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

Звіт

з лабораторної роботи №4 з дисципліни
«Аналіз даних в інформаційних системах»

„Вивідна статистика”

Виконав(ла) *ІП-11 Тарасьонок Дмитро Євгенович*
(шифр, прізвище, ім'я, по батькові)

Перевірила *Ліхоузова Т. А.*
(прізвище, ім'я, по батькові)

Київ 2023

ЗМІСТ

1 МЕТА ЛАБОРАТОРНОЇ РОБОТИ

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.

2 ЗАВДАННЯ

2.1 Основне завдання

Скачати дані із 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 CO2 найбільш близький до нормального
5. Побудувати кругову діаграму населення по регіонам

2.2 Додаткове завдання

Відповісти на питання (файл Data2.csv):

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 shape-файл с областями України.
2. Побудувати картограми для прибутку населення на 1 особу і ВВП по регіонам за 2016 рік.
3. По даним за 2006-2015 роки для кожного регіону розрахувати коефіцієнт кореляції між прибутком населення на 1 особу та ВВП. Відобразити на картограмі.

3 ВИКОНАННЯ ОСНОВНОГО ЗАВДАННЯ

3.1 Подивитись, проаналізувати структуру

Для початку треба завантажити дані у Python за допомогою бібліотеки pandas.

```
In 2
df = pd.read_csv('Data2.csv', on_bad_lines='skip', encoding='cp1252', delimiter=';')
df
```

Executed in 33ms, 5 Apr at 23:29:22

Out 2 — 1-36 — 217 rows × 6 columns pd.Dataframe

	Country Name	Region	GDP per capita	Populatiion	CO2 emission	Area
0	Afghanistan	South Asia	561,7787463	34656032.0	9809,225	652860
1	Albania	Europe & Central Asia	4124,98239	2876101.0	5716,853	28750
2	Algeria	Middle East & North Africa	3916,881571	40606052.0	145400,217	2381740
3	American Samoa	East Asia & Pacific	11834,74523	55599.0	NaN	200
4	Andorra	Europe & Central Asia	36988,62203	77281.0	462,042	470
5	Angola	Sub-Saharan Africa	3308,700233	28813463.0	34763,16	1246700
6	Antigua and Barbuda	Latin America & Caribbean	14462,17628	100963.0	531,715	440
7	Argentina	Latin America & Caribbean	12440,32098	43847430.0	204024,546	2780400
8	Armenia	Europe & Central Asia	3614,688357	2924816.0	5529,836	29740
9	Aruba	Latin America & Caribbean	NaN	104822.0	872,746	180
10	Australia	East Asia & Pacific	49755,31548	24127159.0	361261,839	7741220
11	Austria	Europe & Central Asia	44757,6349	8747358.0	58712,337	83879
12	Azerbaijan	Europe & Central Asia	3878,709257	9762274.0	37487,741	86600
13	Bahamas, The	Latin America & Caribbean	28785,47767	391232.0	2416,553	13880

Рис. 3.1 – Завантаження даних

Бачимо, що в наборі даних є наступні стовпці: назва країни, регіон, ВВП на душу населення, кількість населення, викиди CO2 та площа. Як було описано в попередній лабораторній роботі, набір даних містить наступні помилки: дійсні числа записані через кому, є пропущенні значення ВВП на душу населення та викидів CO2 та від’ємні значення площі: виправимо ці помилки.

```
In 3
df.columns = ['country_name', 'region', 'gdp_per_capita', 'population', 'co2_emission', 'area']
df.set_index('country_name', inplace=True)
df.gdp_per_capita = df.gdp_per_capita.astype(str).str.replace(',', '.').astype(float)
df.co2_emission = df.co2_emission.astype(str).str.replace(',', '.').astype(float)
df.area = df.area.astype(str).str.replace(',', '.').astype(float)
df.gdp_per_capita = df.gdp_per_capita.abs()
df.area = df.area.abs()
df.gdp_per_capita.fillna(df.gdp_per_capita.mean(), inplace=True)
df.population.fillna(df.population.mean(), inplace=True)
df.co2_emission.fillna(df.co2_emission.mean(), inplace=True)
df.population = df.population.astype(int)
df
```

Executed in 35ms, 5 Apr at 23:29:22

Out 3 — 1-36 — 217 rows × 5 columns pd.Dataframe

country_name	region	gdp_per_capita	population	co2_emission	area
Afghanistan	South Asia	561.778746	34656032	9809.225000	652860.0
Albania	Europe & Central Asia	4124.982390	2876101	5716.853000	28750.0
Algeria	Middle East & North Africa	3916.881571	40606052	145400.217000	2381740.0
American Samoa	East Asia & Pacific	11834.745230	55599	165114.116337	200.0
Andorra	Europe & Central Asia	36988.622030	77281	462.042000	470.0
Angola	Sub-Saharan Africa	3308.700233	28813463	34763.160000	1246700.0
Antigua and Barbuda	Latin America & Caribbean	14462.176280	100963	531.715000	440.0
Argentina	Latin America & Caribbean	12440.320980	43847430	204024.546000	2780400.0
Armenia	Europe & Central Asia	3614.688357	2924816	5529.836000	29740.0
Aruba	Latin America & Caribbean	13445.593416	104822	872.746000	180.0
Australia	East Asia & Pacific	49755.315480	24127159	361261.839000	7741220.0
Austria	Europe & Central Asia	44757.634900	8747358	58712.337000	83879.0
Azerbaijan	Europe & Central Asia	3878.709257	9762274	37487.741000	86600.0

