x 22' Room	3247.0200	8
Hewlett Packard 610 Color Digital Copier / Printer	3124.9375	7
Plantronics CS510 - Over-the-Head monaural Wireless Headset System	3085.0325	10
Canon Imageclass D680 Copier / Fax	2799.9600	5
Fellowes PB300 Plastic Comb Binding Machine	2518.0551	6
Ibico Ibimaster 300 Manual Binding System	2318.3370	8
Logitech Z-906 Speaker sys - home theater - 5.1-CH	2243.9320	4


Reto2-Tienda.ipynb
Archivo Editar Ver Insertar Entorno de ejecución Herramientas Ayuda
Compartir
Comandos + Código + Texto Ejecutar todas
RAM Disco
GBC DocuBind TL300 Electric Binding System 2233.5051 11
Razer Tiamat Over Ear 7.1 Surround Sound PC Gaming Headset 2155.8922 7
Canon PC940 Copier 2092.4535 3
Pasos siguientes: Generar código con profit_for_product New interactive sheet
• Igual que el anterior pero con beneficios.
Reto2-Tienda.ipynb
Archivo Editar Ver Insertar Entorno de ejecución Herramientas Ayuda
Compartir
Comandos + Código + Texto Ejecutar todas
RAM Disco
[75]
0 s
profit_for_product=df_used.pivot_table(index='Product Name',values='Profit',aggfunc=['sum','count'])
profit_for_product=profit_for_product.sort_values(by=('sum','Profit'),ascending=True)
profit_for_product.head(20)
sum count
Profit Profit
Product Name
Cubify CubeX 3D Printer Double Head Print -8879.9704 3
Lexmark MX611dhe Monochrome Laser Printer -4589.9730 4
Cubify CubeX 3D Printer Triple Head Print -3839.9904 1
Chromcraft Bull-Nose Wood Oval Conference Tables & Bases -2876.1156 5
Bush Advantage Collection Racetrack Conference Table -1934.3976 7
GBC DocuBind P400 Electric Binding System -1878.1662 6
Cisco TelePresence System EX90 Videoconferencing Unit -1811.0784 1
Martin Yale Chadless Opener Electric Letter Opener -1299.1836 6
Balt Solid Wood Round Tables -1201.0581 4
BoxOffice By Design Rectangular and Half-Moon Meeting Room Tables -1148.4375 3
Riverside Furniture Oval Coffee Table, Oval End Table, End Table with Drawer -1147.4000 5
Epson TM-T88V Direct Thermal Printer - Monochrome - Desktop -1057.2300 2
Hon 2090 □Pillow Soft□ Series Mid Back Swivel/Tilt Chairs -989.0496 6
O'Sullivan 4-Shelf Bookcase in Odessa Pine -975.0988 7
Bretford □Just In Time□ Height-Adjustable Multi-Task Work Tables -964.1940 4
Zebra GK420t Direct Thermal/Thermal Transfer Printer -938.2800 1
Variables Terminal
21:24 Python 3
Reto2-Tienda.ipynb
Archivo Editar Ver Insertar Entorno de ejecución Herramientas Ayuda
Compartir
Comandos + Código + Texto Ejecutar todas
RAM Disco
3.6 Cubic Foot Counter Height Office Refrigerator -872.0752 5
Bevis Oval Conference Table, Walnut -856.0144 8
Tennsco Single-Tier Lockers -825.7480 8
BPI Conference Tables -795.9725 5
Pasos siguientes: Generar código con profit_for_product New interactive sheet
• ascending=True -> Ordena de menor a mayor beneficio(productos que más pierden)
Reto2-Tienda.ipynb
Archivo Editar Ver Insertar Entorno de ejecución Herramientas Ayuda
Compartir
Comandos + Código + Texto Ejecutar todas
RAM Disco
[76]
0 s
sales_for_year_cat=df_used.groupby(['year_order','Category'],as_index=False)['Sales'].sum()
sales_for_year_cat
year_order Category Sales
0 2014 Furniture 157192.8531
1 2014 Office Supplies 151776.4120
2 2014 Technology 175278.2330
3 2015 Furniture 170518.2370
4 2015 Office Supplies 137233.4630
5 2015 Technology 162780.8090
6 2016 Furniture 198901.4360
7 2016 Office Supplies 183939.9820
8 2016 Technology 226364.1800
9 2017 Furniture 215387.2692
10 2017 Office Supplies 246097.1750
11 2017 Technology 271730.8110
Pasos siguientes: Generar código con sales_for_year_cat New interactive sheet
• Agrupa por año y categoría la suma de ventas.



8. EDA POR MESES, SEGMENTOS, SHIPPING, ETC



	month_order	year_order	Sales
0	1	2014	14236.8950
1	1	2015	18174.0756
2	1	2016	18542.4910
3	1	2017	43971.3740
4	2	2014	4519.8920
5	2	2015	11951.4110
6	2	2016	22978.8150
7	2	2017	20301.1334
8	3	2014	55691.0090
9	3	2015	38726.2520
10	3	2016	51715.8750
11	3	2017	58872.3528
12	4	2014	28295.3450
13	4	2015	34195.2085
14	4	2016	38750.0390
15	4	2017	36521.5361
16	5	2014	23648.2870
17	5	2015	30131.6865
18	5	2016	56987.7280

Reto2-Tienda.ipynb

Archivo Editar Ver Insertar Entorno de ejecución Herramientas Ayuda

Comandos + Código + Texto Ejecutar todas

Compartir

Cuenta de Google
David Blanco
davidbcmos@gmail.com

29	8	2015	36898.3322
30	8	2016	31115.3743
31	8	2017	63120.8880
32	9	2014	81777.3508
33	9	2015	64595.9180
34	9	2016	73410.0249
35	9	2017	87866.6520
36	10	2014	31453.3930
37	10	2015	31404.9235
38	10	2016	59687.7450
39	10	2017	77776.9232
40	11	2014	78628.7167
41	11	2015	75972.5635
42	11	2016	79411.9658
43	11	2017	118447.8250
44	12	2014	69545.6205
45	12	2015	74919.5212
46	12	2016	96999.0430
47	12	2017	83829.3188

Pasos siguientes: Generar código con sales_by_month | New interactive sheet

- Ventas agrupadas por mes y año.

22:57 Python 3

Reto2-Tienda.ipynb

Archivo Editar Ver Insertar Entorno de ejecución Herramientas Ayuda

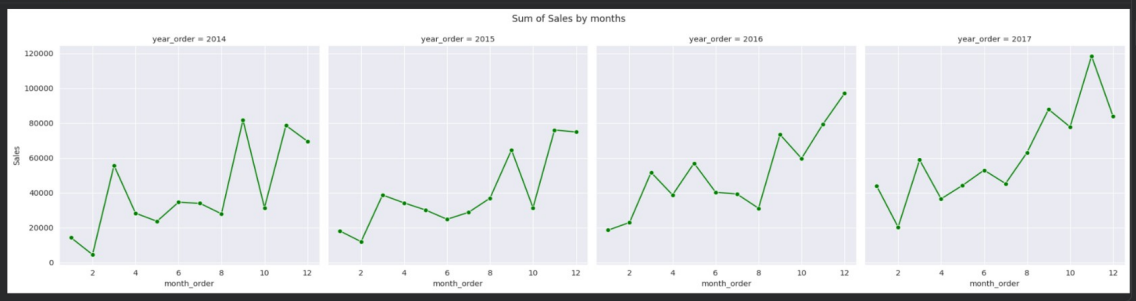
Comandos + Código + Texto Ejecutar todas

Compartir

RAM Disco

[216] 0 s

```
g=sns.relplot(data=sales_by_month,x='month_order',y='Sales',
              kind='line',color='green',col='year_order', marker='o')
g.fig.suptitle('Sum of Sales by months',y=1.05)
plt.show()
```



- Gráfico lineal de cada año mostrando ventas por mes.

Reto2-Tienda.ipynb

Archivo Editar Ver Insertar Entorno de ejecución Herramientas Ayuda

Comandos + Código + Texto Ejecutar todas

Compartir

RAM Disco

0 s

```
sales_year=df_used.pivot_table(index='year_order',values=['Sales','Profit'],aggfunc='sum')
sales_year.sort_values(by='Sales',ascending=False)
```

year_order	Profit	Sales
2017	93439.2696	733215.2552
2016	81795.1743	609205.5980
2014	49543.9741	484247.4981
2015	61618.6037	470532.5090

- Ventas y beneficios totales por año.
- Ordena de mayor a menor venta.

