МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

Факультет інформатики та обчислювальної техніки Кафедра інформатики та програмної інженерії

Практикум №4

з курсу «Аналіз даних в інформаційнних системах» на тему: «Вивідна статистика»

Викладач:

Ліхоузова Т.A.

Виконав:

студент 2 курсу групи ІП-15 ФІОТ  
Мєшков Андрій Ігорович

Київ-2023

**Практикум №4**

**Вивідна статистика**

**Мета роботи**:

* ознайомитись з методами визначення точкових оцінок параметрів розподілу; дослідити, що впливає на якість точкових оцінок;
* методикою визначення інтервальних оцінок параметрів розподілу; дослідити, що впливає на якість інтервальних оцінок; методами перевірки статистичних гіпотез про вигляд закону розподілу;
* дослідити, що впливає на ширину критичної області.

**Завдання:**

Скачати потрібні дані.

**Основне завдання**

Скачати дані файлу Data2.csv.

1. Подивитись, проаналізувати структуру
2. Вказати, чи є параметри, що розподілені за нормальним законом
3. Перевірити гіпотезу про рівність середнього і медіани для одного з параметрів
4. Вказати, в якому регіоні розподіл викидів СО2 найбільш близький до нормального
5. Побудувати кругову діаграму населення по регіонам

**Додаткове завдання**

Завдання 1

1. Завантажити карту України  Ukraine.jpg
2. Розмістити бульбашки, що відповідають їх населенню, на довільних 5 містах (статистику взяти в інтернеті)
3. Знайти найбільшу відстань між містами в пікселях та кілометрах

Завдання 2

1. Завантажити файл з даними про конфлікти conflicts.csv
2. Побудувати просторовий розподіл конфліктів у Європі та Україні
3. Для Європи побудувати ізокліни для розподілу конфліктів
4. Для України:
   1. визначити, який регіон представлено на реальній карті;
   2. класифікувати точки по рокам;
   3. побудувати розподіл щільності.

**Хід роботи:**

**Основне завдання:**

*Імпортуємо потрібні бібліотеки.*

*import* pandas *as* pd

*import* matplotlib.pyplot *as* plt

*import* scipy.stats *as* stats

*Напишемо шлях до файлу.*

data\_path = 'Data2.csv'

*Зчитаємо файл.*

def read\_dataset(path):

df = pd.read\_csv(path, sep=';', encoding='cp1252')

*return* df

*Виправимо дані, використовуючи функції з попередньої роботи.*

def remove\_typo(df):

df.rename(columns={"Populatiion": "Population"}, inplace=True)

*return* df

def clean\_up(df):

df['Area'] = df['Area'].str.replace(',', '.').astype(float)

df["GDP per capita"] = df["GDP per capita"].str.replace(',', '.').astype(float)

df["CO2 emission"] = df["CO2 emission"].str.replace(',', '.').astype(float)

*return* df

def fix\_negative(df):

fix\_gdp = df[df['GDP per capita'] < 0]

area\_gdp = df[df['Area'] < 0]

fix\_gdp['GDP per capita'] \*= -1

area\_gdp['Area'] \*= -1

df[df['GDP per capita'] < 0] = fix\_gdp

df[df['Area'] < 0] = area\_gdp

*return* df

def fix\_NaN(df):

df = df.fillna(df.mean())

*return* df

*Проаналізуємо структуру.*

def print\_exploring(df):

print('Data frame info:')

df.info()

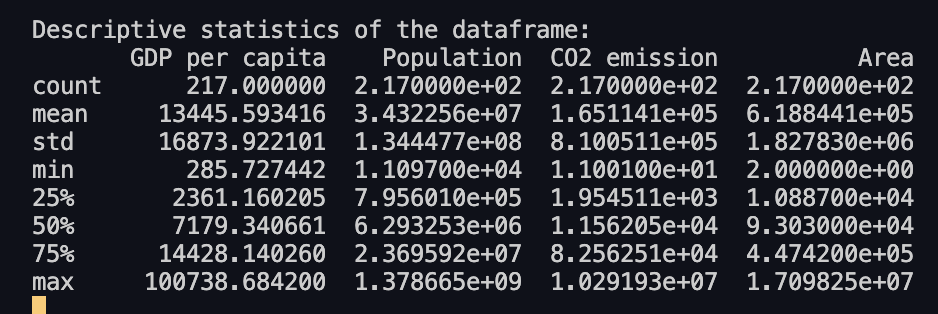
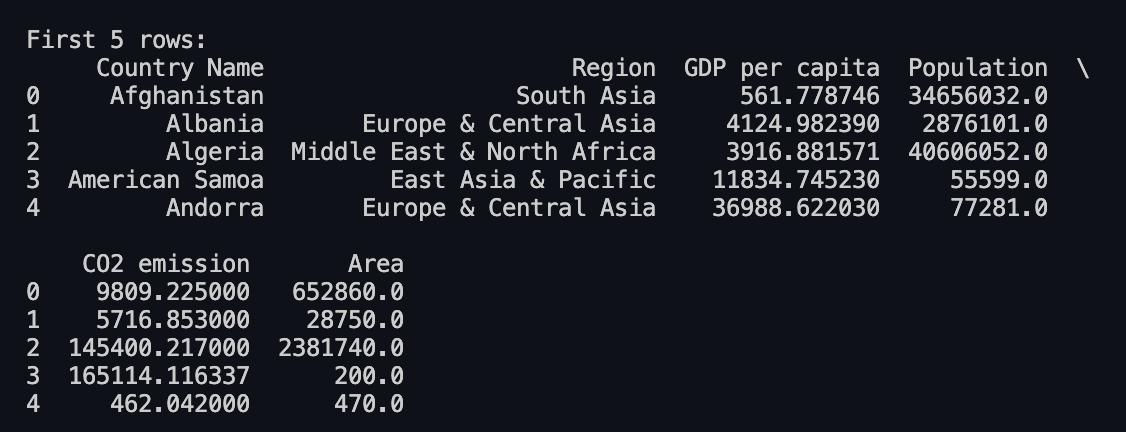
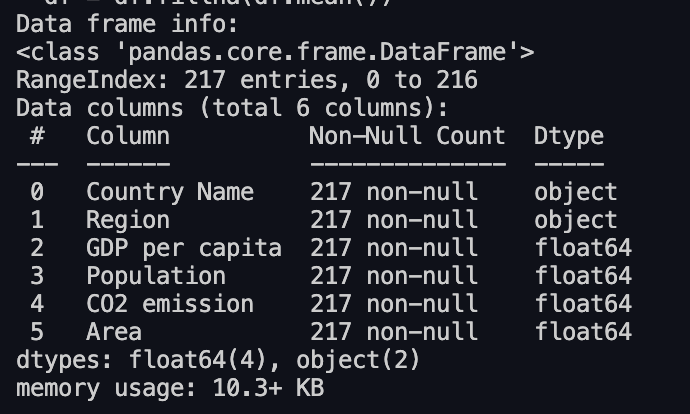
pd.set\_option("display.max\_columns", None)