Рис. 3.2 – Виправлення помилок

3.2 Вказати, чи є параметри, що розподілені за нормальним законом

Для перевірки такої гіпотези використаємо функцію бібліотеки scipy normaltest. Вона заснована на тесті Д'Агостіно і Пірсона. Рівень значущості оберемо стандартний – 0.05.

```
In 4
def test_normality(column: pd.Series, alpha=0.05):
    _, p = stats.normaltest(column)
    return alpha < p
Executed in 18ms, 5 Apr at 23:29:22

In 5
df[['gdp_per_capita', 'population', 'co2_emission', 'area']].apply(test_normality)
Executed in 28ms, 5 Apr at 23:29:23

Out 5
4 rows  Length: 4, dtype: bool pd.Series
```

	<unnamed>
gdp_per_capita	False
population	False
co2_emission	False
area	False

Рис. 3.3 – Перевірка гіпотези про нормальність розподілу стовпців набору даних

Як бачимо, жоден із стовпців не розподілений за нормальним законом. Відобразимо гістограми й перевіримо це.

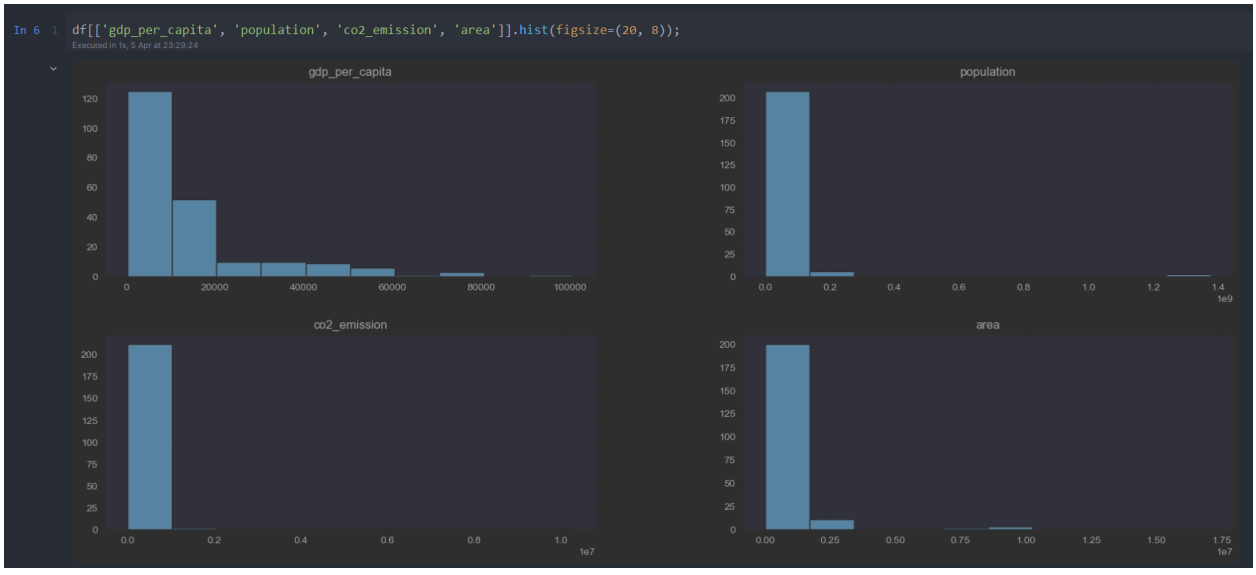

Рис. 3.4 – Гістограми для кожного стовпця

3.3 Перевірити гіпотезу про рівність середнього і медіани для одного з параметрів

Для перевірки цієї гіпотези використаємо функцію бібліотеки scipy ttest_1samp: вона використовує Т-критерій Стьюдента. Рівень значущості оберемо такий самий – 0.05.

```
In 7  1  def ttest(column: pd.Series, alpha=0.05):
      2      stat, pvalue = stats.ttest_1samp(column, column.median())
      3      return alpha < pvalue
Executed in 113ms, 5 Apr at 23:29:24

In 8  1  df[['gdp_per_capita', 'population', 'co2_emission', 'area']].apply(ttest)
Executed in 96ms, 5 Apr at 23:29:24

Out 8    4 rows   Length: 4, dtype: bool pd.Series ↗
```

	<unnamed>
gdp_per_capita	False
population	False
co2_emission	False
area	False

Рис. 3.5 – Перевірка гіпотези про рівність середнього і медіани

Як бачимо, гіпотеза не справджується для жодного зі стовпців.

3.4 Вказати, в якому регіоні розподіл викидів CO2 найбільш близький до нормального

Для такої перевірки використаємо функцію бібліотеки scipy anderson, що задіює тест Андерсона-Дарлінга. Функція повертає статистику та критичні значення для наступних рівней значущості: 15%, 10%, 5%, 2.5%, 1%. Оберемо 5%, а потім віднімемо отриману статистику від цього критичного значення. Чим менше це значення буде, тим розподіл є ближчим до нормального

```
In 9
def normality_difference(group: pd.Series):
    anderson_result = stats.anderson(group)
    differences = anderson_result.statistic - anderson_result.critical_values[2]
    differences = np.abs(differences)
    return differences.min()
Executed in 80ms, 5 Apr at 23:29:24

In 10
region_group = df.groupby('region').co2_emission.aggregate(normality_difference)
region_group[[region_group.idxmin()]]
Executed in 63ms, 5 Apr at 23:29:24

Out 10   1 row   Length: 1, dtype: float64 pd.Series
```

region	co2_emission
South Asia	1.375605

Рис. 3.6 – Пошук регіону, розподіл викидів CO2 якого є найбільш близьким до нормального

Бачимо, що для регіону «Південна Азія» така різниця дорівнює 1.3756, що все одно доволі багато.