[218] 0 s

```
sales_mon=df_used.pivot_table(index='month_order',values='Sales',columns='year_order',aggfunc='sum',margins=True)
sales_mon
```

month_order \ year_order	2014	2015	2016	2017	All
1	14236.8950	18174.0756	18542.4910	43971.3740	9.492484e+04
2	4519.8920	11951.4110	22978.8150	20301.1334	5.975125e+04
3	55691.0090	38726.2520	51715.8750	58872.3528	2.050055e+05
4	28295.3450	34195.2085	38750.0390	36521.5361	1.377621e+05
5	23648.2870	30131.6865	56987.7280	44261.1102	1.550288e+05
6	34595.1276	24797.2920	40344.5340	52981.7257	1.527187e+05
7	33946.3930	28765.3250	39261.9630	45264.4160	1.472381e+05

Variables Terminal

22:57 Python 3

Reto2-Tienda.ipynb

Archivo Editar Ver Insertar Entorno de ejecución Herramientas Ayuda

Comandos + Código + Texto Ejecutar todas

8	27909.4685	36898.3322	31115.3743	63120.8880	1.590441e+05
9	81777.3508	64595.9180	73410.0249	87866.6520	3.076499e+05
10	31453.3930	31404.9235	59687.7450	77776.9232	2.003230e+05
11	78628.7167	75972.5635	79411.9658	118447.8250	3.524611e+05
12	69545.6205	74919.5212	96999.0430	83829.3188	3.252935e+05
All	484247.4981	470532.5090	609205.5980	733215.2552	2.297201e+06

Pasos siguientes: Generar código con sales_mon | New interactive sheet

- Tabla tipo "matriz":
  - Cada fila es un mes.
  - Sales es lo que quiero calcular.
  - Cada columna será un año diferente.
  - **margins=True** -> Añade una fila y una columna extra llamadas **All**:
    - La **fila All** muestra la suma total por columna(ventas totales por año).
    - La **columna All** muestra la suma total por fila(ventas totales en un mes sumando todos los años).

Reto2-Tienda.ipynb

Archivo Editar Ver Insertar Entorno de ejecución Herramientas Ayuda

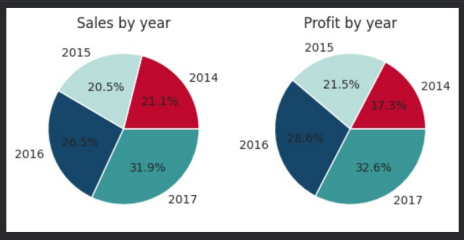
Comandos + Código + Texto Ejecutar todas

[219] ✓ 0 s

```
fig,ax=plt.subplots(1,2)
ax[0].pie(sales_year['Sales'],labels=sales_year.index,
          autopct='%1.1f%%',
          colors=['#BF092F','#BADFDB','#16476A','#3B9797'])
ax[0].set_title('Sales by year')

ax[1].pie(sales_year['Profit'],labels=sales_year.index,
          autopct='%1.1f%%',
          colors=['#BF092F','#BADFDB','#16476A','#3B9797'])
ax[1].set_title('Profit by year')

plt.show()
```



- Dos gráficos de pastel con distribución por ventas y beneficios por año.

Reto2-Tienda.ipynb

Archivo Editar Ver Insertar Entorno de ejecución Herramientas Ayuda

Comandos + Código + Texto Ejecutar todas

[220] ✓ 0 s

```
sns.countplot(data=df_used,x='Days_to_ship',hue='Days_to_ship', palette='Greens')
plt.title('Days to ship')
plt.show()
```

- Muestra cuántos pedidos tardan X días en enviarse.

Reto2-Tienda.ipynb

Archivo Editar Ver Insertar Entorno de ejecución Herramientas Ayuda

Comandos + Código + Texto ▷ Ejecutar todas

```
sales_state=df.groupby(['State','Category'])['Sales'].sum().sort_values(ascending=False).reset_index()
sales_state=sales_state.head(20)
sales_state
```

	State	Category	Sales
0	California	Technology	159271.0820
1	California	Furniture	156064.6015
2	California	Office Supplies	142351.9480
3	New York	Technology	127483.5000
4	New York	Furniture	93372.7290
5	New York	Office Supplies	90020.0420
6	Texas	Technology	65104.2240
7	Texas	Furniture	60593.2918
8	Washington	Technology	50536.7100
9	Washington	Furniture	48020.1520
10	Florida	Technology	46968.0360
11	Texas	Office Supplies	44490.5300
12	Pennsylvania	Technology	42215.2690
13	Washington	Office Supplies	40084.4080
14	Pennsylvania	Furniture	39354.9310
15	Michigan	Office Supplies	37723.7590
16	Ohio	Technology	35675.9920
17	Pennsylvania	Office Supplies	34941.7140
18	Illinois	Technology	31983.6730

Variables Terminal 22:57 Python 3

Reto2-Tienda.ipynb

Archivo Editar Ver Insertar Entorno de ejecución Herramientas Ayuda

Comandos + Código + Texto ▷ Ejecutar todas

19	Illinois	Furniture	28274.5220

Pasos siguientes: Generar código con sales_state New interactive sheet

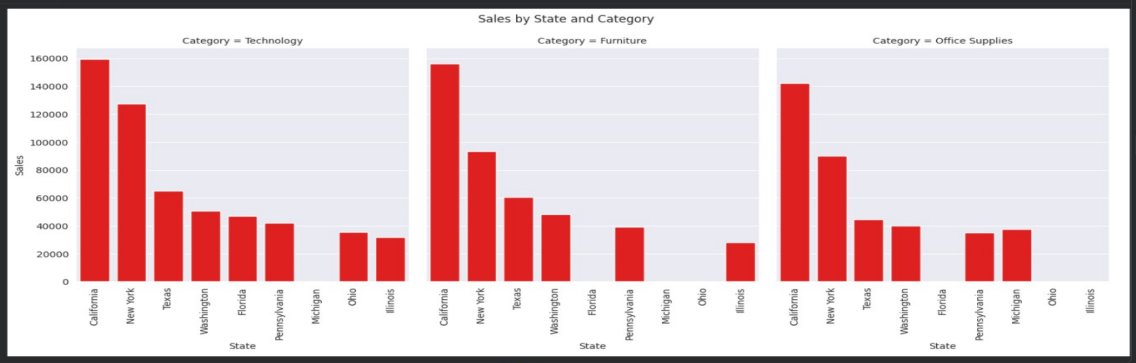
- Agrupa ventas por estado y categoría.
- Ordena descendentemente y toma top 20 combinaciones.

Reto2-Tienda.ipynb

Archivo Editar Ver Insertar Entorno de ejecución Herramientas Ayuda

Comandos + Código + Texto ▷ Ejecutar todas

```
g=sns.catplot(data=sales_state,x='State',y='Sales',
              kind='bar', color='red',col='Category')
g.fig.suptitle('Sales by State and Category',y=1.05)
g.set_xticklabels(rotation=90)
plt.show()
```



Variables Terminal 22:57 Python 3

Reto2-Tienda.ipynb

Archivo Editar Ver Insertar Entorno de ejecución Herramientas Ayuda

Comandos + Código + Texto ▷ Ejecutar todas

- **sns.catplot** -> Tipo de gráfico de Seaborn para visualizar **categorías**.
- **kind='bar'** -> Indica que el gráfico será de barras.
- **col='Category'** -> Haz un gráfico separado para cada categoría.

```
sales_state=df.groupby(['Region','Category'])['Sales'].sum().sort_values(ascending=False).reset_index()
sales_state=sales_state.head(20)
sales_state
```

	Region	Category	Sales
0	East	Technology	264973.9810
1	West	Furniture	252612.7435
2	West	Technology	251991.8320
3	West	Office Supplies	220853.2490
4	East	Furniture	208291.2040
5	East	Office Supplies	205516.0550
6	Central	Technology	170416.3120
7	Central	Office Supplies	167026.4150
8	Central	Furniture	163797.1638
9	South	Technology	148771.9080
10	South	Office Supplies	125651.3130
11	South	Furniture	117298.6840

Pasos siguientes: Generar código con sales_state New interactive sheet

- Igual pero agrupando por región + categoría.

Variables Terminal 22:57 Python 3

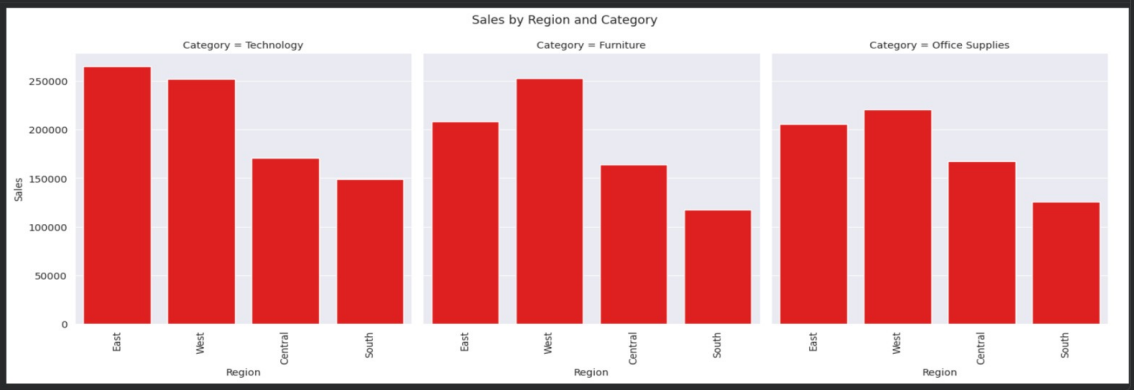
Reto2-Tienda.ipynb

Archivo Editar Ver Insertar Entorno de ejecución Herramientas Ayuda

Comandos + Código + Texto Ejecutar todas

[224]

```
g=sns.catplot(data=sales_state,x='Region',y='Sales',
              kind='bar',color='red',col='Category')
g.fig.suptitle('Sales by Region and Category',y=1.05)
g.set_xticklabels(rotation=90)
plt.show()
```



- Gráfico de barras por región y categoría.