print('\nFirst 5 rows:')

print(df.head())

print('\nDescriptive statistics of the dataframe:')

print(df.describe())



*Перевіремо, чи є параметри, що розподілені за нормальним законом.*

def normally\_visual\_test(df):

fig, axs = plt.subplots(1, 4, figsize=(16, 4))

fig.suptitle('Histograms', fontsize=16)

axs[0].set\_title('GDP per capita')

axs[0].hist(df['GDP per capita'])

axs[1].set\_title('Population')

axs[1].hist(df['Population'])

axs[2].set\_title('CO2 emission')

axs[2].hist(df['CO2 emission'])

axs[3].set\_title('Area')

axs[3].hist(df['Area'])

plt.show()



*Проведемо тести за критеріями.*

column = ['GDP per capita', 'Population', 'CO2 emission', 'Area']

def shapiro\_test(df, columns=0, alpha=0.05):

print("\nShapiro–Wilk test:")

*if* columns==0:

data = df

columns = [1]

*for* column *in* columns:

*if* column != 1:

data = df[column]

stat, p = stats.shapiro(data)

print('Statistics=%.3f, p=%.3f' % (stat, p))

*if* p > alpha:

print('The data correspond to a normal distribution')

*else*:

print('The data do not correspond to a normal distribution')

def ks\_test(df, columns=0, alpha=0.05):

print("\nKolmogorov–Smirnov test:")

*if* columns==0:

data = df

columns = [1]

*for* column *in* columns:

*if* column != 1:

data = df[column]

stat, p = stats.kstest(data, 'norm')

print('Statistics=%.3f, p=%.3f' % (stat, p))

*if* p > alpha:

print('The data correspond to a normal distribution')

*else*:

print('The data do not correspond to a normal distribution')

def dagostino\_test(df, columns=0, alpha=0.05):

print("\nD'Agostino's test:")

*if* columns==0:

data = df

columns = [1]

*for* column *in* columns:

*if* column != 1:

data = df[column]

stat, p = stats.normaltest(data)

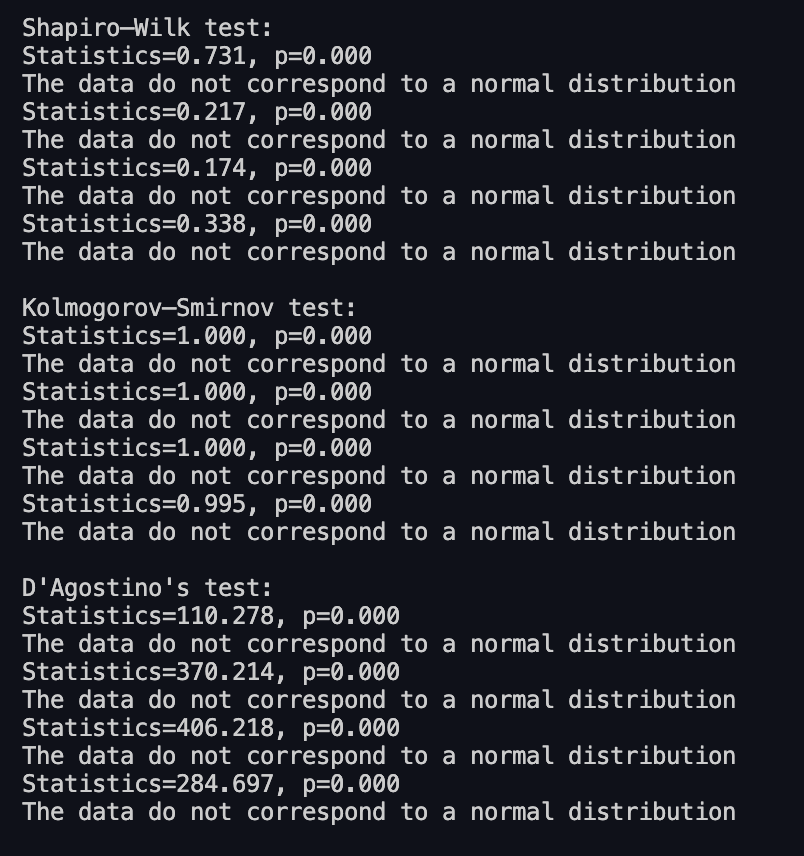
print('Statistics=%.3f, p=%.3f' % (stat, p))

*if* p > alpha:

print('The data correspond to a normal distribution')

*else*:

print('The data do not correspond to a normal distribution')



*Перевіримо гіпотезу про рівність середнього і медіани для одного з параметрів*

def mean\_median(df, columns):

*for* column *in* columns:

print(f'\n{column}:')

mean\_gdp = df[column].mean()

