Visualització de dades

Pràctica 1

Modelització i Visualització

Víctor Fosch Tena

Índex

1. Resum de la pràctica

L'objectiu d'aquesta pràctica és crear una representació clara d'un conjunt de dades, amb llibertat per escollir el tipus de dades i la forma i eines per representar-les.

Les dades que jo he escollit representen la taxa d'atur d'Espanya a les diferents províncies, durant el període 2002 – 2023.

Pel que fa a la representació utilitzo python tant per tractar-les com representar-les.

2. Descripció de les dades

Per a trobar les dades que buscava he utilitzat el portal de dades del gobern: https://datos.gob.es/ . El publicador del conjunt de dades escollit és l'INE (Institut Nacional d'Estadística) amb títol *'Tasa de actividad, paro y empleo por provincia y sexo.'*.

Tot i que el JSON descarregat conté informació de la taxa d'atur separada per sexe masculí, femení, i ambdós, la gràfica només fa una representació de les dades d'ambdós sexes, descartant també el total nacional.

També cal aclarar que les dades originals venen no només separades per any, des del 2002 fins al 2023, sinó també per trimestres dintre de cada any. En aquest treball però, s'ha cregut convenient simplificar el volum d'informació calculant la mitjana dels valors dels quatre trimestres de cada any, ja que l'interés de la gràfica es veure com ha anat evolucionant la taxa d'atur nacional, i veure com varia en cada trimestre dintre d'un mateix any suposa una carrega visual que afegeix poc valor informatiu a la gràfica i dificulta la comprensió d'aquesta.