Variables Terminal 22:57 Python 3

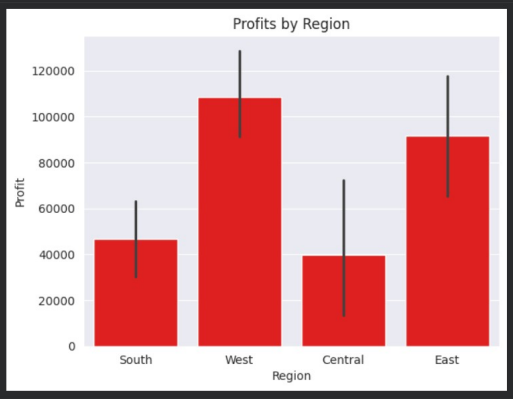
Reto2-Tienda.ipynb

Archivo Editar Ver Insertar Entorno de ejecución Herramientas Ayuda

Comandos + Código + Texto Ejecutar todas

[225]

```
bars=sns.barplot(data=df_used,x='Region',y='Profit',color='red',estimator=np.sum)
plt.title('Profits by Region')
plt.show()
```



- Suma de beneficio total por región.
- **estimator=np.sum** -> Indica como calcular cada barra, por defecto es **estimator=np.mean**.

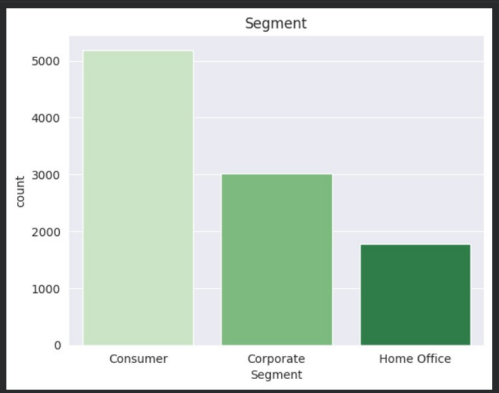
Reto2-Tienda.ipynb

Archivo Editar Ver Insertar Entorno de ejecución Herramientas Ayuda

Comandos + Código + Texto Ejecutar todas

[226]

```
sns.countplot(data=df_used,x='Segment',hue='Segment',palette='Greens')
plt.title('Segment')
plt.show()
```



- Cuenta cuántos registros hay en cada segmento(Consumer,Corporate,etc).

Reto2-Tienda.ipynb

Archivo Editar Ver Insertar Entorno de ejecución Herramientas Ayuda

Comandos + Código + Texto ▷ Ejecutar todas

Compartir

RAM Disco

[227] ✓ 0 s

```
df_used.pivot_table(index='Segment',values='Sales',aggfunc='sum').sort_values(by='Sales',ascending=False)
```

Segment	Sales
Consumer	1.161401e+06
Corporate	7.061464e+05
Home Office	4.296531e+05

- Ventas por segmento, ordenadas de mayor a menor.

[228] ✓ 0 s

```
df_used.pivot_table(index='Segment',values='Profit',aggfunc='sum').sort_values(by='Profit',ascending=False)
```

Segment	Profit
Consumer	134119.2092
Corporate	91979.1340
Home Office	60298.6785

- Beneficio por segmento.

Reto2-Tienda.ipynb

Archivo Editar Ver Insertar Entorno de ejecución Herramientas Ayuda

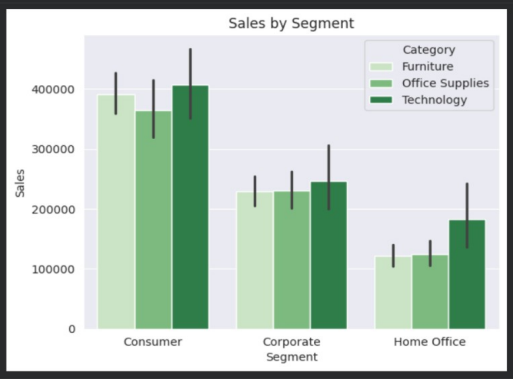
Comandos + Código + Texto ▷ Ejecutar todas

Compartir

RAM Disco

[229] ✓ 1 s

```
sns.barplot(data=df_used,x='Segment',y='Sales',
            hue='Category',palette='Greens',estimator=sum)
plt.title('Sales by Segment')
plt.show()
```



- Gráfico de barras apiladas por segmento y categoría(sales).

Reto2-Tienda.ipynb

Archivo Editar Ver Insertar Entorno de ejecución Herramientas Ayuda

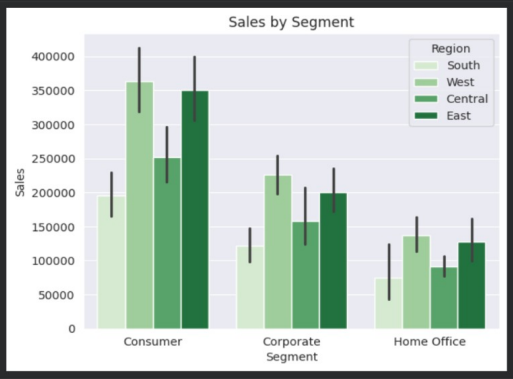
Comandos + Código + Texto ▷ Ejecutar todas

