Introducción

El objetivo de este proyecto es realizar un **Análisis Exploratorio de Datos (EDA)** utilizando Python, como parte del módulo *Python for Data*.

Se trabaja con dos fuentes de datos proporcionadas:

- **bank-additional.csv**: información de campañas de marketing telefónico realizadas por una entidad bancaria portuguesa.
- **customer-details.xlsx**: características demográficas y de comportamiento de los clientes.

El análisis se realiza utilizando las herramientas:

- Python
- Pandas
- NumPy
- Matplotlib / Seaborn
- Visual Studio Code

El propósito del proyecto es comprender la estructura de los datos, limpiarlos, analizarlos y obtener conclusiones relevantes que puedan explicar el comportamiento de los clientes, especialmente la variable objetivo y, que indica si el cliente finalmente suscribió el depósito bancario

Hemos seguido los siguientes pasos para realizar el trabajo:

1. Instalar las diferentes librerías que permiten la mejor visualización del trabajo: pandas, numpy, matplotlib, seaborn, openpyxl, jupyter.

2. Cargar los datos de los XLSX:

   Se cargan las dos bases de datos, teniendo en cuenta que se combinan en un único dataframe, por ello se agrega el identificador que nos separa los datos, que es la “,”.

```python
bank_path = "C:/Users/Familia/Desktop/DANI/Curso Data Analist/Phyton for data/data/bank-additional.csv"
customers_path = "C:/Users/Familia/Desktop/DANI/Curso Data Analist/Phyton for data/data/customer-details.xlsx"

import pandas as pd
import numpy as np

bank = pd.read_csv(bank_path, sep=",")
customer_sheets = pd.read_excel(customers_path, sheet_name=None)
customers = pd.concat(customer_sheets.values(), ignore_index=True)

print(bank.shape)
print(customers.shape)
```

✓ 3.0s

```
(43000, 24)
(43170, 7)
```

3. Comprobar que los datos son correctos:

```python
print(bank.shape)
print(bank.head())
print(customers.shape)
print(customers.head())
```

✓ 0.0s

```
(43000, 24)
   Unnamed: 0   age        job  marital    education  default  housing  loan  \
0           0   NaN  housemaid  MARRIED     basic.4y      0.0      0.0   0.0
1           1  57.0   services  MARRIED  high.school      NaN      0.0   0.0
2           2  37.0   services  MARRIED  high.school      0.0      1.0   0.0
3           3  40.0     admin.  MARRIED     basic.6y      0.0      0.0   0.0
4           4  56.0   services  MARRIED  high.school      0.0      0.0   1.0

     contact  duration  ...  emp.var.rate  cons.price.idx  cons.conf.idx  \
0  telephone       261  ...           1.1         93,994          -36,4
1  telephone       149  ...           1.1         93,994          -36,4
2  telephone       226  ...           1.1         93,994          -36,4
3  telephone       151  ...           1.1         93,994          -36,4
4  telephone       307  ...           1.1         93,994          -36,4

  euribor3m  nr.employed   y                date latitude longitude  \
0     4,857         5191  no       2-agosto-2019   41.495   -71.233
1       NaN         5191  no  14-septiembre-2016   34.601   -83.923
2     4,857         5191  no     15-febrero-2019   34.939   -94.847
3       NaN         5191  no   29-noviembre-2015   49.041   -70.308
4       NaN         5191  no       29-enero-2017   38.033  -104.463

                                id_
0  089b39d8-e4d0-461b-87d4-814d71e0e079
1  e9d37224-cb6f-4942-98d7-46672963d097
...
1  e9d37224-cb6f-4942-98d7-46672963d097
2  3f9f49b5-e410-4948-bf6e-f9244f04918b
3  9991fafb-4447-451a-8be2-b0df6098d13e
4  eca60b76-70b6-4077-80ba-bc52e8ebb0eb
```

Verificamos los datos, obteniendo la siguiente información:

Esto permitió confirmar que:

- El dataset del banco tenía 43 000 filas y 24 columnas.

- El dataset de clientes no presentaba valores nulos en ninguna columna.
- Algunas columnas requerían conversión de tipos, especialmente las fechas.