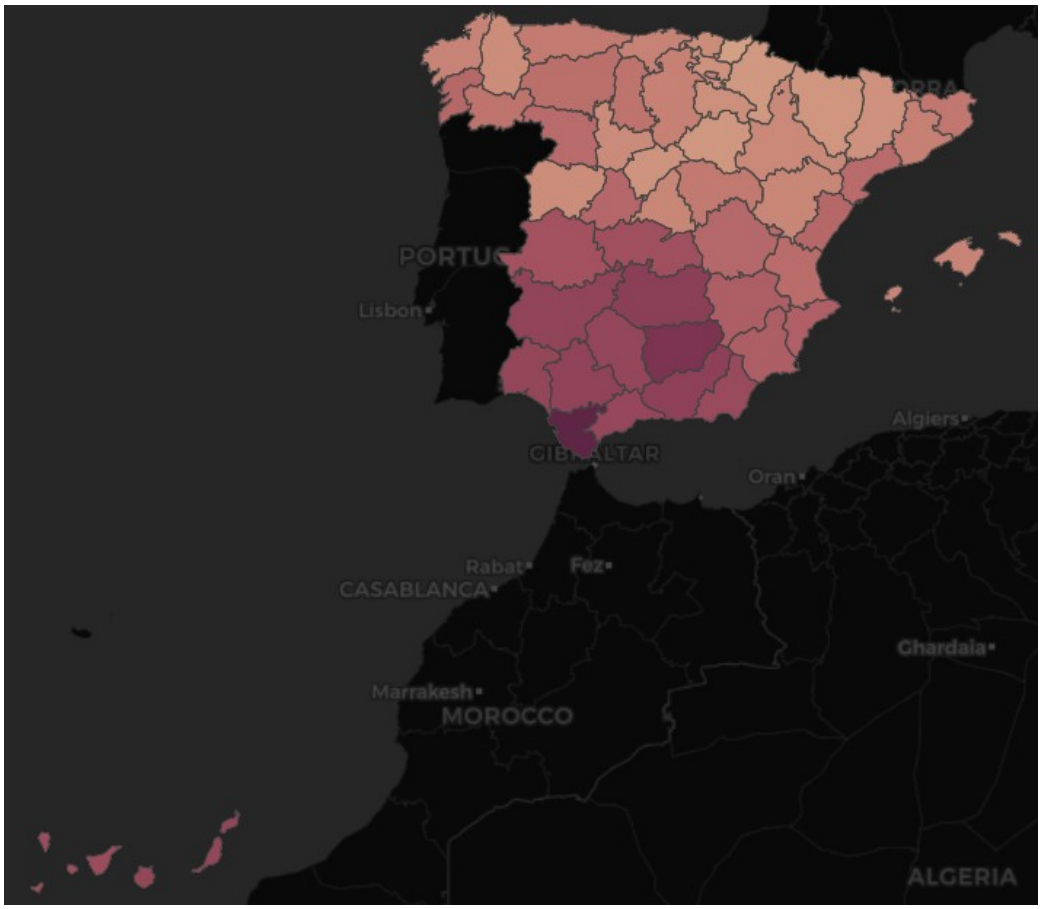
3. Explicació de la gràfica

Els punts calu de la gràfica són dos, el mapa i la línia temporal.

Pel que fa al mapa, es tracta d'una representació del país amb les delimitacións provincials, on cada província rep un color en funció de la taxa d'atur que té un any determinat.

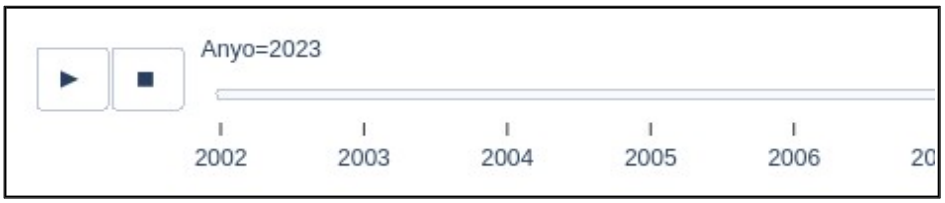
El color que pot tenir una província va del groc clar per un nivell baix d'atur, al roig vi per un nivell alt d'atur.

Per a obtenir més informació en qualsevol momnet, és pot passar el ratolí per damunt d'una província. Això mostra un diàleg amb informació de l'any, el nom de la província i el valor exacte de l'atur en punts percentuals.



Pel que fa a la línia de temps, aquesta es l'eina que s'ha escollit per poder veure la informació de les dades en diferents anys, ja que és una forma intuitiva d'interactuar amb una variable temporal.

En qualsevol moment es pot escollir un any en concret per veure les dades d'aquell any, o bé clicar el boto de *play* per veure una animació de l'evolució de l'atur.



4. Codi

A continuació s'adjunta el codi utilitzat tant per processar com per representar les dades. El codi està degudament comentat per a facilitar-ne la comprensió.

El codi necessita tres llibreries de python:

- **Pandas**: per a carregar els fitxer en format *.json*, crear el *DataFrame* i tractar les dades.
- **Geopandas**: per a carregar i utilitzar el fitxer *.geojson* que conté la informació geogràfica de totes les provincies d'Espanya.
- **Plotly.express**: per a crear el mapa i representar les dades amb la funció *choropleth_mapbox()*.

Execució: És important assegurar-se de tenir conexió a internet, ja que alguns dels paràmetres passats a la funció *choropleth_mapbox* necessiten accés a una API.

Per executar el codi hi ha dues opcions. La primera és descarregar al teu entorn les tres llibreries mencionades, i després executar el fitxer *occ_mapgraph.py*.

L'altra opció és crear un entorn virtual i descarregar-li les llibreries. Per fer-ho executa primer la comanda 'python -m venv ~/venv' per crear l'entorn, seguit de 'source ~/venv/bin/activate' per executar l'entorn. A continuació descarrega les llibreries amb 'pip install pandas geopandas plotly.express'. Per últim executa el fitxer occ_mapgraph.py.