Compartir

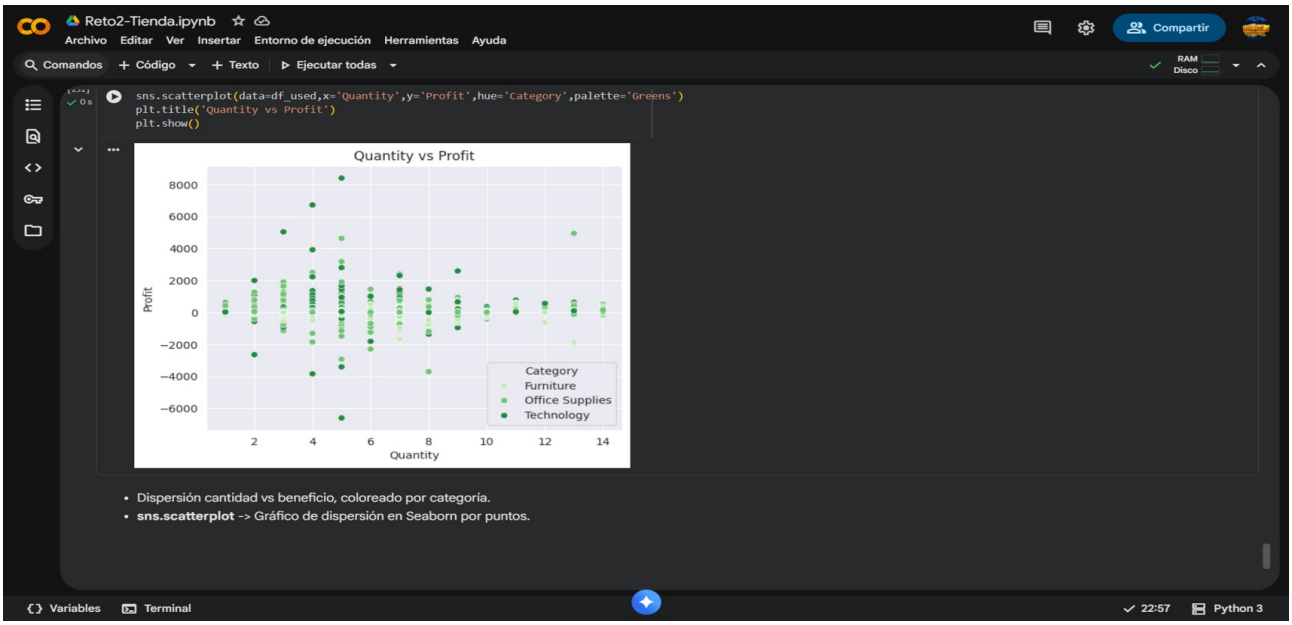
RAM Disco

[230] ✓ 1 s

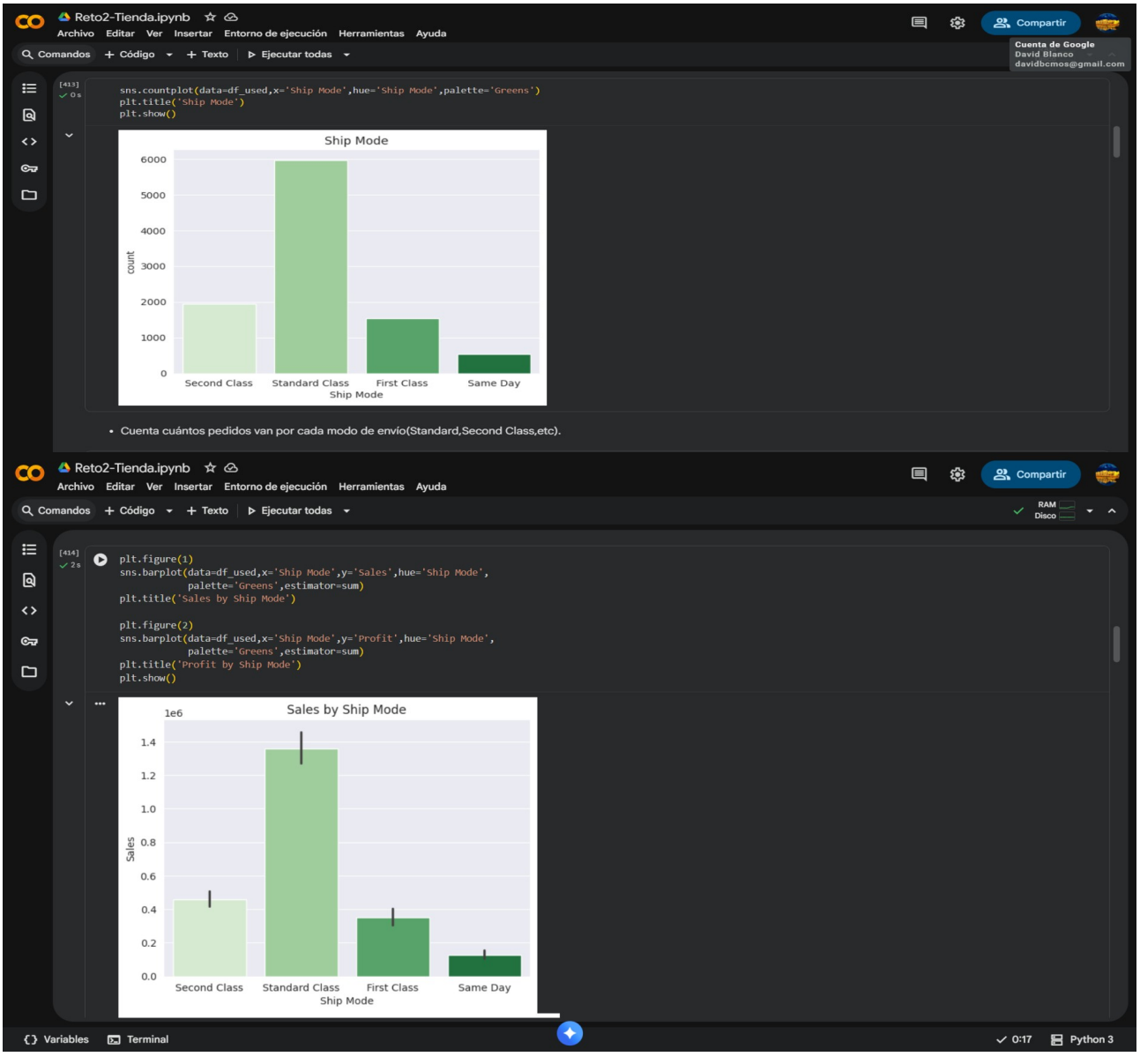
```
sns.barplot(data=df_used,x='Segment',y='Sales',
            hue='Region',palette='Greens',estimator=sum)
plt.title('Sales by Segment')
plt.show()
```



- Similar pero separa por región.



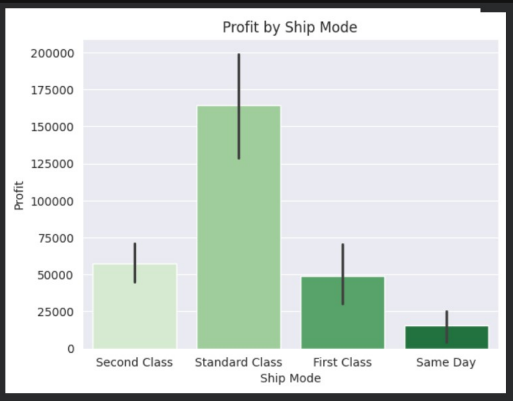
9. SHIP MODE, DISTRIBUCIONES Y CORRELACIÓN



Reto2-Tienda.ipynb

Archivo Editar Ver Insertar Entorno de ejecución Herramientas Ayuda

Comandos + Código + Texto ▷ Ejecutar todas



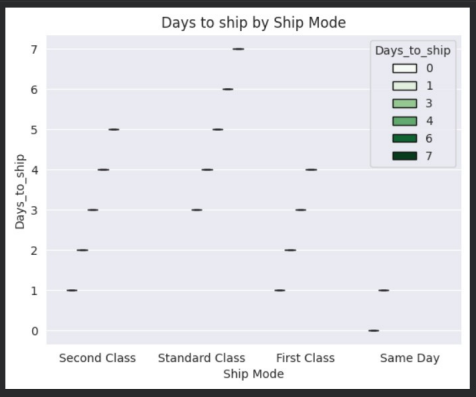
- **plt.figure(1)** -> Gráfico 1: ventas por modo de envío.
- **plt.figure(2)** -> Gráfico 2: beneficios por modo de envío.

Reto2-Tienda.ipynb

Archivo Editar Ver Insertar Entorno de ejecución Herramientas Ayuda

Comandos + Código + Texto ▷ Ejecutar todas

[415]
```
sns.boxplot(data=df_used,x='Ship Mode',y='Days_to_ship',hue='Days_to_ship',palette='Greens')
plt.title('Days to ship by Ship Mode')
plt.show()
```



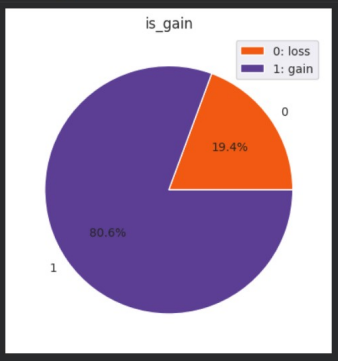
- **sns.boxplot** -> Gráfico de cajas de Seaborn.
- Boxplot del nº de días para enviar según el modo de envío.

Reto2-Tienda.ipynb

Archivo Editar Ver Insertar Entorno de ejecución Herramientas Ayuda

Comandos + Código + Texto ▷ Ejecutar todas

[416]
```
count_gain=df_used.groupby('is_gain').size().reset_index(name='count')
plt.pie(count_gain['count'],labels=count_gain['is_gain'],
        autopct='%1.1f%%',colors=['#F25912','#5C3E94'])
plt.title('is_gain')
plt.legend(['0: loss','1: gain'])
plt.show()
```



- **df_used.groupby('is_gain')** -> Agrupa el dataframe por la columna is_gain.
- **is_gain** tiene valores:
  - 1 -> el pedido generó beneficio.
  - 0 -> el pedido generó pérdida.

Variables Terminal 0:17 Python 3

Reto2-Tienda.ipynb

Archivo Editar Ver Insertar Entorno de ejecución Herramientas Ayuda

Compartir

Comandos + Código + Texto Ejecutar todas

RAM Disco

- **groupby** crea dos grupos:
  - Grupo 0: todas las filas donde **is_gain=0**.
  - Grupo 1:todas las filas donde **is_gain=1**.
- **size()** -> Cuenta cuántas filas hay en cada grupo.
- **.reset_index(name='count')** -> Convierte ese dataframe ordenado con dos columnas y se guarda en la variable count_gain.
- **count_gain['count']** -> Son los nº que va a usar para calcular el tamaño de la pòrción en el gráfico de pastel.
- **labels=count_gain['is_gain']** -> Etiquetas en el pastel(**0 ó 1**).
- **plt.legend(['0: loss','1: gain'])** -> Añade leyenda manual para que se explique mejor.

Reto2-Tienda.ipynb

Archivo Editar Ver Insertar Entorno de ejecución Herramientas Ayuda

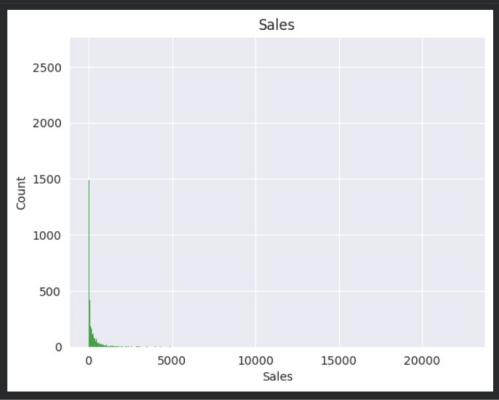
Compartir

Comandos + Código + Texto Ejecutar todas

RAM Disco

[417] ✓ 1s

```
sns.histplot(data=df_used,x='Sales',color='green')
plt.title('Sales')
plt.show()
```



- **sns.histplot** -> Histograma de Seaborn.
- Crea un histograma de ventas.

