Análisis y visualización de datos - Diez Lorenzo Tomas

API - 2

Situación

- Una vez que se entregó la primera parte del informe, continuás con la segunda parte, enfocada en los aspectos demográficos y sociales. Los datos para esta sección no están completamente listos para analizar, necesitan preprocesamiento antes de poder construir los gráficos o tablas. Como sabés, algunas de estas tareas son más prácticas y eficientes haciéndolas en Python que manualmente (en Excel, por ejemplo).

Requerimientos

- Para realizar esta actividad descargá el archivo ZIP, que encontrarás en la plataforma al final de este documento. Importante: Al momento de importar los archivos .csv, usando pd.read_csv(), usar la opción encoding = "latin-1 " para que importe correctamente los nombres de las provincias con tildes.

Consignas

- Como próximo paso, decidís integrar distintas bases de datos que tenés disponibles. Estas contienen variables demográficas: población, hogares y viviendas, esperanza de vida y fecundidad.

1. En Jupyter, creá un nuevo notebook e importó las librerías necesarias, y luego la base de proyecciones de población por año (en formato .csv) y las otras (ej. expectativa de vida, fecundidad). Tené en cuenta que algunas bases contienen datos de varios años y otras tienen únicamente el año del censo 2010. Hacé los chequeos básicos (head, describe, etc.).
2. Calculá un campo nuevo, densidad (población/superficie) y usar la función descrita sobre ese campo nuevo.
3. Identificá si existe algún valor extremo en la densidad de población y explicó a qué podría deberse esto.

Resolucion

0. Setup

In [ ]:
```
import pandas as pd
import numpy as np
import os
```

```
def getCsvPath(fileName):
  return f"{os.getcwd()}/api-2-archivos/ARCHIVOS/{fileName}.csv"

test_path = getCsvPath("test_path")
test_path
```

Out[ ]:
```
'/Users/lorenzotomasdiez/study/teclab/Analisis y Visualizacion de datos/
api-2/api-2-archivos/ARCHIVOS/test_path.csv'
```

1. Importamos los csv con los que vamos a trabajar y la codificacion correcta.

In [ ]:
```
poblacion_dir = getCsvPath("poblacion")
hogares_dir = getCsvPath("hogares_viviendas_superficie")
esperanza_dir = getCsvPath("esperanza_de_vida")

df_poblacion = pd.read_csv(poblacion_dir, encoding="latin-1")
df_hogares = pd.read_csv(hogares_dir, encoding="latin-1")
df_esperanza = pd.read_csv(esperanza_dir, encoding="latin-1")
```

2. Verificamos el contenido de la tabla poblacion.

In [ ]:
```
df_poblacion.head()
```

Out[ ]:

	provincia	anio	poblacion_total	poblacion_varones	poblacion_mujeres
0	Total País	2010	40788453	19940704	20847749
1	Total País	2011	41261490	20180791	21080699
2	Total País	2012	41733271	20420391	21312880
3	Total País	2013	42202935	20659037	21543898
4	Total País	2014	42669500	20896203	21773297

In [ ]:
```
df_poblacion.describe()
```

Out[ ]:

	anio	poblacion_total	poblacion_varones	poblacion_mujeres
count	775.000000	7.750000e+02	7.750000e+02	7.750000e+02
mean	2025.000000	3.777746e+06	1.856888e+06	1.920858e+06
std	8.950048	9.560571e+06	4.699604e+06	4.861043e+06
min	2010.000000	1.316610e+05	6.723500e+04	6.442600e+04
25%	2017.000000	5.845510e+05	2.906740e+05	2.934905e+05
50%	2025.000000	1.017731e+06	5.061010e+05	5.161370e+05
75%	2033.000000	1.855285e+06	9.138865e+05	9.404745e+05
max	2040.000000	5.277848e+07	2.603809e+07	2.674038e+07

In [ ]:
```
df_poblacion.dtypes
```

Out[ ]:
```
provincia            object
anio                  int64
poblacion_total       int64
poblacion_varones     int64
poblacion_mujeres     int64
dtype: object
```

In [ ]:
```
df_poblacion.tail()
```

Out[ ]:

	provincia	anio	poblacion_total	poblacion_varones	poblacion_mujeres
770	Tierra del Fuego	2036	241593	122567	119026
771	Tierra del Fuego	2037	245734	124625	121109
772	Tierra del Fuego	2038	249853	126670	123183
773	Tierra del Fuego	2039	253948	128702	125246
774	Tierra del Fuego	2040	258020	130721	127299

In [ ]:
```
df_poblacion.isnull().sum()
```

Out[ ]:
```
provincia            0
anio                 0
poblacion_total      0
poblacion_varones    0
poblacion_mujeres    0
dtype: int64
```

3. Eliminamos las filas que pertenencen a Total Pais, haciendo uso de un filtrado, ya que no nos sirve para nuestro analisis. Verificamos que el filtro se haya hecho satisfactoriamente