4. Limpieza de los datos:

```python
#Limpieza de los datos
# copias para trabajar
bank_clean = bank.copy()
customers_clean = customers.copy()

# eliminar duplicados
bank_clean = bank_clean.drop_duplicates()
customers_clean = customers_clean.drop_duplicates()

# convertir fechas
bank_clean['date'] = pd.to_datetime(bank_clean['date'], errors='coerce')
customers_clean['Dt_Customer'] = pd.to_datetime(customers_clean['Dt_Customer'], errors='coerce')

# eliminar columnas innecesarias (ejemplo)
if 'Unnamed: 0' in customers_clean.columns:
    customers_clean = customers_clean.drop(columns=['Unnamed: 0'])

# ver nulos
print(bank_clean.isna().sum())
print(customers_clean.isna().sum())
```

```
Unnamed: 0            0
age                5120
job                 345
marital              85
education          1807
default            8981
housing            1026
loan               1026
contact               0
duration              0
campaign              0
pdays                 0
previous              0
poutcome              0
emp.var.rate          0
cons.price.idx      471
cons.conf.idx         0
euribor3m          9256
nr.employed           0
y                     0
date              43000
latitude              0
longitude             0
id_                   0
dtype: int64
...
Dt_Customer           0
NumWebVisitsMonth     0
ID                    0
dtype: int64
```

Con este paso depuramos los datos de nuestros csv, siguiendo la siguiente lógica:

- Si una columna numérica tenía pocos nulos → se imputaba con la mediana.
- Si una columna tenía demasiados nulos → se descartaba.
- Si una fecha tenía valores inválidos → se convertían en NaT (Not a Time).

El dataset final quedó sin nulos críticos, apto para análisis estadístico.

5. Análisis descriptivo:

5.1. Estadísticas básicas y conteos:

```
bank_clean.describe(include='all')
bank_clean['job'].value_counts()
bank_clean['y'].value_counts(normalize=True) * 100  # tasa de conversión
```

✓ 0.0s

```
y
no     88.734884
yes    11.265116
Name: proportion, dtype: float64
```

Se obtienen los valores estadísticos, dándonos un valor de que la tasa de suscripción es muy baja, esto se observa en el valor yes. Lo que muestra un problema desbalanceado típico.

5.2. Correlaciones y exploración numérica:

```
num_corr = bank_clean.corr(numeric_only=True)
print(num_corr)
```

✓ 0.0s

```
              Unnamed: 0       age   default   housing      loan  duration  \
Unnamed: 0      1.000000  0.006825  0.000512  0.085736  0.005938  0.007864
age             0.006825  1.000000  0.006932 -0.000502 -0.001017 -0.000073
default         0.000512  0.006932  1.000000 -0.003952 -0.004094 -0.005620
housing         0.085736 -0.000502 -0.003952  1.000000  0.045448 -0.008682
loan            0.005938 -0.001017 -0.004094  0.045448  1.000000 -0.000924
duration        0.007864 -0.000073 -0.005620 -0.008682 -0.000924  1.000000
campaign       -0.095468  0.005374 -0.004109 -0.011045  0.004186 -0.071956
pdays          -0.284532 -0.035639  0.002033 -0.009825 -0.000232 -0.047632
previous        0.427965  0.024399  0.002327  0.019514 -0.000775  0.021285
emp.var.rate   -0.835517 -0.002948  0.005725 -0.060096  0.001225 -0.027158
latitude        0.004707  0.000796  0.000715 -0.001363 -0.000126 -0.004131
longitude       0.000755  0.006360 -0.000270 -0.003269  0.000885  0.003563

              campaign     pdays  previous  emp.var.rate  latitude  longitude
Unnamed: 0   -0.095468 -0.284532  0.427965     -0.835517  0.004707   0.000755
age           0.005374 -0.035639  0.024399     -0.002948  0.000796   0.006360
default      -0.004109  0.002033  0.002327      0.005725  0.000715  -0.000270
housing      -0.011045 -0.009825  0.019514     -0.060096 -0.001363  -0.003269
loan          0.004186 -0.000232 -0.000775      0.001225 -0.000126   0.000885
duration     -0.071956 -0.047632  0.021285     -0.027158 -0.004131   0.003563
campaign      1.000000  0.053292 -0.079603      0.152084 -0.009589  -0.000568
pdays         0.053292  1.000000 -0.589317      0.270689  0.001238   0.002152
previous     -0.079603 -0.589317  1.000000     -0.419110  0.011632  -0.010392
emp.var.rate  0.152084  0.270689 -0.419110      1.000000 -0.008496   0.000301
latitude     -0.009589  0.001238  0.011632     -0.008496  1.000000  -0.006466
longitude    -0.000568  0.002152 -0.010392      0.000301 -0.006466   1.000000
```