Reto2-Tienda.ipynb

Archivo Editar Ver Insertar Entorno de ejecución Herramientas Ayuda

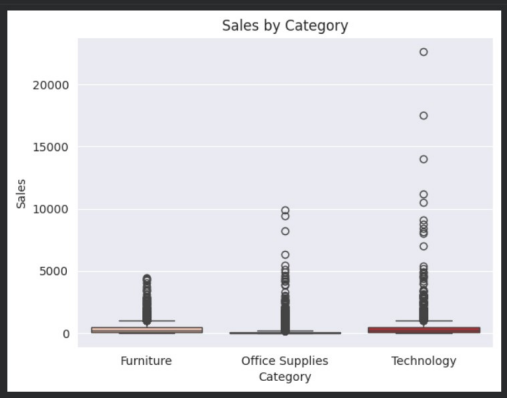
Compartir

Comandos + Código + Texto Ejecutar todas

RAM Disco

[418] ✓ 0s

```
sns.boxplot(data=df_used,x='Category',y='Sales',hue='Category',palette='Reds')
plt.title('Sales by Category')
plt.show()
```



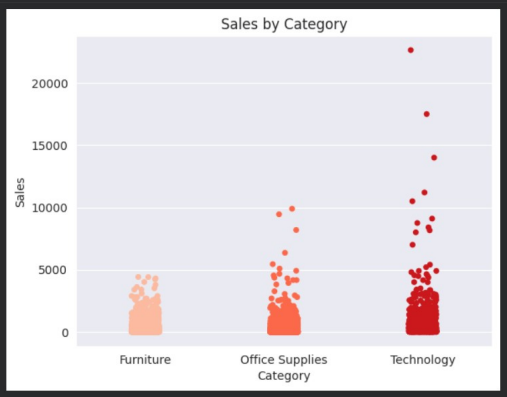
- Boxplot de ventas por categoría.

Reto2-Tienda.ipynb

Archivo Editar Ver Insertar Entorno de ejecución Herramientas Ayuda

Comandos + Código + Texto Ejecutar todas

```
[419]
sns.stripplot(data=df_used,x='Category',y='Sales',hue='Category',palette='Reds')
plt.title('Sales by Category')
plt.show()
```



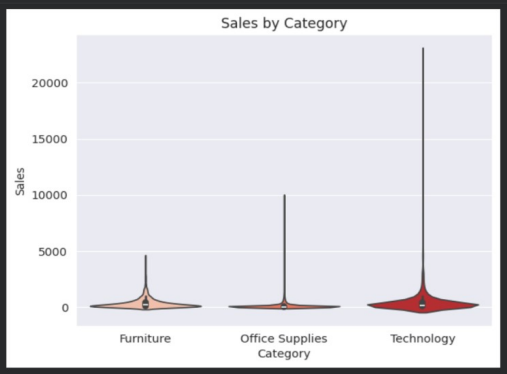
- **sns.stripplot** -> Gráfico de puntos donde cada punto representa una fila del dataset.

Reto2-Tienda.ipynb

Archivo Editar Ver Insertar Entorno de ejecución Herramientas Ayuda

Comandos + Código + Texto Ejecutar todas

```
[420]
sns.violinplot(data=df_used,x='Category',y='Sales',hue='Category',palette='Reds')
plt.title('Sales by Category')
plt.show()
```



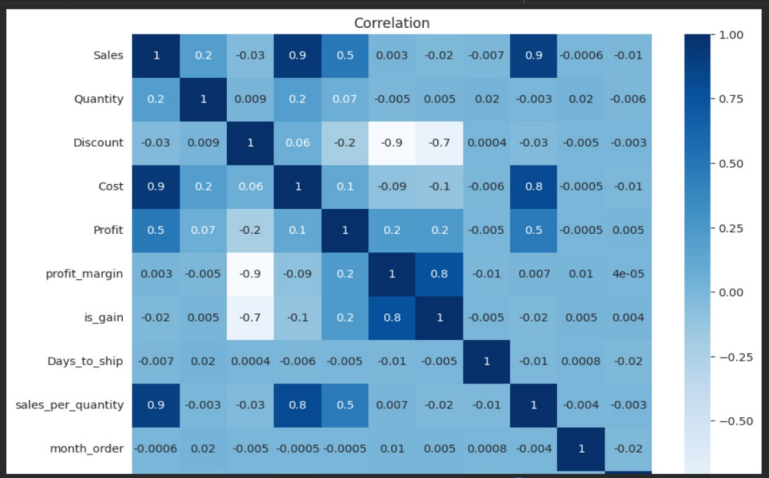
- **sns.violinplot** -> Gráfico que muestra la forma de distribución(igual a un histograma), pero vertical y suavizado.

Reto2-Tienda.ipynb

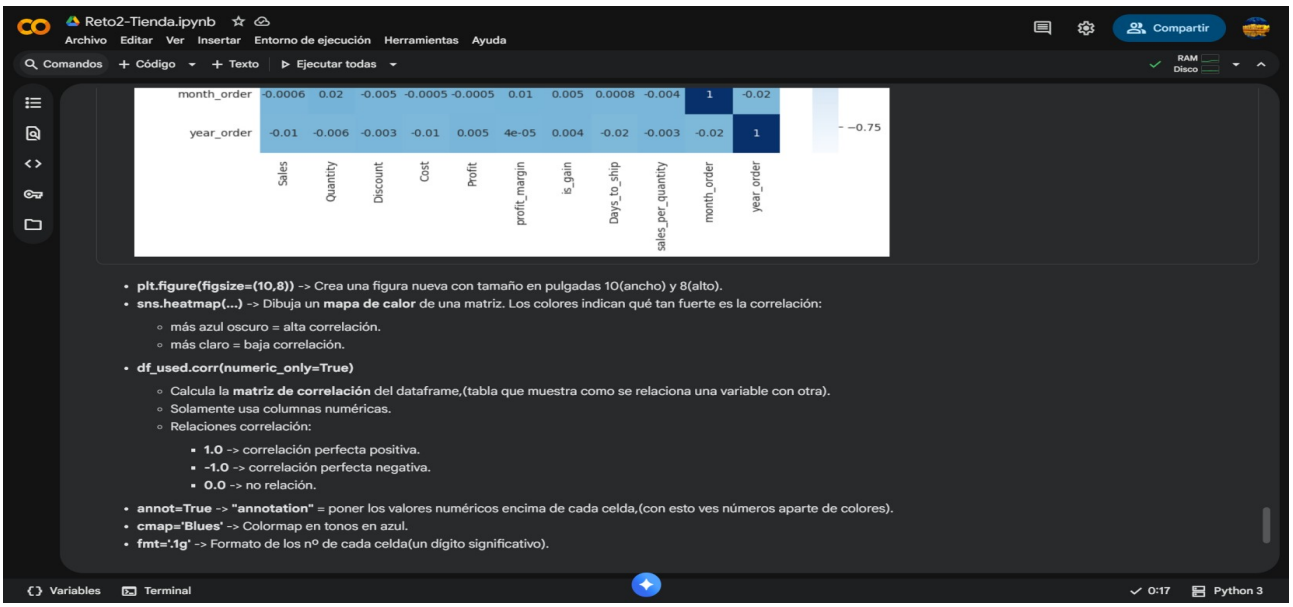
Archivo Editar Ver Insertar Entorno de ejecución Herramientas Ayuda

Comandos + Código + Texto Ejecutar todas

```
[421]
plt.figure(figsize=(10,8))
sns.heatmap(df_used.corr(numeric_only=True),annot=True,cmap='Blues',fmt='.1g')
plt.title('Correlation')
plt.show()
```