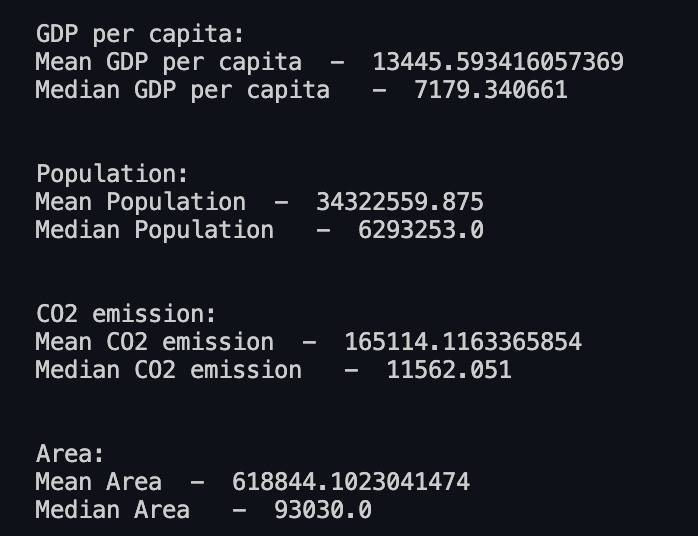
median\_gdp = df[column].median()

print(f'Mean {column}', ' - ' ,mean\_gdp)

print(f'Median {column} ', ' - ' , median\_gdp, '\n')

*if* mean\_gdp == median\_gdp:

print(f"The mean and median {column} are the same: {mean\_gdp}")



*Перевіримо, в якому регіоні розподіл викидів СО2 найбільш близький до нормального*

def closest\_co2(df):

df['CO2 emission'].hist(by=df['Region'], layout=(4, 2), figsize=(10, 20))

plt.show()

*for* region *in* df['Region'].unique():

region\_emissions = df[df['Region'] == region]['CO2 emission']

print(f'\nCheck for the region {region}:')

*try*:

shapiro\_test(region\_emissions)

*except* ValueError *as* e:

print(str(e))

*try*:

ks\_test(region\_emissions)

*except* ValueError *as* e:

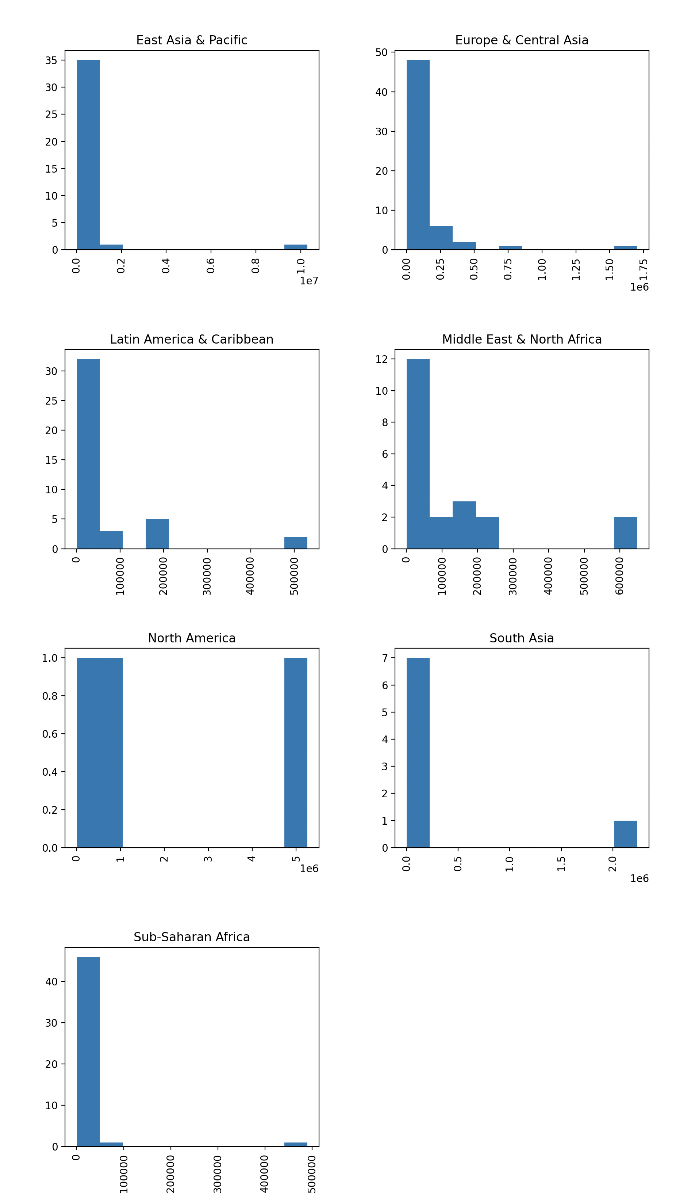
print(str(e))