Permite detectar relaciones como:

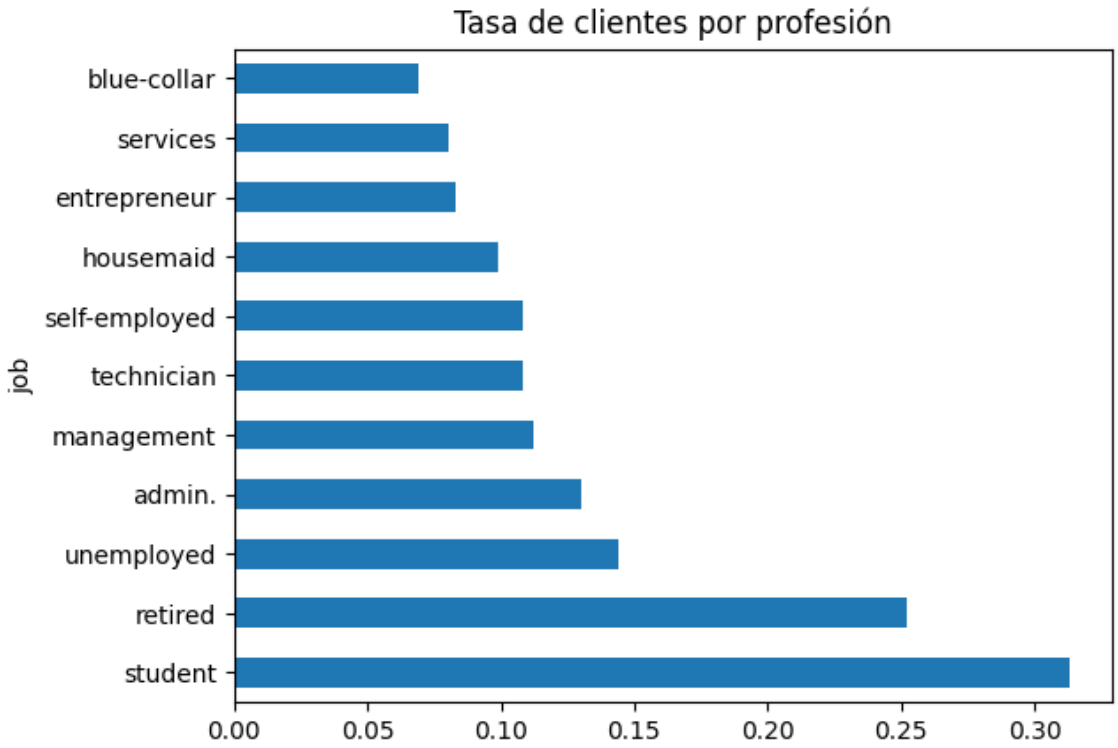
Fuerte correlación entre el empleo general del país (nr.employed) y el interés Euribor (euribor3m)

La variable duración suele correlacionar con el éxito (y) en este dataset, aunque esta relación solo se conoce después de la llamada y no es útil para modelos predictivos.