3.5 Побудувати кругову діаграму населення по регіонам

Для побудови кругової діаграми використаємо простий метод .pie об’єкту класу DataFrame, встановимо виведення значень у форматі відсотків.

```
In 11
df.groupby('region').population.sum().plot.pie(ylabel='', autopct='%1.1f%%');
Executed in 138ms, 5 Apr at 23:29:24
```

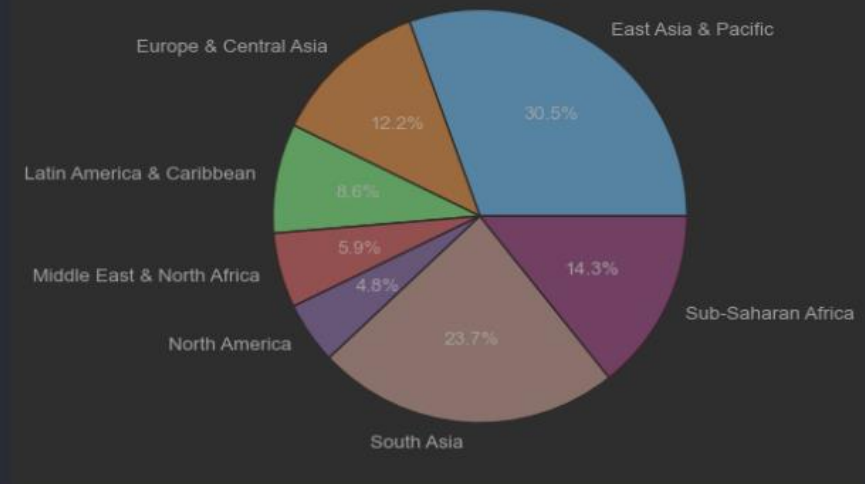

Рис. 3.7 – Побудова кругової діаграми населення за регіонами

4 ВИКОНАННЯ ПЕРШОГО ДОДАТКОВОГО ЗАВДАННЯ

4.1 Завантажити карту України Ukraine.jpg

4.2 Розмістити бульбашки, що відповідають їх населенню, на довільних 5 містах (статистику взяти в інтернеті)

Для початку знайдемо в інтернеті інформацію по населенню. Дані було взято з офіційного сайту Держстату. Також було знайдено в інтернеті дані про координати міст для подальшого розрахунку відстаней, а також прораховано координати точок міст на завантаженій карті: про це мова піде пізніше. Розмір бульбашок розрахуємо, поділивши кількість населення на 5000.

```
In 2
population = pd.read_csv('population.csv')
population = gpd.GeoDataFrame(population, geometry=gpd.points_from_xy(population.map_longitude, population.map_latitude))
population.population /= 1000
population['bubble_size'] = population.population / 5
population
```

Executed in 91ms, 6 Apr at 16:02:57

Out 2 — 26 rows × 9 columns pd.Dataframe

	name	city	population	latitude	longitude	map_latitude	map_longitude	geometry	bubble_size
0	Вінницька	Вінниця	1507.738	49.2167	28.4333	49.06	28.6100	POINT (28.61000 49.06000)	301.5476
1	Волинська	Луцьк	1020.770	50.7478	25.3244	50.69	25.5200	POINT (25.52000 50.69000)	204.1540
2	Дніпропетровська	Дніпр	3093.151	48.4675	35.0400	48.32	35.2100	POINT (35.21000 48.32000)	618.6302
3	Донецька	Донецьк	4056.405	48.0089	37.8042	47.76	37.8300	POINT (37.83000 47.76000)	811.2810
4	Житомирська	Житомир	1177.564	50.2544	28.6578	50.07	28.8200	POINT (28.82000 50.07000)	235.5128
5	Закарпатська	Ужгород	1243.721	48.6239	22.2950	48.48	22.6300	POINT (22.63000 48.48000)	248.7442
6	Запорізька	Запоріжжя	1636.322	47.8378	35.1383	47.70	35.1200	POINT (35.12000 47.70000)	327.2644
7	Івано-Франківська	Івано-Франківськ	1350.565	48.9228	24.7106	48.73	24.7900	POINT (24.79000 48.73000)	270.1130
8	Київська	Київ	4746.244	50.4500	30.5236	50.28	30.5700	POINT (30.57000 50.28000)	949.2488
9	Кіровоградська	Кропивницький	902.275	48.5103	32.2667	48.36	32.4100	POINT (32.41000 48.36000)	180.4550
10	Луганська	Луганськ	2101.653	48.5833	39.3333	48.42	39.2600	POINT (39.26000 48.42000)	420.3306
11	Львівська	Львів	2476.113	49.8419	24.0315	49.66	24.0900	POINT (24.09000 49.66000)	495.2226
12	Миколаївська	Миколаїв	1090.492	46.9667	32.0000	46.89	31.9700	POINT (31.97000 46.89000)	218.0984
13	Одеська	Одеса	2349.749	46.4775	30.7326	46.38	30.7400	POINT (30.74000 46.38000)	469.9498

Рис. 4.1 – Завантаження даних про кількість населення в регіонах України

Дані ми завантажили за допомогою пакету geopandas, що є обгорткою над pandas для роботи з географічними даними. Для коректного відображення координат за допомогою сервісу Google Earth було знайдено точні координати крайніх точок України. Вони необхідні для коректного відображення координат на нашій карті. При першій спробі відобразити карту я стикнувся з проблемою, що географічні координати не відповідають точкам на завантаженій карті. Я вручну визначив координати цих точок та додав їх у набір даних і відображатиму бульбашки саме за цими координатами. Для відображення карти використаємо функцію imshow() пакету matplotlib, а для відображення бульбашок - .plot(). Також задамо обмеження координат для карти.