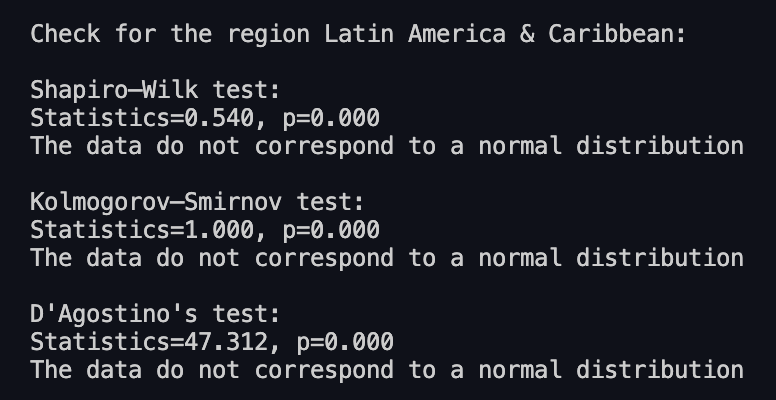
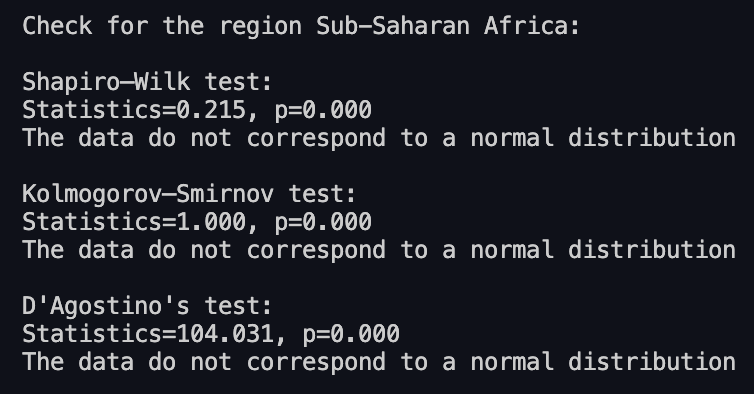
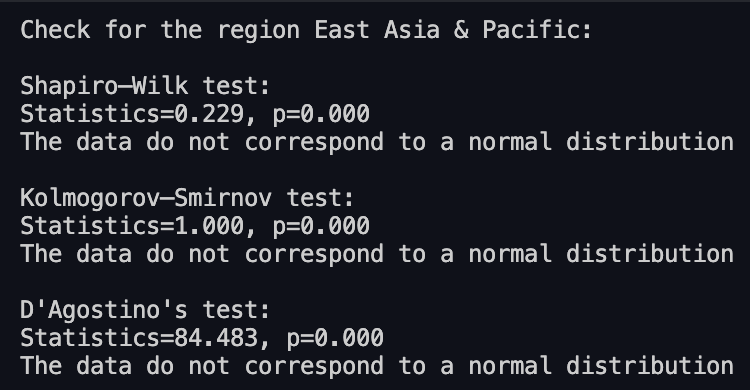
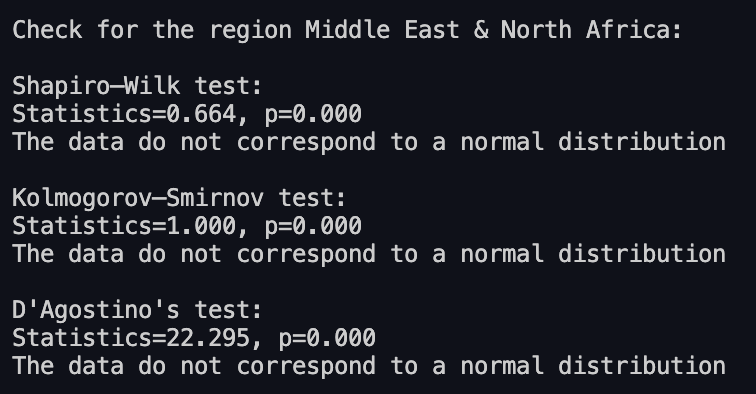
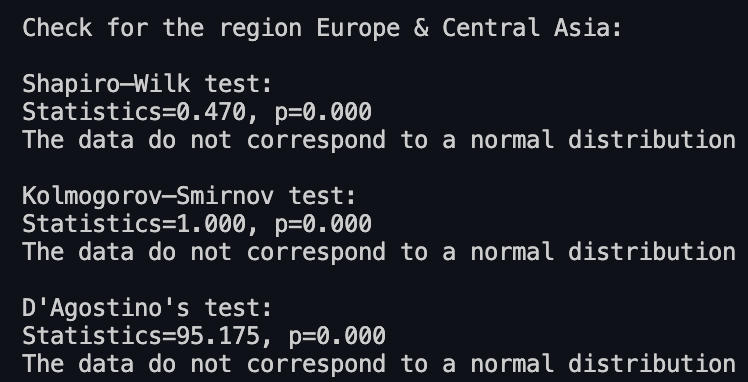
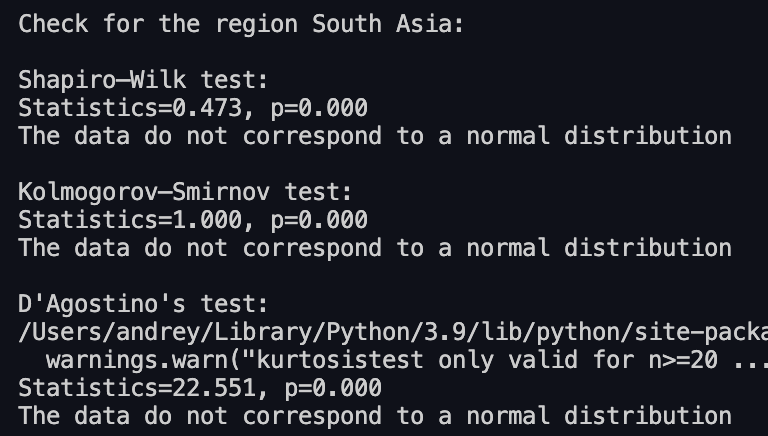
*try*:

dagostino\_test(region\_emissions)

*except* ValueError *as* e:

print(str(e))