In [ ]:
```
df_poblacion = df_poblacion[df_poblacion["provincia"] != "Total País"]
df_poblacion[df_poblacion["provincia"] == "Total País"].sum()
```

Out[ ]:
```
provincia            0
anio                 0
poblacion_total      0
poblacion_varones    0
poblacion_mujeres    0
dtype: object
```

4. Verificamos la base de datos de hogares_viviendas_superficie

In [ ]:
```
df_hogares.head()
```

Out[ ]:

	provincia_id	provincia	hogares	viviendas_particulares	viviendas_particulares
0	2	Capital Federal	1150134	1423973	
1	6	Buenos Aires	4789484	5377786	
2	10	Catamarca	96001	113634	
3	14	Córdoba	1031843	1232211	
4	18	Corrientes	267797	292644	

In [ ]:
```
df_hogares.describe()
```

Out[ ]:

	provincia_id	hogares	viviendas_particulares	viviendas_particulares_h
count	24.000000	2.400000e+01	2.400000e+01	2.40(
mean	48.000000	5.071531e+05	5.755052e+05	4.715
std	28.284271	9.631395e+05	1.087472e+06	8.908
min	2.000000	3.895600e+04	4.336000e+04	3.668
25%	25.000000	1.369578e+05	1.513558e+05	1.270
50%	48.000000	2.086070e+05	2.393215e+05	1.942
75%	71.000000	3.701838e+05	4.034278e+05	3.41'
max	94.000000	4.789484e+06	5.377786e+06	4.425

In [ ]:
```
df_hogares["provincia"].unique()
```

Out[ ]:
```
array(['Capital Federal', 'Buenos Aires', 'Catamarca', 'Córdoba',
       'Corrientes', 'Chaco', 'Chubut', 'Entre Ríos', 'Formosa', 'Juju
y',
       'La Pampa', 'La Rioja', 'Mendoza', 'Misiones', 'Neuquén',
       'Río Negro', 'Salta', 'San Juan', 'San Luis', 'Santa Cruz',
       'Santa Fe', 'Santiago del Estero', 'Tucumán', 'Tierra del Fueg
o'],
      dtype=object)
```

In [ ]:
```
df_hogares.dtypes
```

Out[ ]:
```
provincia_id                          int64
provincia                            object
hogares                               int64
viviendas_particulares                int64
viviendas_particulares_habitadas      int64
superficie_km2                        int64
dtype: object
```

In [ ]:
```
df_hogares.tail()
```

Out[ ]:

	provincia_id	provincia	hogares	viviendas_particulares	viviendas_particulares
19	78	Santa Cruz	81796	93881	
20	82	Santa Fe	1023777	1143651	
21	86	Santiago del Estero	218025	242034	
22	90	Tucumán	368538	396040	
23	94	Tierra del Fuego	38956	43360	

5. Realizamos un merge para tener superficie_km2 y poblacion todo en una misma tabla, podemos usar la columna `provincia` como referencia.

In [ ]:
```
df_total = pd.merge(df_poblacion[["provincia", "anio", "poblacion_total"]
df_total.head()
```

Out[ ]:

	provincia	anio	poblacion_total	superficie_km2
0	Capital Federal	2010	3028481	200
1	Capital Federal	2011	3033639	200
2	Capital Federal	2012	3038860	200
3	Capital Federal	2013	3044076	200
4	Capital Federal	2014	3049229	200

6. Creamos la nueva columna densidad y realizamos el calculo

In [ ]:
```
df_total["densidad"] = df_total["poblacion_total"] / df_total["superficie
df_total[df_total["provincia"] == "Buenos Aires"].head()
```

Out[ ]:

	provincia	anio	poblacion_total	superficie_km2	densidad
31	Buenos Aires	2010	15716942	307571	51.100208
32	Buenos Aires	2011	15909607	307571	51.726616
33	Buenos Aires	2012	16100618	307571	52.347647
34	Buenos Aires	2013	16289599	307571	52.962077
35	Buenos Aires	2014	16476149	307571	53.568604

7. Identeficar valores extremos o fuera de rango

Estadisticas basicas

In [ ]:
```
df_total["densidad"].describe()
```

Out[ ]:
```
count      744.000000
mean       654.743585
std       3063.717821
min          0.131340
25%          4.424877
50%          9.406474
75%         20.322919
max      15437.170000
Name: densidad, dtype: float64
```

Normalizamos los valores de la columna densidad para tener una idea de cómo se distribuyen los datos en términos de desviación estándar.