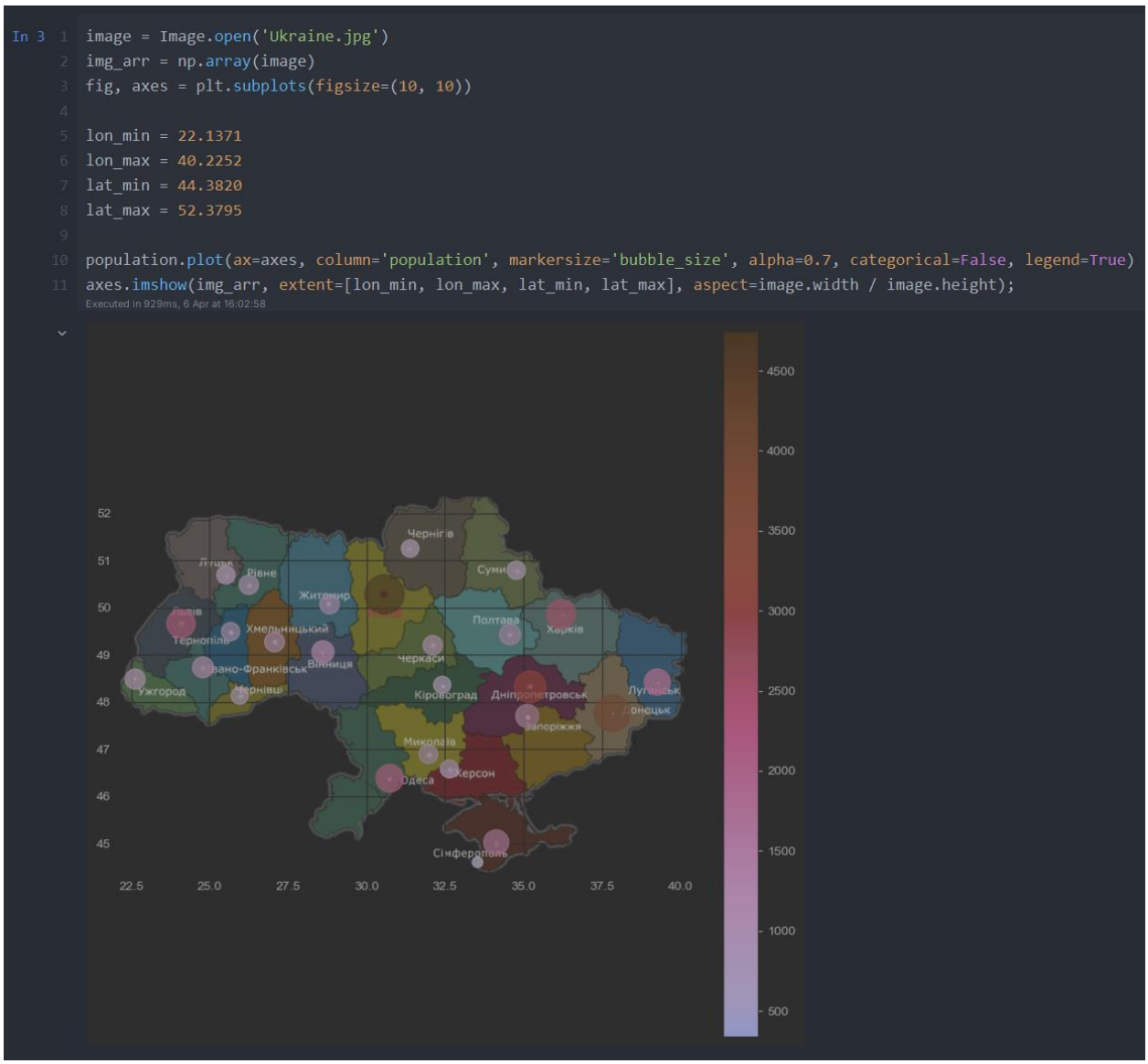

Рис. 4.2 – Зображення даних про кількість населення на мапі

4.3 Знайти найбільшу відстань між містами в пікселях та кілометрах

Для початку було побудовано новий DataFrame, у якому є наступні стовпці: перше та друге міста, відстань у кілометрах та відстань у пікселях. Для розрахунку відстаней у кілометрах можна використати функцію distance пакету geopy, оскільки ця дія є досить складною для обчислення. Розрахунок відстані в пікселях було проведено наступним чином: було обраховано кількість пікселів на 1 градус широти та довготи й ці значення було поділено на різницю відповідних координат кожної з точок.

In 4

```python
cities = []
km_distances = []
pixels_distances = []

pixels_per_lat = image.height / (lat_max - lat_min)
pixels_pet_lon = image.width / (lon_max - lon_min)

for _, city1 in population.iterrows():
    name1 = city1.city
    lat1 = city1.map_latitude
    lon1 = city1.map_longitude
    px_lat1 = pixels_per_lat * (lat1 - lat_min)
    px_lon1 = pixels_pet_lon * (lon1 - lon_min)
    for _, city2 in population.iterrows():
        name2 = city2.city
        lat2 = city2.map_latitude
        lon2 = city2.map_longitude
        px_lat2 = pixels_per_lat * (lat2 - lat_min)
        px_lon2 = pixels_pet_lon * (lon2 - lon_min)
        cities.append((name1, name2))
        km_distances.append(distance.distance((lat1, lon1), (lat2, lon2)).km)
        pixels_distances.append(math.sqrt((px_lat2 - px_lat1) ** 2 + (px_lon2 - px_lon1) ** 2))

distances = pd.DataFrame.from_dict({
    'origin': list(zip(*cities))[0],
    'destination': list(zip(*cities))[1],
    'km_distance': km_distances,
    'pixels_distances': pixels_distances
})

distances
```