*Побудуємо кругову діаграму населення по регіонам*

def pie\_chart(df):

fig, ax = plt.subplots(figsize=(8, 8))

fig.suptitle('Pie chart', fontsize=16)

labels = pd.unique(df['Region'])

wedges, texts, autotexts = ax.pie(df.groupby('Region').sum()['Population'], labels=labels,

autopct='%1.1f%%', textprops=dict(color='w'))

ax.set\_title('Population by region')

ax.legend(wedges, labels,

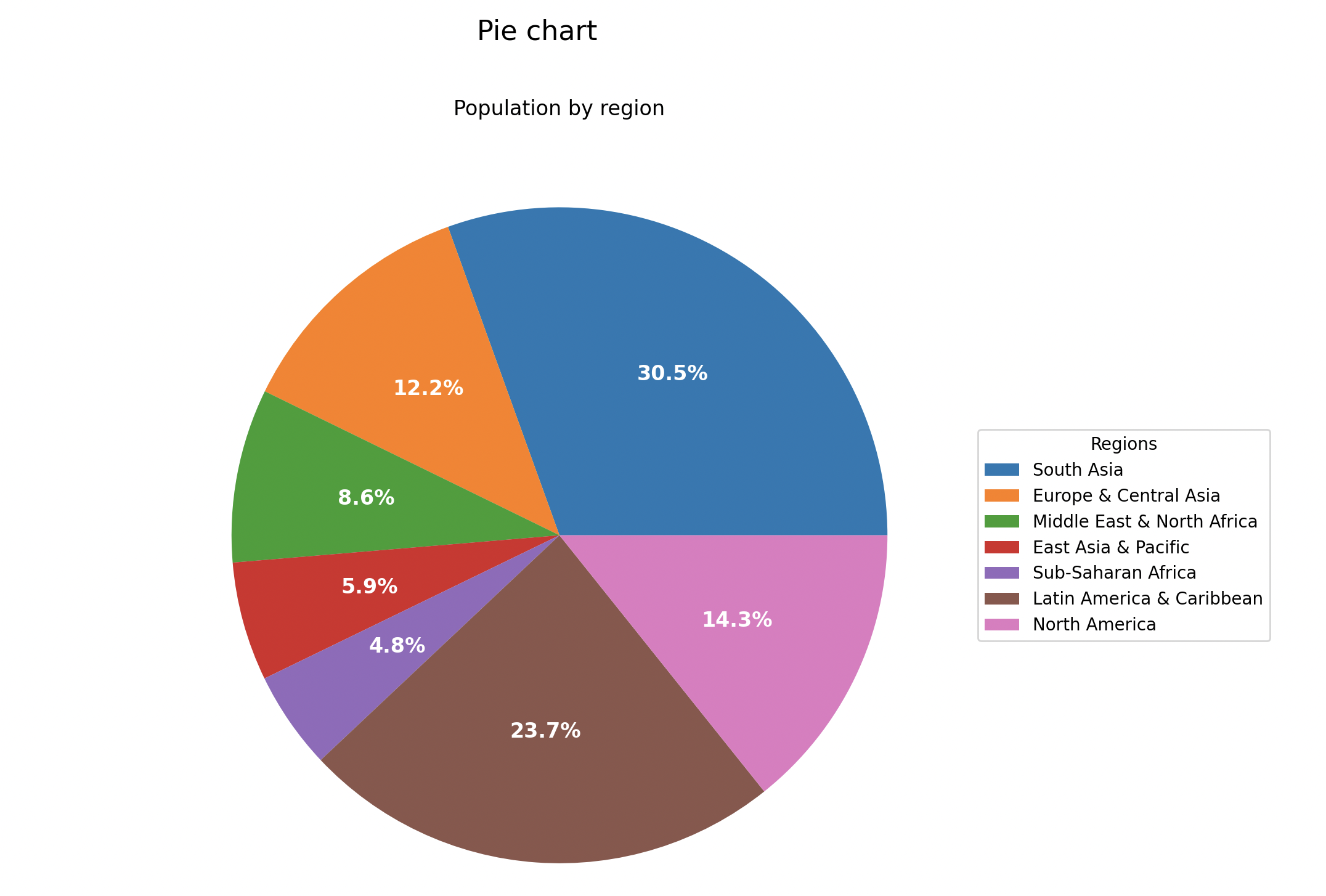
title='Regions',

loc='center left',

bbox\_to\_anchor=(1, 0, 0, 1))

plt.setp(autotexts, size=12, weight='bold')

plt.show()

****

**Додаткове завдання:**

***Завдання 1***

*Імпортуємо потрібні бібліотеки.*

*import* numpy *as* np

*import* matplotlib.pyplot *as* plt

*import* matplotlib.image *as* mpimg

*from* scipy.spatial *import* distance

*Завантажемо карту.*

img\_path = 'Ukraine.jpg'

def map\_downloading(img\_path):

map\_img = mpimg.imread(img\_path)

*return* map\_img

*Розмістити бульбашки, що відповідають їх населенню, на довільних 5 містах*

cities = ['Київ', 'Краматорськ', 'Харків', 'Львів', 'Миколаїв']

cities\_coords = np.array([(387, 146), (715, 256), (647, 176), (91, 188), (452, 381)])

cities\_population = np.array([2884, 185, 1419, 721, 486])

def bubles(map\_img, coords, population):

fig, ax = plt.subplots(figsize=(15, 15))

fig.suptitle('Ukraine', fontsize=16)

ax.imshow(map\_img)

ax.scatter(

coords[:, 0],

coords[:, 1],

s=population \* 2,

c='green',

alpha=0.5,

linewidth=2

)

ax.axis('off')

plt.show()



*Знайдемо найбільшу відстань між містами в пікселях та кілометрах*

def greatest\_distance(map\_img, cities, coords):

distances = distance.cdist(coords, coords, 'euclidean')

city\_A, city\_B = np.unravel\_index(distances.argmax(), distances.shape)

pixel\_distance = distances[city\_A, city\_B]

ukraine\_width\_km = 1316

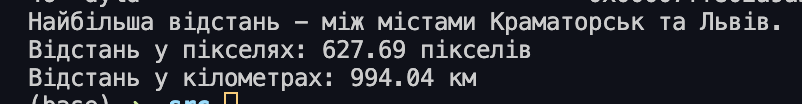
km\_per\_pixel = ukraine\_width\_km / map\_img.shape[1]

km\_distance = distances[city\_A, city\_B] \* km\_per\_pixel

print(f'Найбільша відстань - між містами {cities[city\_A]} та {cities[city\_B]}.')

