Міністерство освіти і науки України
НТУУ «КПІ ім. Ігоря Сікорського»
Навчально-науковий інститут атомної та теплової енергетики
Кафедра цифрових технологій в енергетиці

Лабораторна робота №6

з дисципліни «Вступ до інтелектуального аналізу даних»

Тема «Географічна візуалізація та аналіз даних у Python»

Варіант №19

Студента 3-го курсу НН ІАТЕ гр. ТР-12
Ковальова Олександра
Перевірив: д.т.н., проф. Путренко В. В.

КИЇВ 2024

Мета: Опрацювати приклад роботи з географічною візуалізацією та аналізом даних, використовуючи Jupyter Notebook. Виконати поставлене завдання.

Хід роботи

Для початку підключаємо потрібні бібліотеки:

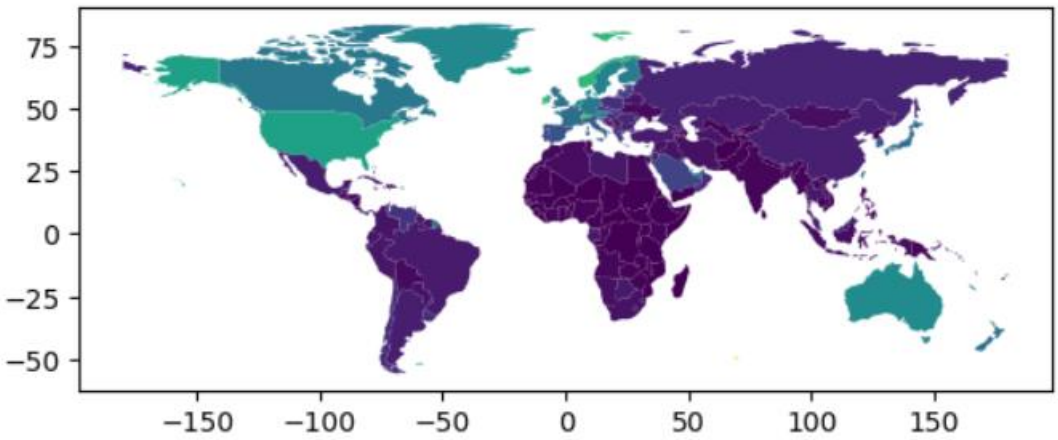
```
[2]
import pandas as pd
import geopandas as gpd
import matplotlib.pyplot as plt

import warnings
warnings.filterwarnings('ignore')
```
Executed at 2024.04.14 16:28:00 in 5ms

Комірка 1:

```
[6]
world = gpd.read_file(
    gpd.datasets.get_path(
        'naturalearth_lowres'))

world = world[
    (world.pop_est > 0) &
    (world.name != "Antarctica")]

world['gdp_per_cap'] = (
        world.gdp_md_est /
        world.pop_est)

world.plot(
    column='gdp_per_cap')

plt.show()
```
Executed at 2024.04.14 16:49:35 in 148ms

У коді виконується завантаження та аналіз географічних даних про країни світу. Спочатку, дані про країни зчитуються з файлу 'naturalearth_lowres' та записуються у змінну 'world'. Далі, проводиться фільтрація даних, при якій видаляються країни з негативною або нульовою оцінкою населення та Антарктида. Після цього, обчислюється показник ВВП на душу населення для кожної країни, що розраховується як відношення загального ВВП до населення. Нарешті, дані візуалізуються у вигляді кольорової карти, де кожна країна представлена відповідним кольором в залежності від її ВВП на душу населення.

Результат:



Комірка 2:

```
data = gpd.read_file(
    gpd.datasets.get_path(
        'naturalearth_lowres'))

data = data[
    (data.pop_est > 0) &
    (data.name != 'Antarctica')]
data['centroid_column'] = data.centroid
data['gdp_per_cap'] = (data.gdp_md_est
                       / data.pop_est)

centroids = list(data['centroid_column'])
df = pd.DataFrame({
    'y':[centroids[i].y for i in
         range(len(centroids))],
    'x':[centroids[i].x for i in
         range(len(centroids))],
    'data':list(data['gdp_per_cap'])})

base = world.plot(
    color='white',
    edgecolor='black')
df.plot(
    kind='scatter',
    x='x', y='y',
    s=df['data']*1000,
    ax=base)
plt.show()
```