Executed in 304ms, 6 Apr at 16:03:00

Out 4

1-36 676 rows × 4 columns pd.Dataframe

	origin	destination	km_distance	pixels_distances
0	Вінниця	Вінниця	0.000000	0.000000
1	Вінниця	Луцьк	286.669844	181.262258
2	Вінниця	Дніпр	492.676679	307.503028
3	Вінниця	Донецьк	697.243431	433.016474
4	Вінниця	Житомир	113.355306	70.501321
5	Вінниця	Ужгород	444.164478	277.643817
6	Вінниця	Запоріжжя	505.199121	313.517063
7	Вінниця	Івано-Франківськ	282.466080	176.974920
8	Вінниця	Київ	196.022148	123.388368
9	Вінниця	Кропивницький	290.258916	181.164552
10	Вінниця	Луганськ	785.898953	491.277477

Рис. 4.3 – Обрахунок відстаней між містами

Для отримання найбільших відстаней між містами було використано методи .loc та idxmax()

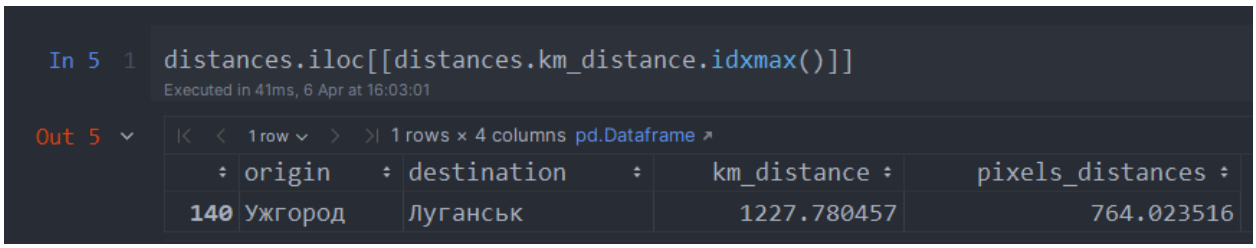

Рис. 4.4 – Пара міст із найбільшою відстанню в кілометрах

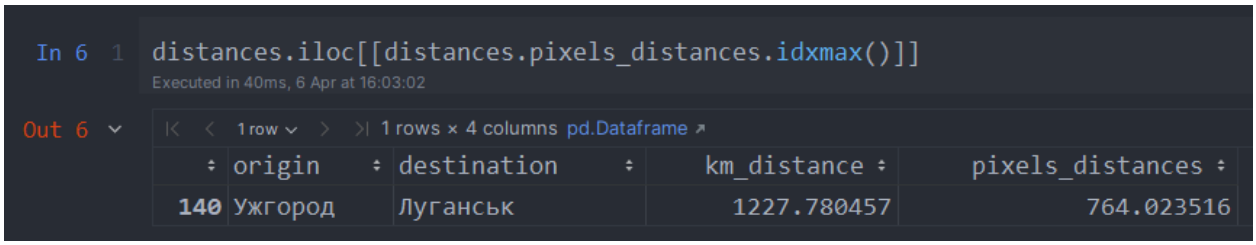

Рис. 4.5 – Пара міст із найбільшою відстанню в пікселях

5 ВИКОНАННЯ ДРУГОГО ДОДАТКОВОГО ЗАВДАННЯ

5.1 Завантажити shape-файл с областями України.

Для роботи з shape-файлом було використано пакет geopandas.

```
In 7  1  ukr_regions = gpd.read_file('UKR_ADM1.shp')
         Executed in 546ms, 6 Apr at 16:03:04
```

Рис. 5.1 – Завантаження мапи

5.2 Побудувати картограми для прибутку населення на 1 особу і ВВП по регіонам за 2016 рік.

Для початку завантажимо дані про ВВП за регіонами.

```
In 8  1  ukr_gdp = pd.read_csv('ukr_GDP.csv', encoding='cp1251', skiprows=1, delimiter=';', index_col='Name')
      2  ukr_gdp
```

Out 8 — 27 rows × 12 columns pd.Dataframe

Name	UKRname	2006	2007	2008	2009	2010	2011	2012	2013	2014	2015	2016
Autonomous Republic of Crimea	Автономна Республіка Крим	NaN	NaN	NaN	NaN	NaN	NaN	44536	46393	NaN	NaN	NaN
Vinnytsia Oblast	Вінницька	NaN	NaN	NaN	NaN	NaN	NaN	33024	36191	43990.0	59871.0	74411.0
Volyn Oblast	Волинська	NaN	NaN	NaN	NaN	NaN	NaN	20005	20622	24195.0	31688.0	35744.0
Dnipropetrovsk Oblast	Дніпропетровська	NaN	NaN	NaN	NaN	NaN	NaN	147970	152905	176540.0	215206.0	244478.0
Donetsk Oblast	Донецька	NaN	NaN	NaN	NaN	NaN	NaN	170775	164926	119983.0	115012.0	137500.0
Zhytomyr Oblast	Житомирська	NaN	NaN	NaN	NaN	NaN	NaN	24849	25676	29815.0	38425.0	47919.0
Zakarpattia Oblast	Закарпатська	NaN	NaN	NaN	NaN	NaN	NaN	21404	21400	24120.0	28952.0	32390.0
Zaporizhia Oblast	Запорізька	NaN	NaN	NaN	NaN	NaN	NaN	54828	54352	65968.0	89061.0	104323.0
Ivano-Frankivsk Oblast	Івано-Франківська	NaN	NaN	NaN	NaN	NaN	NaN	32286	33196	37643.0	45854.0	51404.0
Kyiv Oblast	Київська	NaN	NaN	NaN	NaN	NaN	NaN	69663	68931	79561.0	104030.0	128638.0
Kirovohrad Oblast	Кіровоградська	NaN	NaN	NaN	NaN	NaN	NaN	22056	25313	28758.0	38447.0	46021.0
Luhansk Oblast	Луганська	NaN	NaN	NaN	NaN	NaN	NaN	58767	55108	31393.0	23849.0	31356.0
Lviv Oblast	Львівська	NaN	NaN	NaN	NaN	NaN	NaN	61962	63329	72923.0	94690.0	114842.0