print(f'Відстань у пікселях: {pixel\_distance:.2f} пікселів')

print(f'Відстань у кілометрах: {km\_distance:.2f} км')



***Завдання 2***

*Імпортуємо потрібні бібліотеки.*

*import* geoviews *as* gv

*from* geoviews *import* dim

*import* pandas *as* pd

*from* geoviews.tile\_sources *import* CartoDark

*from* geoviews.tile\_sources *import* StamenTerrain

gv.extension('bokeh', 'matplotlib')

*Зчитаємо файл.*

data\_path = 'conflicts.csv'

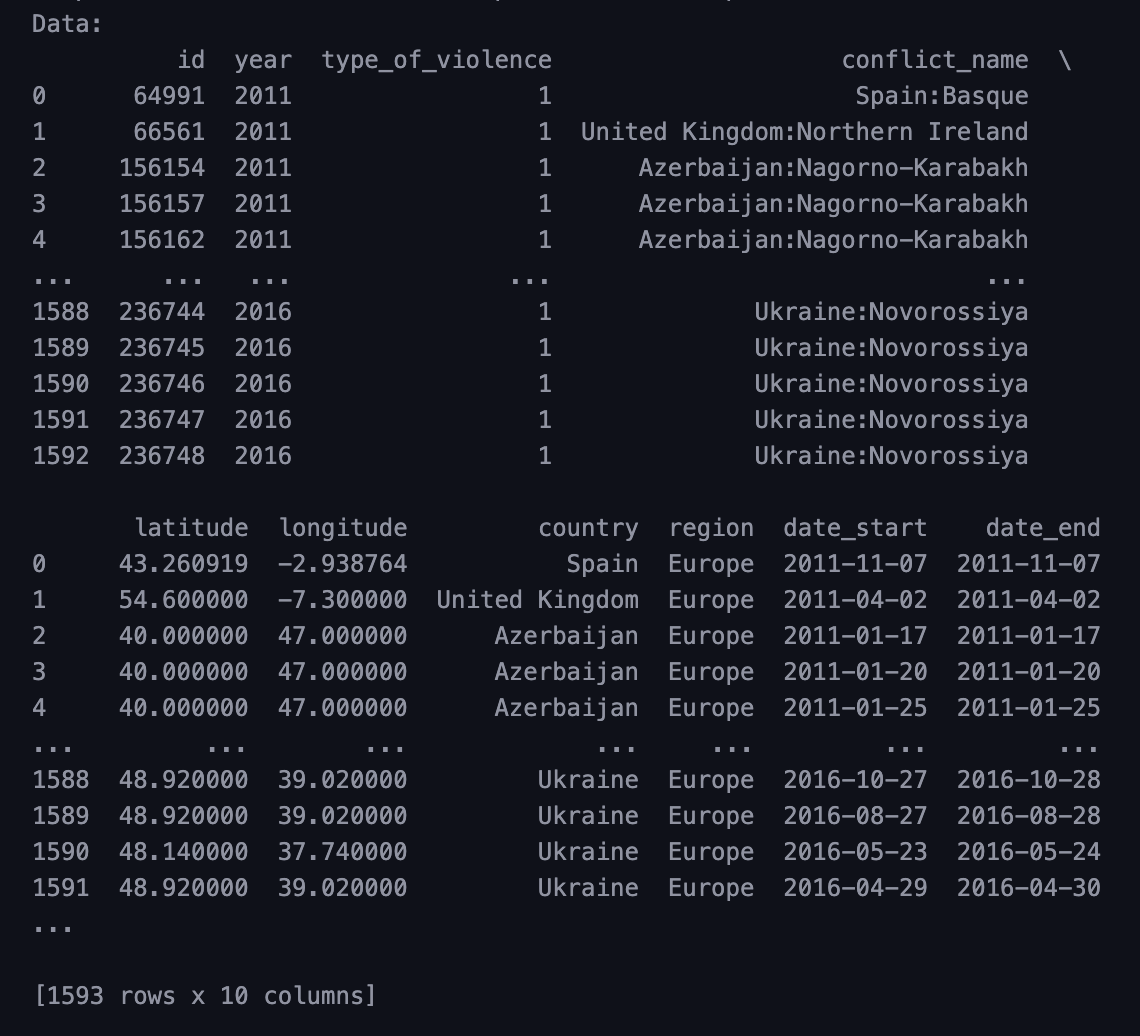
def read\_dataset(path):

df = pd.read\_csv(path, sep=';', encoding='cp1252', decimal=',')

print("Data:")

print(df)

*return* df



*Згрупуємо дані*

def grouping(conflicts):

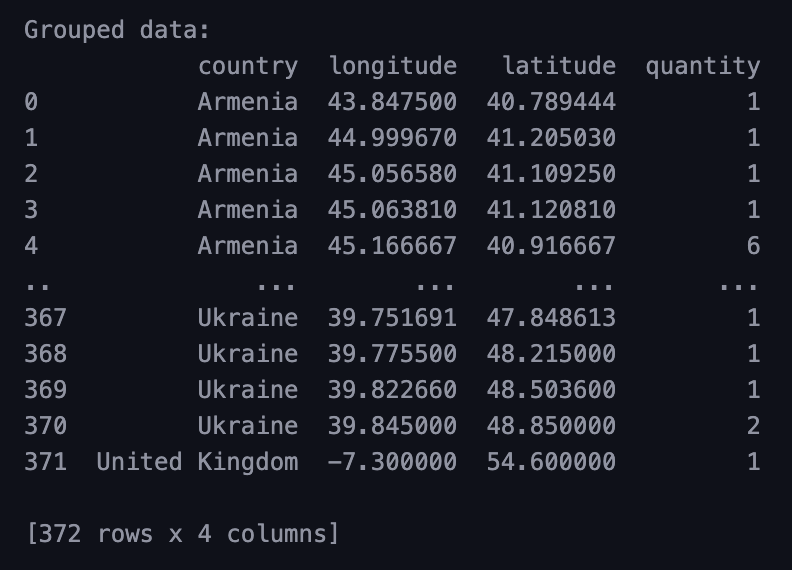
grouped\_conflicts = conflicts.groupby(['country', 'longitude', 'latitude']).size().to\_frame('quantity').reset\_index()

print("\nGrouped data:")

grouped\_conflicts

print(grouped\_conflicts)