Executed at 2024.04.14 16:51:24 in 297ms

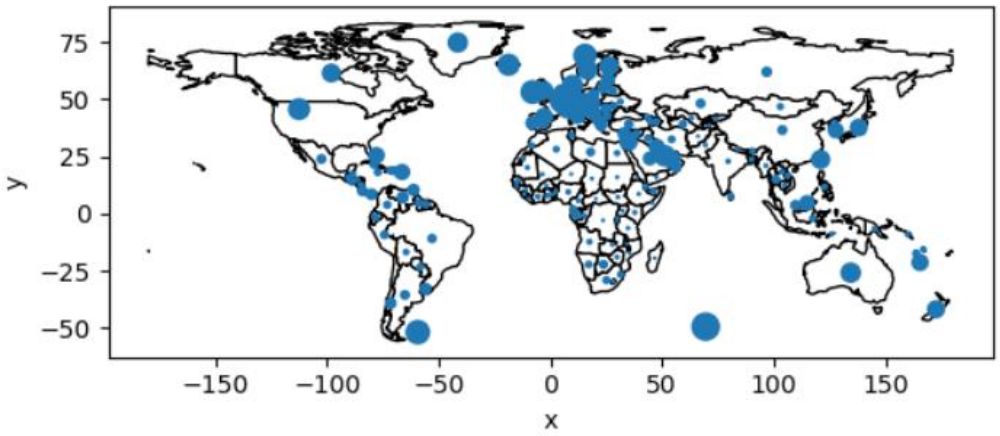
У даному коді виконується аналіз географічних даних за допомогою бібліотеки GeoPandas в середовищі Python. Початкові дані, що використовуються, представлені у вигляді набору даних "naturalearth_lowres", який містить інформацію про країни світу. Спочатку зчитуються дані та виконується попередня обробка, в результаті якої відфільтровуються країни з нульовим оціненим населенням та виключається Антарктида.

Далі проводиться розрахунок показника ВВП на душу населення для кожної країни. Для цього значення ВВП ділиться на населення. Після цього визначаються

центроїди кожної країни та створюється DataFrame з координатами центроїдів і відповідними значеннями ВВП на душу населення.

На останньому етапі відбувається візуалізація даних на карті світу. Карта побудована за допомогою функції plot() бібліотеки GeoPandas, де кожна країна має білий фон та чорні межі. Потім за допомогою функції scatter() DataFrame з координатами центроїдів відтворюється на карті, де розмір крапки відповідає величині ВВП на душу населення країни.

Результат:



Комірка 3:

```
import libpysal
import numpy as np
from esda.moran import Moran

f = libpysal.io.open(libpysal.examples.get_path("stl_hom.txt"))
y = np.array(f.by_col['HR8893'])
w = libpysal.io.open(
    libpysal.examples.get_path('stl.gal')).read()

mi = Moran(y, w, two_tailed=False)
print(mi.I)
```

Executed at 2024.04.14 16:28:02 in 1s 463ms

У наведеному коді виконується аналіз просторової автокореляції за допомогою бібліотеки PySAL в середовищі Python. Спочатку завантажуються дані, що використовуються, з файлу "stl_hom.txt", який містить дані про зайнятість у районі Сент-Луїса у 1989-1993 роках. Після цього створюється матриця сусідства за допомогою файлу "stl.gal", який містить інформацію про сусідні райони.

Далі обчислюється індекс просторової автокореляції Морана (Moran's I) за допомогою класу Moran з модуля esda.moran. Для цього використовуються дані про зайнятість (y) та матриця сусідства (w). Аргумент two_tailed встановлено в False, щоб використовувати односторонню альтернативу при обчисленні. Значення індексу Морана (Moran's I) виводиться на екран за допомогою функції print().

Результат:

```
0.24365582621771695
```

Висновок: Під час виконання лабораторної роботи були набуті практичні навички роботи з географічною візуалізацією та аналізом даних. Була проведена робота з бібліотеками Pandas, NumPy, PySAL, Esda.

Програмний код

Notebook.ipynb:

```
import pandas as pd
import geopandas as gpd
import matplotlib.pyplot as plt

import warnings
warnings.filterwarnings('ignore')

world = gpd.read_file(
    gpd.datasets.get_path(
        'naturalearth_lowres'))

world = world[
    (world.pop_est > 0) &
    (world.name != "Antarctica")]

world['gdp_per_cap'] = (
        world.gdp_md_est /
        world.pop_est)

world.plot(
    column='gdp_per_cap')

plt.show()

world = gpd.read_file(
    gpd.datasets.get_path(
        'naturalearth_lowres'))

world = world[
    (world.pop_est > 0) &
    (world.name != "Antarctica")]

data = gpd.read_file(
    gpd.datasets.get_path(
        'naturalearth_lowres'))

data = data[
    (data.pop_est > 0) &
    (data.name != 'Antarctica')]
data['centroid_column'] = data.centroid
data['gdp_per_cap'] = (data.gdp_md_est
                       / data.pop_est)

centroids = list(data['centroid_column'])
df = pd.DataFrame({
    'y':[centroids[i].y for i in
         range(len(centroids))],
    'x':[centroids[i].x for i in
         range(len(centroids))],
    'data':list(data['gdp_per_cap'])})
```

```
base = world.plot(
    color='white',
    edgecolor='black')
df.plot(
    kind='scatter',
    x='x', y='y',
    s=df['data']*1000,
    ax=base)
plt.show()

import libpysal
import numpy as np
from esda.moran import Moran

f = libpysal.io.open(libpysal.examples.get_path("stl_hom.txt"))
y = np.array(f.by_col['HR8893'])
w = libpysal.io.open(
    libpysal.examples.get_path('stl.gal')).read()

mi = Moran(y, w, two_tailed=False)
print(mi.I)
```