Рис. 5.2 – Завантаження даних про ВВП регіонів України

Бачимо, що є пропущені значення. Для їх заповнення використаємо поліноміальну регресію.

```
In 157  1  def fill_regression(row: pd.Series, dtype: type = None):
        2      row = row.iloc[1:]
        3      row.index = row.index.astype(int)
        4      notna_cols = row[row.notna()]
        5      na_cols = row[row.isna()]
        6      years = notna_cols.index - 2006
        7      values = notna_cols.values
        8      degree = 2
        9      model = np.poly1d(np.polyfit(years ** degree, values.astype(int), degree))
       10      for index, num in na_cols.items():
       11          row[index] = model(index - 2006)
       12      if dtype:
       13          row = row.astype(dtype)
       14      return row
```

Рис. 5.3 – Побудова регресійної моделі

```
In 158  ukr_gdp = ukr_gdp.apply(fill_regression, axis=1)
        ukr_gdp

        C:\ProgramData\Anaconda3\lib\site-packages\pandas\core\apply.py:873: RankWarning: Polyfit may be poorly conditioned\n  …

Out 158  1-11 > >| 27 rows × 11 columns pd.Dataframe ↗                                                                                CSV
```

Name	2006	2007	2008	2009	2010	2011	2012	2013	2014	2015	2016
Autonomous Republic of Crimea	35283.003577	35621.589488	35955.514927	36284.779897	36609.384396	36929.328425	44536	46393	37861.197688	38162.499835	38459.141
Vinnytsia Oblast	28516.130204	28418.794778	28332.866864	28258.346463	28195.233574	28143.528198	33024	36191	43990.000000	59871.000000	74411.000
Volyn Oblast	15667.728951	15708.254676	15752.155515	15799.431468	15850.082534	15904.108715	20005	20622	24195.000000	31688.000000	35744.000
Dnipropetrovsk Oblast	123314.284164	123538.077386	123782.406390	124047.271176	124332.671745	124638.608095	147970	152905	176540.000000	215206.000000	244478.000
Donetsk Oblast	320222.339952	315159.780316	310161.966682	305228.899051	300360.577422	295557.001796	170775	164926	119983.000000	115012.000000	137500.000
Zhytomyr Oblast	27183.992119	26948.923735	26722.874382	26505.844061	26297.832772	26098.840514	24849	25676	29815.000000	38425.000000	47919.000
Zakarpattia Oblast	19905.346031	19875.814166	19849.486164	19826.362023	19806.441743	19789.725325	21404	21400	24120.000000	28952.000000	32390.000
Zaporizhia Oblast	49916.847913	49708.576517	49515.929274	49338.906184	49177.507246	49031.732461	54828	54352	65968.000000	89061.000000	104323.000
Ivano-Frankivsk Oblast	27575.305756	27610.655278	27650.237327	27694.051905	27742.099011	27794.378646	32286	33196	37643.000000	45854.000000	51404.000
Kyiv Oblast	82066.839556	81193.191874	80346.916198	79528.012528	78736.480864	77972.321207	69663	68931	79561.000000	104030.000000	128638.000
Kirovohrad Oblast	17139.634228	17181.740286	17228.924751	17281.187623	17338.528903	17400.948591	22056	25313	28758.000000	38447.000000	46021.000

Рис. 5.4 – Заповнення пропущених значень

Далі об’єднаємо набір даних, створений на базі shape-файлу із набором даних про ВВП.

```
In 163  regions_gdp = pd.merge(ukr_regions, ukr_gdp, left_on='Name', right_index=True)
        regions_gdp

Out 163  1-11 > >| 27 rows × 20 columns pd.Dataframe ↗                                                                              CSV
```