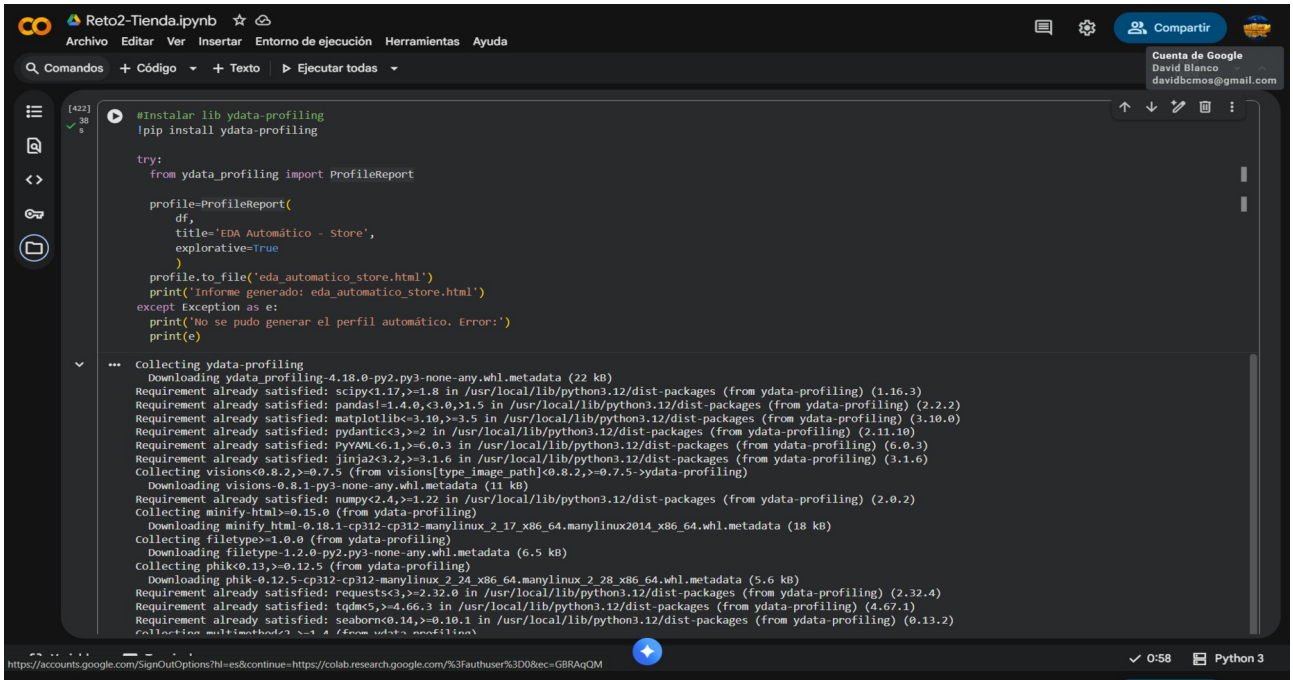


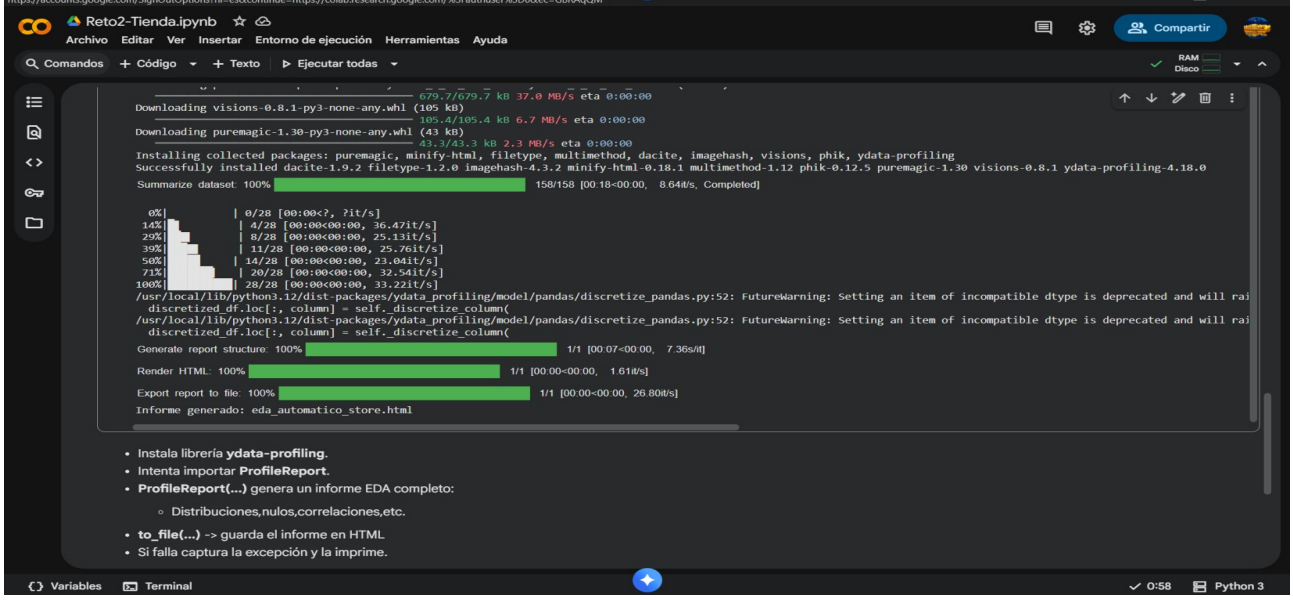
Variables Terminal 0:17 Python 3



- **plt.figure(figsize=(10,8))** -> Crea una figura nueva con tamaño en pulgadas 10(ancho) y 8(alto).
- **sns.heatmap(...)** -> Dibuja un **mapa de calor** de una matriz. Los colores indican qué tan fuerte es la correlación:
  - más azul oscuro = alta correlación.
  - más claro = baja correlación.
- **df_used.corr(numeric_only=True)**
  - Calcula la **matriz de correlación** del dataframe,(tabla que muestra como se relaciona una variable con otra).
  - Solamente usa columnas numéricas.
  - Relaciones correlación:
    - **1.0** -> correlación perfecta positiva.
    - **-1.0** -> correlación perfecta negativa.
    - **0.0** -> no relación.
- **annot=True** -> **"annotation"** = poner los valores numéricos encima de cada celda,(con esto ves números aparte de colores).
- **cmap='Blues'** -> Colormap en tonos en azul.
- **fmt='.1g'** -> Formato de los nº de cada celda(un dígito significativo).

10. EDA AUTOMÁTICO (YDATA-PROFILING)

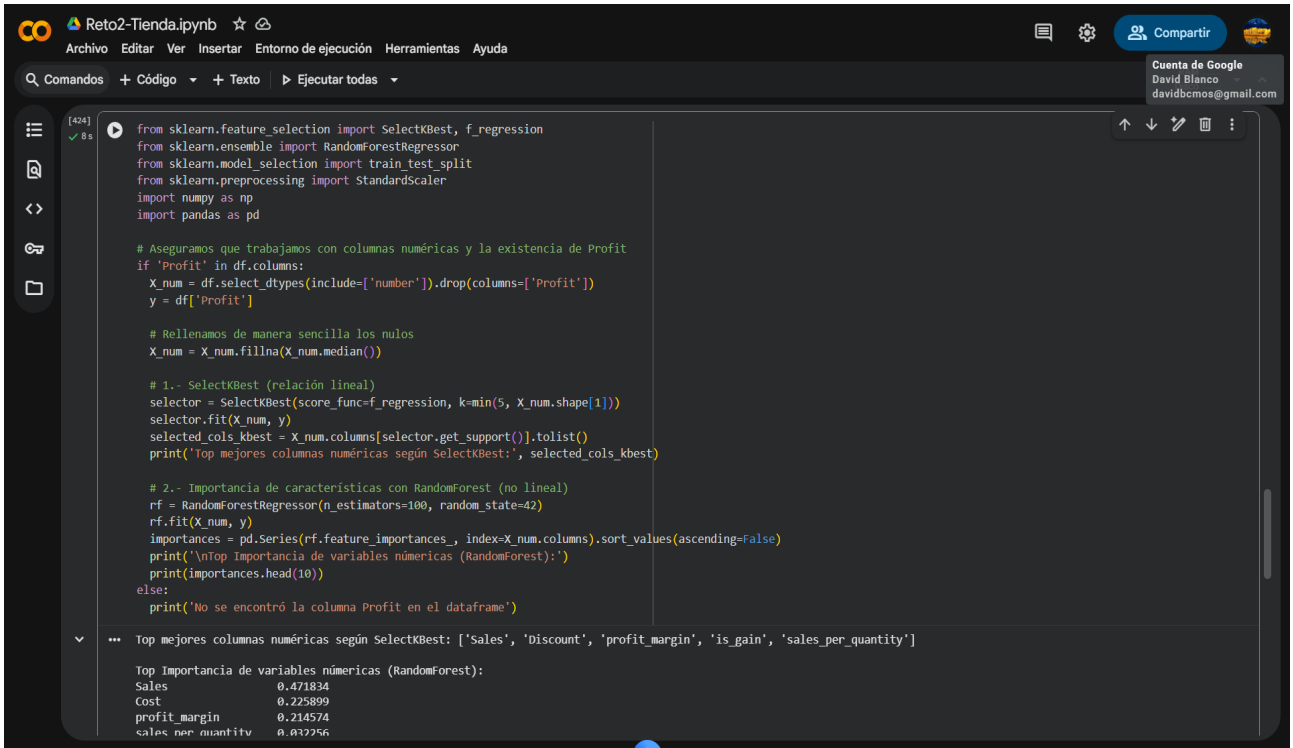




- Instala librería **ydata-profiling**.
- Intenta importar **ProfileReport**.
- **ProfileReport(...)** genera un informe EDA completo:
  - Distribuciones,nulos,correlaciones,etc.
- **to_file(...)** -> guarda el informe en HTML
- Si falla captura la excepción y la imprime.