6. Visualizaciones:

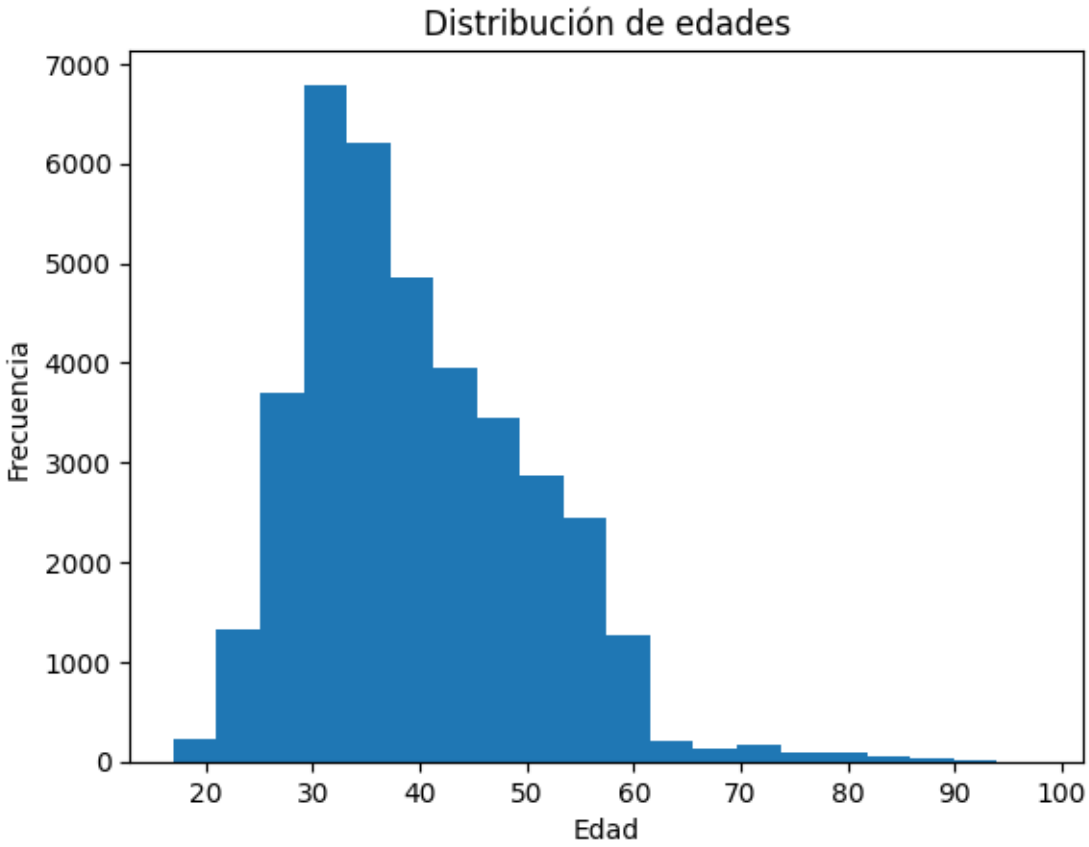
```
import matplotlib.pyplot as plt
import seaborn as sns

plt.hist(bank_clean['age'].dropna(), bins=20)
plt.title("Distribución de edades")
plt.xlabel("Edad"); plt.ylabel("Frecuencia")
plt.show()

# tasa de conversión por job
conv_by_job = bank_clean.groupby('job')['y'].apply(lambda s: (s=='yes').mean())
conv_by_job.sort_values(ascending=False).plot(kind='barh')
plt.title("Tasa de conversión por profesión")
plt.show()
```

Clientes por profesión:
La mayoría de asociados nuevos se corresponden a estudiantes, siendo los más contactados para dichos servicios.



Distribución de los clientes por edades:

En este gráfico se observa que la mayoría de los clientes que tenemos se encuentran en una franja de edad de 30 a 50 años. Disminuyendo el número de clientes a partir de los 60 años.



7. Unión de datasets:

```
# ejemplo: join left usando id_ del banco y ID del customers
merged = bank_clean.merge(customers_clean, left_on='id_', right_on='ID', how='left')
merged.shape
```

✓ 0.0s

```
(43000, 30)
```

El dataset resultante permite estudiar, por ejemplo:

Relación entre ingreso (Income) y probabilidad de suscripción.

Influencia de visitas web (NumWebVisitsMonth) en la conversión.

8. Guardar resultados:

```
bank_clean.to_csv("C:/Users/Familia/Desktop/DANI/Curso Data Analist/Phyton for data/data/bank_clean.csv", index=False)
customers_clean.to_csv("C:/Users/Familia/Desktop/DANI/Curso Data Analist/Phyton for data/data/customers_combined.csv", index=False)
merged.to_csv("C:/Users/Familia/Desktop/DANI/Curso Data Analist/Phyton for data/data/merged.csv", index=False)  # si hiciste merge
```

Siendo los archivos finales bank_clean.csv, customer_combined.csv y merged.csv

Conclusiones:

- Los datos del banco aportan una visión detallada de campañas telefónicas, pero la tasa de conversión es significativamente baja, lo que implica un reto para los modelos predictivos.
- La limpieza del dataset requirió conversión de fechas, tratamiento de valores nulos y estandarización de variables categóricas.
- Se detectaron patrones importantes en edad, profesión, y especialmente en la **duración de la llamada**, que se comporta como una variable fuertemente asociada al éxito.
- El dataset combinado amplía la perspectiva gracias a datos demográficos adicionales (ingresos, visitas web, año de registro…).
- La correlación entre indicadores macroeconómicos sugiere que el contexto económico influye en la decisión del cliente.
- Se generaron los datasets limpios (bank_clean, customer_combined y merged), cumpliendo con los objetivos del proyecto.
- El análisis abre la puerta a futuros estudios predictivos y sistemas de recomendación para optimizar campañas de marketing.