	ISO_Code	Level	Name	adm	adm_int	feature_id	gbid	iso	geometry	2006	200
0	UA-65	ADM1	Kherson Oblast	ADM1	1	0	UKR_ADM1_1_3_3_0	UKR	POLYGON ((35.46760 46.14516, 35.46262 46.13371...	19260.406124	19138.246
1	UA-07	ADM1	Volyn Oblast	ADM1	1	1	UKR_ADM1_1_3_3_1	UKR	POLYGON ((26.10729 51.00529, 26.08816 51.00426...	15667.728951	15708.254
2	UA-56	ADM1	Rivne Oblast	ADM1	1	2	UKR_ADM1_1_3_3_2	UKR	POLYGON ((27.73464 51.59371, 27.73370 51.59002...	11860.903064	12082.303
3	UA-18	ADM1	Zhytomyr Oblast	ADM1	1	3	UKR_ADM1_1_3_3_3	UKR	POLYGON ((29.73521 49.94438, 29.72536 49.94179...	27183.992119	26948.923
4	UA-32	ADM1	Kyiv Oblast	ADM1	1	4	UKR_ADM1_1_3_3_4	UKR	MULTIPOLYGON (((30.34907 50.48887, 30.34805 50...	82066.839556	81193.191
5	UA-74	ADM1	Chernihiv Oblast	ADM1	1	5	UKR_ADM1_1_3_3_5	UKR	POLYGON ((33.50072 52.07412, 33.50067 52.06984...	23000.388244	22891.955
6	UA-59	ADM1	Sumy Oblast	ADM1	1	6	UKR_ADM1_1_3_3_6	UKR	POLYGON ((35.69266 50.34563, 35.68866 50.33474...	18038.377579	18138.891
7	UA-63	ADM1	Kharkiv Oblast	ADM1	1	7	UKR_ADM1_1_3_3_7	UKR	POLYGON ((38.09361 49.84606, 38.08983 49.84373...	92334.465986	91501.195
8	UA-09	ADM1	Luhansk Oblast	ADM1	1	8	UKR_ADM1_1_3_3_8	UKR	POLYGON ((40.22758 49.26053, 40.22442 49.25486...	132972.257249	130529.711
9	UA-14	ADM1	Donetsk Oblast	ADM1	1	9	UKR_ADM1_1_3_3_9	UKR	POLYGON ((39.09144 47.94099, 39.09062 47.93890...	320222.339952	315159.780
10	UA-23	ADM1	Zaporizhia Oblast	ADM1	1	10	UKR_ADM1_1_3_3_10	UKR	POLYGON ((37.24863 47.45723, 37.24842 47.45682...	49916.847913	49708.576

Рис. 5.5 – Об’єднаний DataFrame

Побудуємо картограму за допомогою методу .plot() нашого GeoDataFrame.

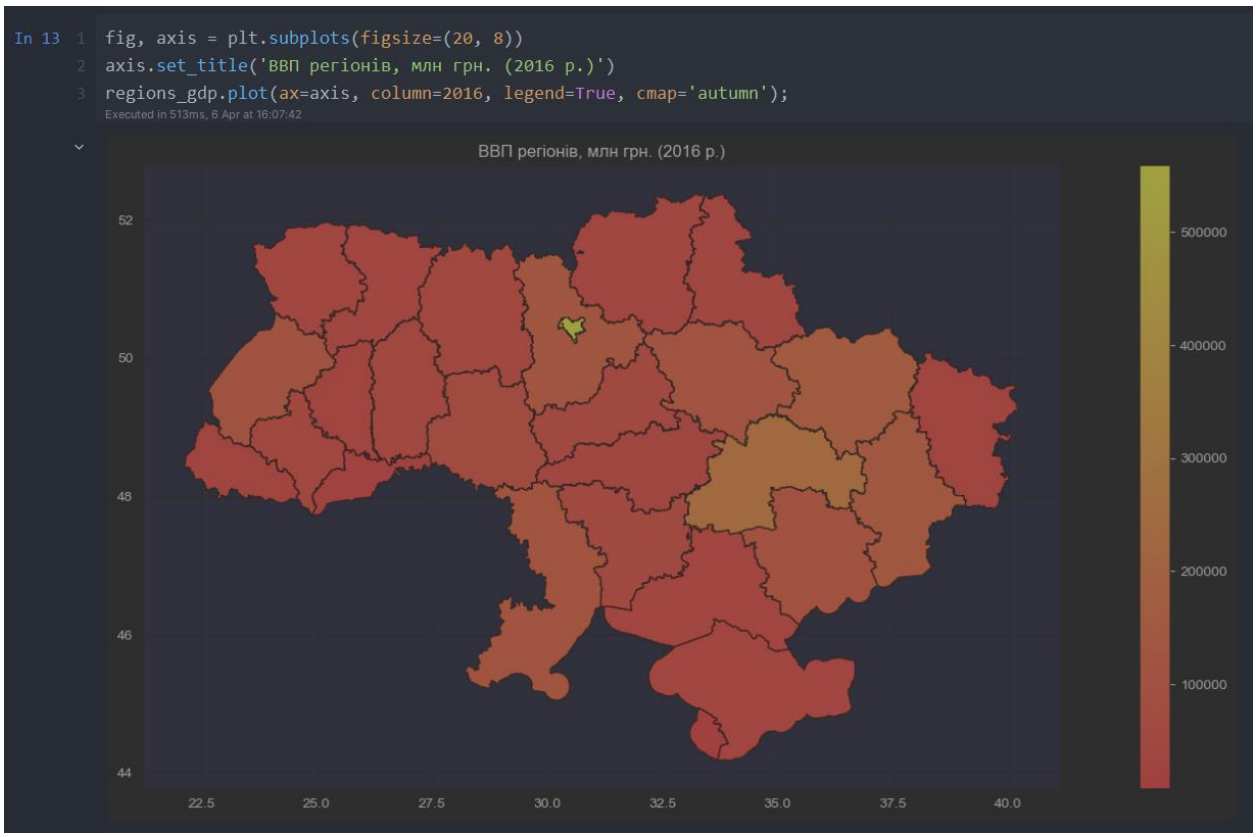

Рис. 5.6 – Картограма ВВП регіонів України в 2016 році

Як бачимо, найвищий ВВП має Київ.

Проведемо всі вищеописані дії для набору даних зарплат.