*return* grouped\_conflicts



*Побудуємо просторовий розподіл Європи*

def visualization\_europe( g\_conflicts):

*# Europe*

points = gv.Points(g\_conflicts, ['longitude', 'latitude'])

tiles = gv.tile\_sources. StamenToner

gv.output(tiles \* points.opts(

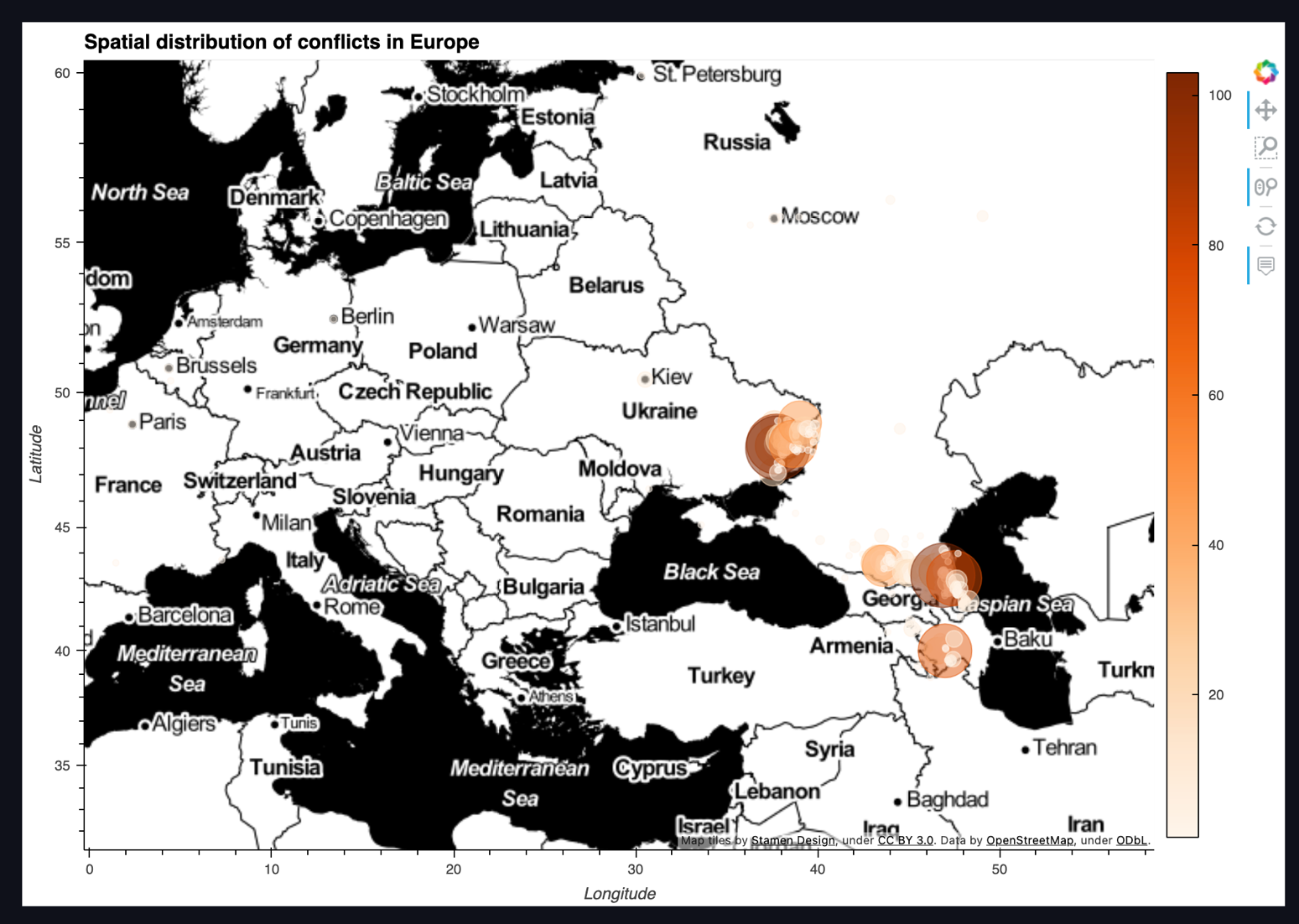
title='Spatial distribution of conflicts in Europe',

color='quantity', size=dim('quantity') \*\* (1/2) \* 5,

cmap='Oranges', tools=['hover'], width=1000, height=700,

show\_legend=False, alpha=0.5, colorbar=True

))



def visualization\_ukraine(conflicts):

*# Ukraine*

ukr\_conflicts = conflicts[conflicts['country'] == 'Ukraine']

ukr\_grouped\_conflicts = ukr\_conflicts.groupby(['year', 'longitude', 'latitude']).size().to\_frame('quantity').reset\_index()

print(ukr\_grouped\_conflicts)

ukr\_grouped\_conflicts['year'] = ukr\_grouped\_conflicts['year'].apply(str)

points = gv.Points(ukr\_grouped\_conflicts, ['longitude', 'latitude'])