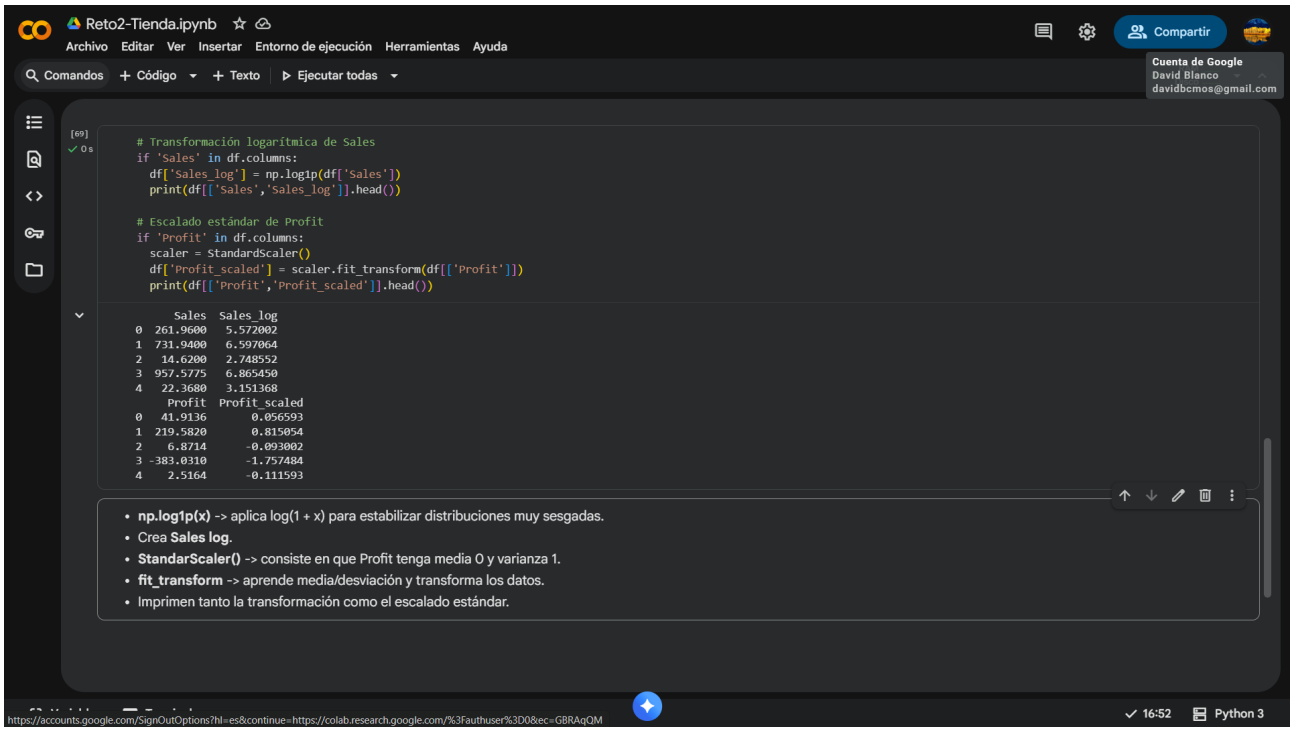
11. INGENIERÍA DE CARACTERÍSTICAS Y PREPARACIÓN PARA ML (SCIKIT-LEARN)

11.1. Selección de atributos

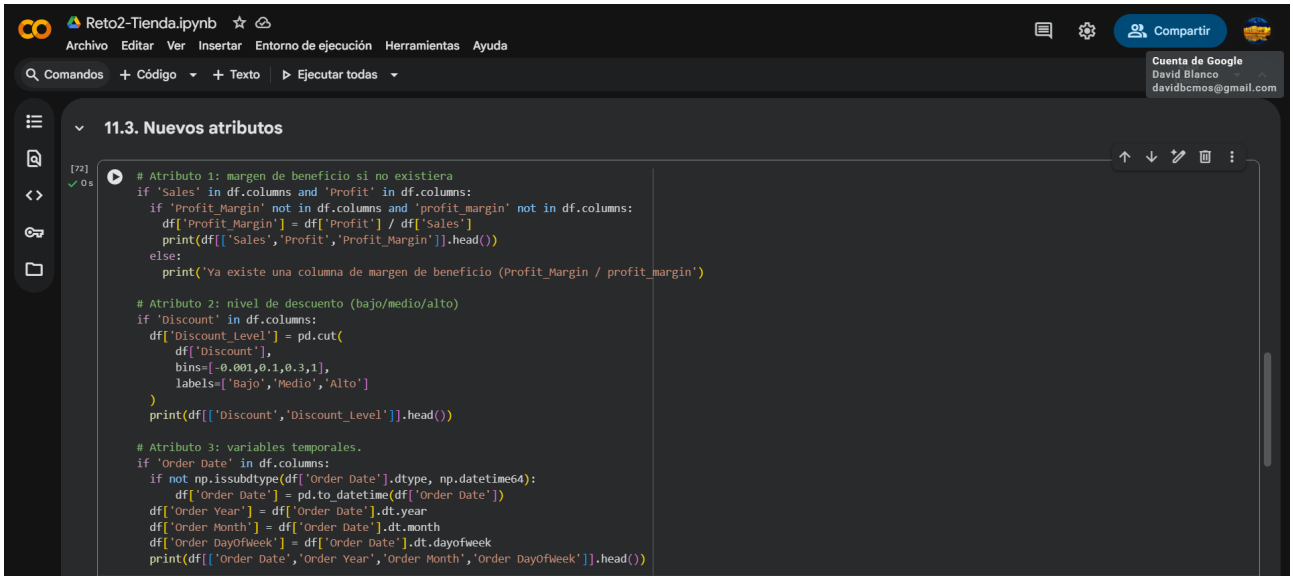


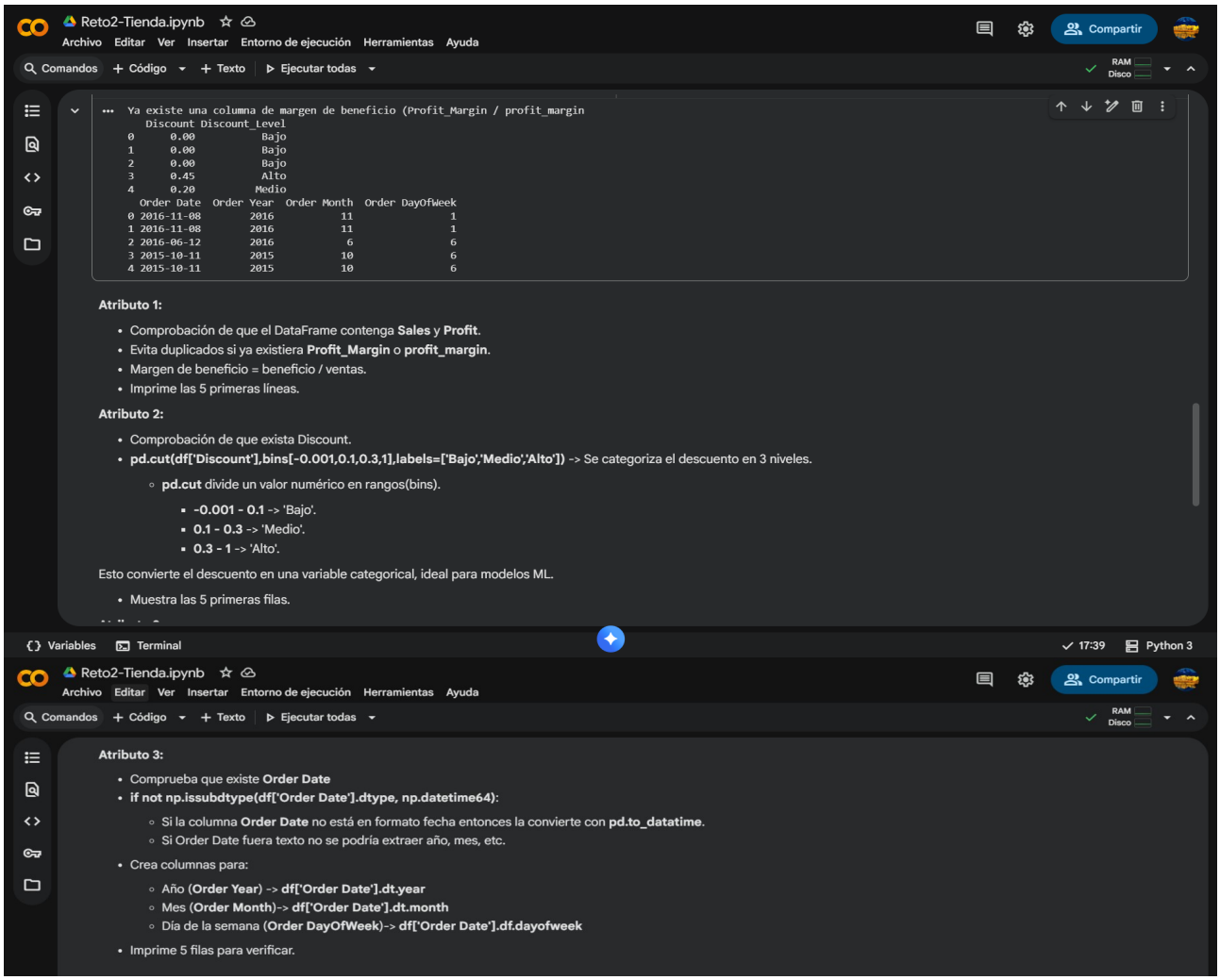
- Importa de **scikit-learn(sklearn)**
  - **SelectKBest, f_regression** -> selección de variables basada en relación lineal con la **y**.
  - **RandomForestRegressor** -> modelo de bosque aleatorio para estimar la importancia de variables.
  - **train_test_split** -> dividir en train/test.
  - **StandardScaler** -> escalado estándar.
- Comprueba que **Profit** existe.
- **X_num = df.select_dtypes(includeÇ=['number']).drop(columns=['Profit'])** -> Toma solo columnas numéricas y quita Profit(objetivo).
- **y = df['Profit']** -> variable objetivo.
- **X_num = X_num.fillna(X_num.median())** -> rellena nulos con la mediana.
- **SelectKBest(...)**:
  - **score_func=f_regression** -> usa F-test de regresión.
  - **k=min(5,X_num.shape[1])** -> selecciona hasta 5 mejores columnas.
- **selector.fit(X_num, y)** -> ajusta la selección al dato.
- **selector.get_support()** -> máscara booleana de columnas seleccionadas.
- **RandomForestRegressor(...)** -> modelo no lineal para ver importancias.
- **rf.fit(X_num, y)** -> entrena el modelo.
- **rf.feature_importances_** -> peso de cada variable.
- **pd.Series(...)sort_values(...)** -> ordena importancias.