4.1 occ_mapgraph.py

```
import plotly.express as px
import pandas as pd
import geopandas as gdp
import format_file as ff

# 'Datos.json' contains all the unemployment separated by provinces
data = pd.read_json('Datos.json')

# 'ff.format_df' function cleans and formats the data for our specific porpouse
# See 'format_file.py' for details
df = ff.format_df(pd.DataFrame(data))

# 'spain.geojson' contains geographic boundaries (lat/lon) for Spanish provinces
geojson = gdp.read_file('spain.geojson')

spain_center={'lat':36.234410, 'lon':-4.884160}

# Creates a choropleth map with the given data
# ----------
# Key parameters:
# - df: the DataFrame containing the data
# - geojson: the GeoJSON file with province boundaries
# - featureidkey: establishes a relation between 'locations' from df and 'properties.name'
from geojson
# - animation_frame: adds a time-line, based on the year in this case
# - color: colors each province based on the 'Valor' column
fig = px.choropleth_mapbox(
        df,
        geojson=geojson,
        locations='Provincia',
        featureidkey='properties.name',
        animation_frame='Anyo',
        color='Valor',
        color_continuous_scale='brwnyl',
        range_color=[0,40],
        zoom=4,
        center=spain_center,
        mapbox_style='carto-darkmatter',
        labels={'taxa_atur': 'Taxa d\'atur'},
        title='Mapa Tasa d\'Atur per Província'
)

fig.show()
```

4.2 format_file.py

```
import pandas as pd

#----------
#Formats unemployment DataFrame to get rid of unwanted data
#----------
#Parameters:
# og_df: pd.DataFrame
# That is the DataFrame is going to get formated
#Returns:
# formated_df: pd.DataFrame
# The DataFrame, only with the needed data
#----------
def format_df(og_df: pd.DataFrame) -> pd.DataFrame:
        #Obtain only the unemployment rate data
        formated_df = og_df[og_df['Nombre'].str.contains(r'^Tasa de paro de la
        población\. .*\. Ambos sexos\. Total\. $', regex=True)]

        #Create a 'Provincia' column for easy map creating
        formated_df['Provincia'] = formated_df['MetaData'].apply(get_province)

        unwanted_cols = ['COD','Nombre','T3_Unidad','T3_Escala','MetaData']
        for col in unwanted_cols:
                formated_df.pop(col)

        #Get rid of unwanted data on 'Data' column
        formated_df['Data'] = formated_df['Data'].apply(filter_data)
        #Puts 'Provincia' as the first column, just for esthetic porpuses
        col = formated_df.pop('Provincia')
        formated_df.insert(0,'Provincia',col)

        #Uncoment the print instructions for visual understanding of the following

        #Creates a row for each dictionari on Data column.
        #So now we have a different row for each year-quarter and province
        df_exploded = formated_df.explode('Data').reset_index(drop=True)
        #print(df_exploded)

        #Creates a column for each key on 'Data' column.
        #So now we have 'T3_Periodo', 'Anyo', 'Valor' columns instead of 'Data' column
        containing them all
        df_normalized = pd.concat([df_exploded.drop(columns=['Data']),
        pd.json_normalize(df_exploded['Data'])], axis=1)
        #print(df_normalized)

        #Calculates the mean of all four quarters unemployment value of each year.
        formated_df = df_normalized.groupby(['Provincia', 'Anyo'], as_index=False)
        ['Valor'].mean()
```

```
        #print(formated_df)

        return formated_df

#----------
#Gets the province name off of the 'MetaData' column
#----------
#Parameters:
# meta_data: list[dict]
# A list of dictionaries where each province name is located
#Returns:
# data.get('Nombre'): str
# The province name of each 'MetaData' column
#----------
def get_province(meta_data: list[dict]) -> str:
        for data in meta_data:
                if data.get('T3_Variable') == 'Provincias':
                        return data.get('Nombre')

        return None

#----------
#Creates a new 'Data' column with only the needed data
#----------
#Parameters:
# data_list: list[dict]
# A list of dictionaries with diferent type of data, some of them we don't need
#Returns:
# new list[dict]
# Returns a new list of dictionaries but filtered with only the data we want
#----------
def filter_data(data_list: list[dict]) -> list[dict]:
        return [{key: record[key] for key in ['T3_Periodo','Anyo','Valor']} for record in
        data_list]
```