# Count the number of unique values in each column

# Used to understand the dataset better

print("Gender:")

print(df['gender'].value\_counts(dropna=False))

Este bloque me muestra por pantalla los diferentes valores de Gender junto a su conteo. De esta manera puedo ver si hay valores repetidos con fallos tipográficos, nulos o incoherentes.

# Changes the data type to category, which is more efficient for categorical data

df["gender"] = df["gender"].astype("category")

print(df["gender"].dtype)

Le cambio el tipo a Categoria, que básicamente hace:  
  
1- crea una referencia (Ej: Male = 1, Female = 2, Other = 3)

2- cambia los valores string por estos números

3- el usuario sigue viendo las strings, pero realmente en la base de datos ahora son números. Ha creado una “tabla de dimensiones” y accede a ella con los números para presentar las strings.

Este paso es bomba porque me ahorra velocidad de computación desde ya y tiempo en el futuro.

# --- Null management ---

# This part of the code is used to manage null values in the dataset.

# gender\_dist will store the normalized distribution

gender\_dist = df["gender"].value\_counts(normalize=True)

gender\_dist es un pandas.Series, no una tupla, y en él se almacenan los diferentes valores de Gender junto al conteo normalizado de cada uno. Esto es: cuenta cuantos hay de cada y divide cada suma entre el total (los normaliza)

# mask will be used to locate the null values in the dataset

mask = df["gender"].isnull()

mask almacena 2 cosas:  
  
1- true si es nulo, false si no lo es

2- la posición de cada elemento

O sea, que mask es una tabla de 2 columnas con true’s y false’s que corresponden a mis registros. Con mask voy a poder encontrar los nulos fácilmente, ya que tienen true

# Adds all the null

n\_nulls = mask.sum()

# Fill the null values with random choices based on the distribution

# This will ensure that the null values are filled in a way that reflects the original distribution

df.loc[mask, "gender"] = np.random.choice(

    gender\_dist.index,

    size=n\_nulls,

    p=gender\_dist.values

)

random.choice toma los parámetros:  
1- gender\_dist.index (Male, Female, Other)

2- size = n\_nulls => cuántos valores random se van a generar

3- p=gender\_dist.values (p = probability) => va a mantener la distribución normalizada anteriormente. O sea, los valores nulos se van a rellenar con los Gender proporcionalmente  
  
.loc[mask, “Gender”]:  
1- pone las tablas (mask y Gender) parejas, una junto a la otra, y realiza el random.choice en aquellos valores de Gender que coincidan con un True de mask (aquellos valores que sean nulos)