11.2. Transformaciones (log y escalado)

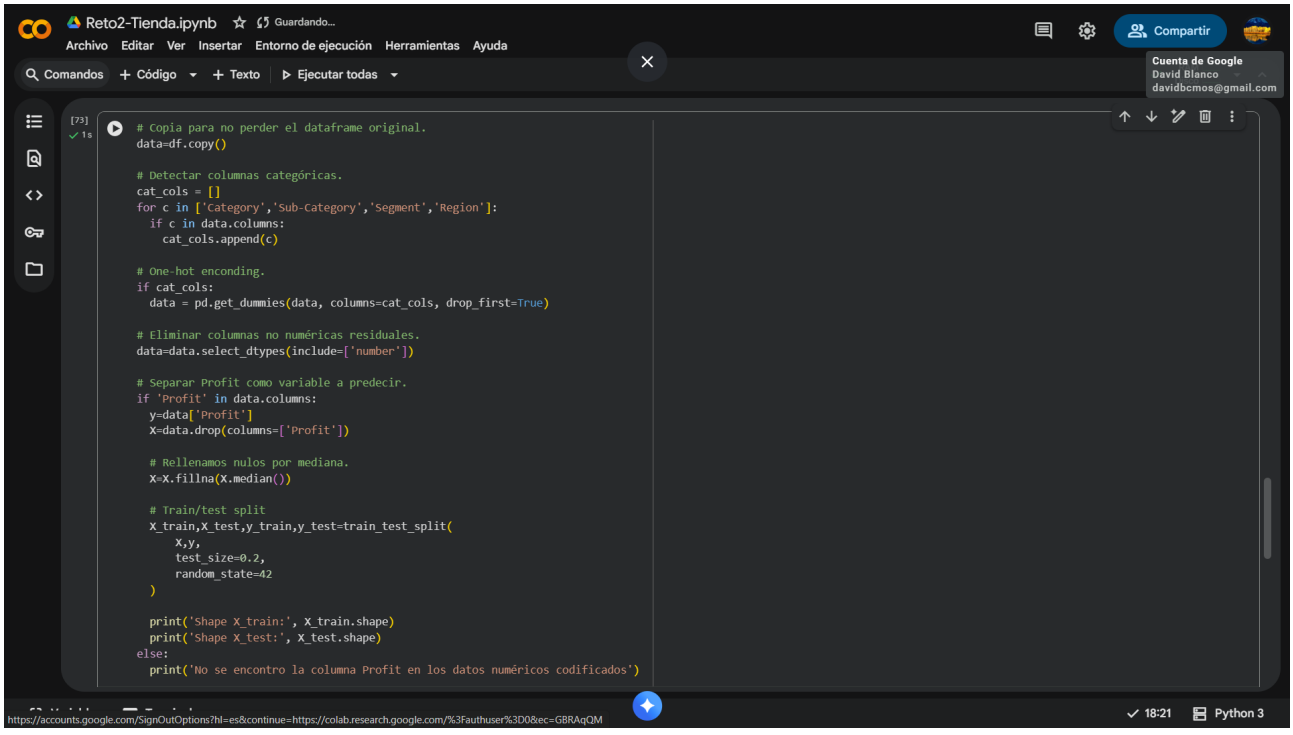


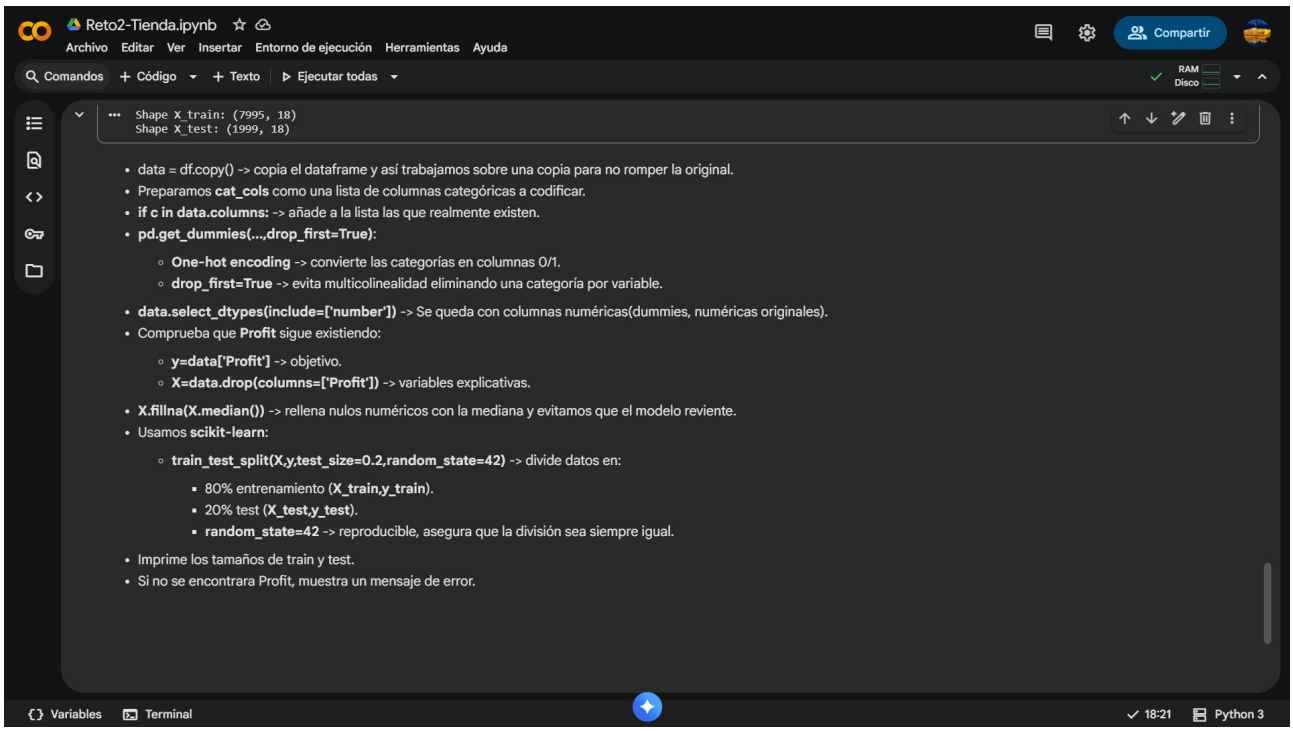
11.3. Nuevos atributos





11.4. Dataset listo para ML con one-hot + train_test_split





- data = df.copy() -> copia el dataframe y así trabajamos sobre una copia para no romper la original.
- Preparamos **cat_cols** como una lista de columnas categóricas a codificar.
- **if c in data.columns:** -> añade a la lista las que realmente existen.
- **pd.get_dummies(...,drop_first=True)**:
  - **One-hot encoding** -> convierte las categorías en columnas 0/1.
  - **drop_first=True** -> evita multicolinealidad eliminando una categoría por variable.
- **data.select_dtypes(include=['number'])** -> Se queda con columnas numéricas(dummies, numéricas originales).
- Comprueba que **Profit** sigue existiendo:
  - **y=data['Profit']** -> objetivo.
  - **X=data.drop(columns=['Profit'])** -> variables explicativas.
- **X.fillna(X.median())** -> rellena nulos numéricos con la mediana y evitamos que el modelo reviente.
- Usamos **scikit-learn**:
  - **train_test_split(X,y,test_size=0.2,random_state=42)** -> divide datos en:
    - 80% entrenamiento (**X_train,y_train**).
    - 20% test (**X_test,y_test**).
    - **random_state=42** -> reproducible, asegura que la división sea siempre igual.
- Imprime los tamaños de train y test.
- Si no se encontrara Profit, muestra un mensaje de error.