tiles = gv.tile\_sources.CartoDark

gv.output(tiles \* points.opts(

title='Spatial distribution of conflicts in Ukraine',

color='year', size=dim('quantity') \*\* (1/2) \* 10,

cmap='Category10', tools=['hover'], width=1000, height=700,

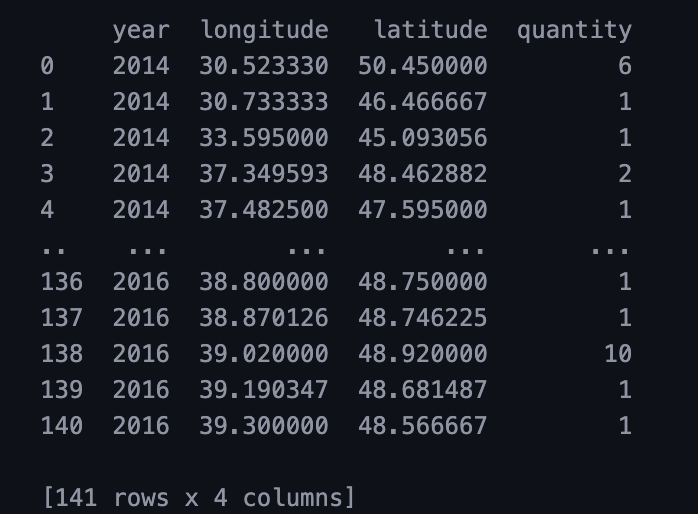
show\_legend=False, alpha=0.8

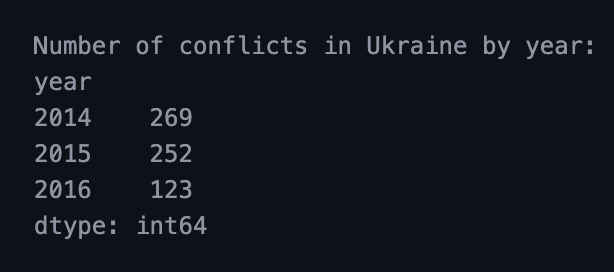
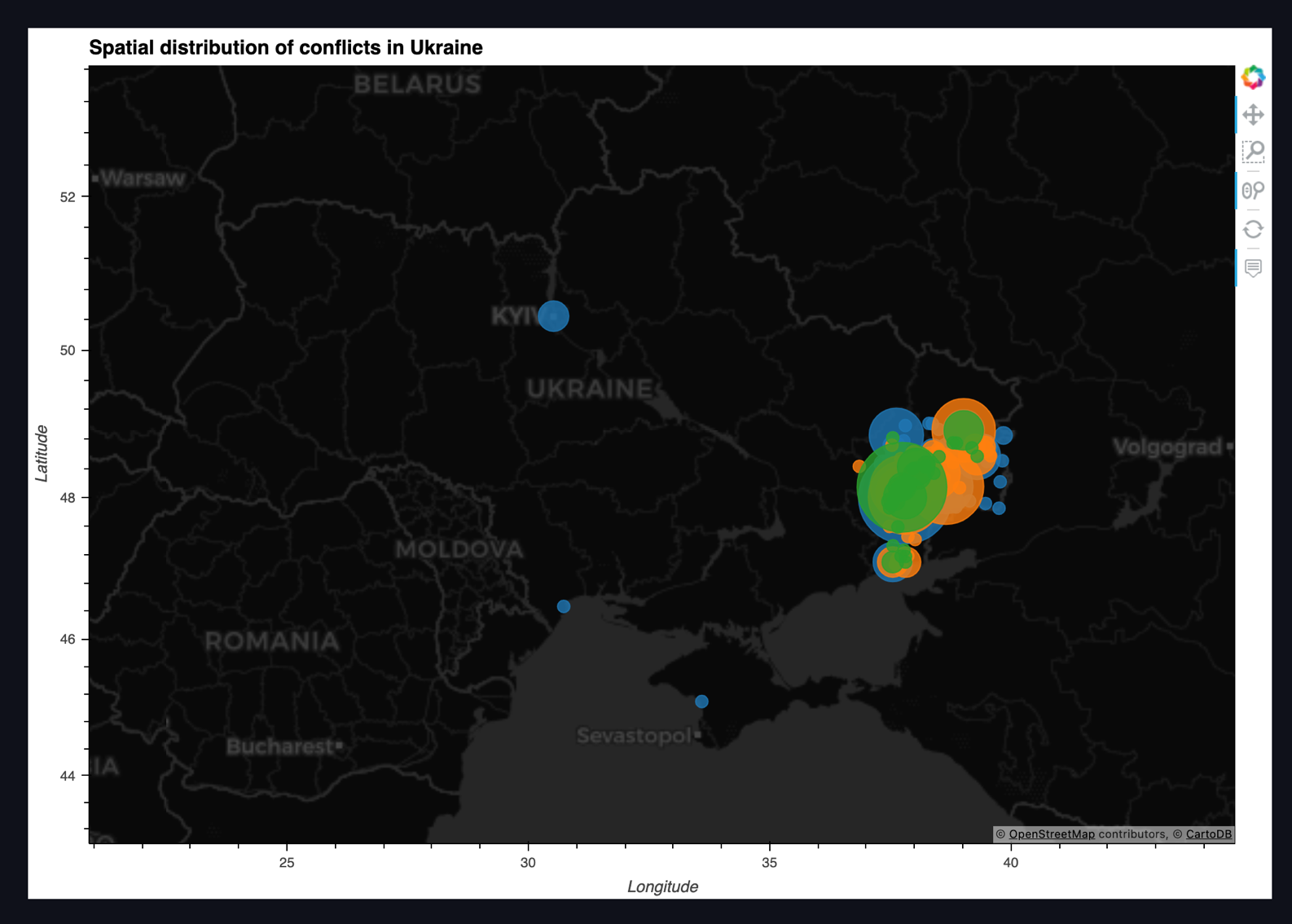
))

*# Number of conflicts in Ukraine by year*

print('\nNumber of conflicts in Ukraine by year:')

print(ukr\_conflicts.groupby(['year']).size())





Висновок

За отриманими даними можна зробити висновок, що

* розподіл викидів СО2 найбільш близький до нормального в Північній Америці
* більшість конфліктів в Європі в проміжку 2011-2016 були в Україні.