In [ ]:
```
df_total['fuera_de_rango'] = (df_total['densidad'] - df_total['densidad']
df_total
```

Out[ ]:

	provincia	anio	poblacion_total	superficie_km2	densidad	fuera_de_rar
0	Capital Federal	2010	3028481	200	15142.405000	4.7287
1	Capital Federal	2011	3033639	200	15168.195000	4.7372
2	Capital Federal	2012	3038860	200	15194.300000	4.7457
3	Capital Federal	2013	3044076	200	15220.380000	4.7542
4	Capital Federal	2014	3049229	200	15246.145000	4.7626
...	...	...	...	...	...	
739	Tierra del Fuego	2036	241593	1002445	0.241004	-0.2136
740	Tierra del Fuego	2037	245734	1002445	0.245135	-0.2136
741	Tierra del Fuego	2038	249853	1002445	0.249244	-0.2136
742	Tierra del Fuego	2039	253948	1002445	0.253329	-0.2136
743	Tierra del Fuego	2040	258020	1002445	0.257391	-0.2136

744 rows × 6 columns

Después, podemos calcular el percentil 99 y filtrar los valores que superan este umbral, identificando así los valores extremadamente altos.

In [ ]:
```
p99 = np.percentile(df_total['densidad'], 99)
p99
```

Out[ ]: 15415.93675

In [ ]:
```
df_atipicos = df_total[df_total['densidad'] >= p99]
df_atipicos
```

Out[ ]:

	provincia	anio	poblacion_total	superficie_km2	densidad	fuera_de_rango
13	Capital Federal	2023	3083770	200	15418.850	4.819016
14	Capital Federal	2024	3085483	200	15427.415	4.821812
15	Capital Federal	2025	3086680	200	15433.400	4.823766
16	Capital Federal	2026	3087338	200	15436.690	4.824839
17	Capital Federal	2027	3087434	200	15437.170	4.824996
18	Capital Federal	2028	3086973	200	15434.865	4.824244
19	Capital Federal	2029	3085971	200	15429.855	4.822608
20	Capital Federal	2030	3084450	200	15422.250	4.820126

In [ ]:
```
df_total = df_total[df_total['densidad'] < p99]
df_total
```

Out[ ]:

	provincia	anio	poblacion_total	superficie_km2	densidad	fuera_de_rar
0	Capital Federal	2010	3028481	200	15142.405000	4.7287
1	Capital Federal	2011	3033639	200	15168.195000	4.7372
2	Capital Federal	2012	3038860	200	15194.300000	4.7457
3	Capital Federal	2013	3044076	200	15220.380000	4.7542
4	Capital Federal	2014	3049229	200	15246.145000	4.7626
...	...	...	...	...	...	
739	Tierra del Fuego	2036	241593	1002445	0.241004	-0.2136
740	Tierra del Fuego	2037	245734	1002445	0.245135	-0.2136
741	Tierra del Fuego	2038	249853	1002445	0.249244	-0.2136
742	Tierra del Fuego	2039	253948	1002445	0.253329	-0.2136
743	Tierra del Fuego	2040	258020	1002445	0.257391	-0.2136

736 rows × 6 columns

In [ ]:
```
df_total[(df_total["anio"] == 2020)]
```

Out[ ]:

	provincia	anio	poblacion_total	superficie_km2	densidad	fuera_de_ra
10	Capital Federal	2020	3075646	200	15378.230000	4.805
41	Buenos Aires	2020	17541141	307571	57.031193	-0.195
72	Catamarca	2020	415438	102602	4.049024	-0.212
103	Córdoba	2020	3760450	165321	22.746354	-0.206
134	Corrientes	2020	1120801	88199	12.707638	-0.209
165	Chaco	2020	1204541	99633	12.089779	-0.209
196	Chubut	2020	618994	224686	2.754929	-0.212
227	Entre Ríos	2020	1385961	78781	17.592579	-0.207
258	Formosa	2020	605193	72066	8.397760	-0.210
289	Jujuy	2020	770881	53219	14.485071	-0.208
320	La Pampa	2020	358428	143440	2.498801	-0.212
351	La Rioja	2020	393531	89680	4.388169	-0.212
382	Mendoza	2020	1990338	148827	13.373501	-0.209
413	Misiones	2020	1261294	29801	42.323882	-0.199
444	Neuquén	2020	664057	94078	7.058579	-0.211
475	Río Negro	2020	747610	203013	3.682572	-0.212
506	Salta	2020	1424397	155488	9.160816	-0.210
537	San Juan	2020	781217	89651	8.713980	-0.210
568	San Luis	2020	508328	76748	6.623339	-0.211
599	Santa Cruz	2020	365698	243943	1.499112	-0.213
630	Santa Fe	2020	3536418	133007	26.588210	-0.205
661	Santiago del Estero	2020	978313	136351	7.174960	-0.211
692	Tucumán	2020	1694656	22524	75.237791	-0.189
723	Tierra del Fuego	2020	173432	1002445	0.173009	-0.213

Conclusión

A partir del análisis realizado sobre la densidad de población en las distintas provincias de Argentina, podemos destacar que la Ciudad de Buenos Aires (Capital Federal) ha sido consistentemente identificada como un valor atípico con respecto a la densidad de población. Esto la coloca en una categoría separada, claramente distinta de las demás provincias, lo que confirma su carácter único como la ciudad más densamente poblada del país.