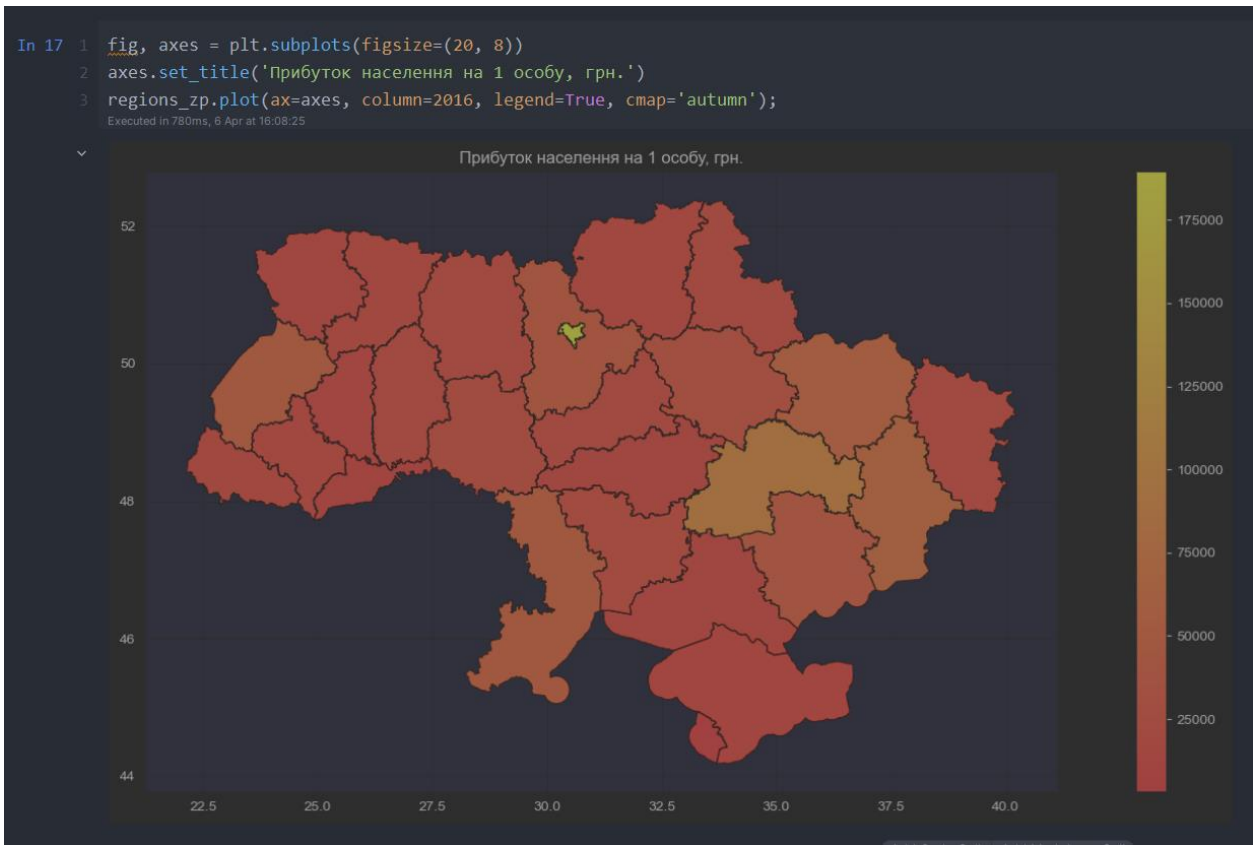

Рис. 5.7 – Картограма прибутку населення України в 2016 році

Як бачимо, найвищі зарплати знову в Києві.

5.3 По даним за 2006-2015 роки для кожного регіону розрахувати коефіцієнт кореляції між прибутком населення на 1 особу та ВВП. Відобразити на картограмі.

Для обрахунку кореляції використаємо метод DataFrame .corrwith(), указавши необхідні роки. Виведемо дані про кореляцію.

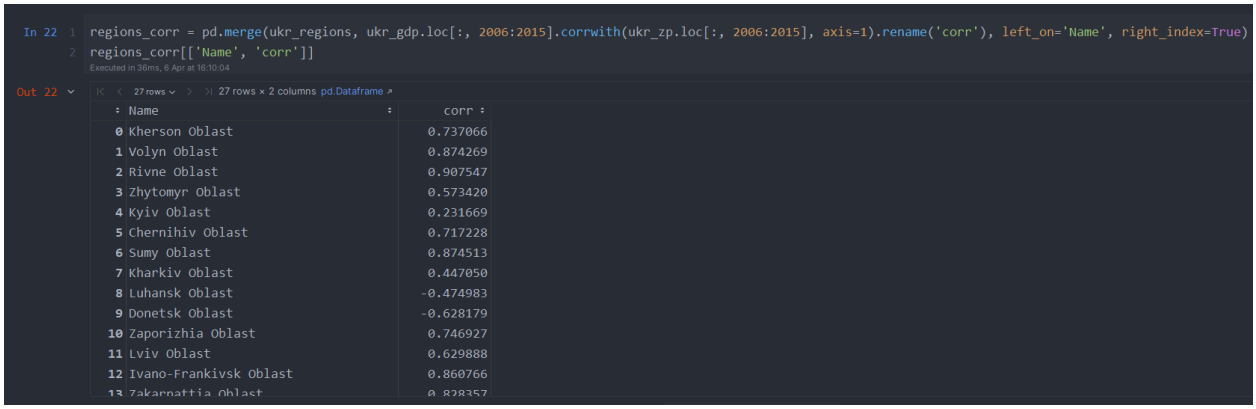

Рис. 5.8 – Обрахунок коефіцієнтів кореляції

Далі побудуємо картограму за допомогою вже згаданого методу .plot().

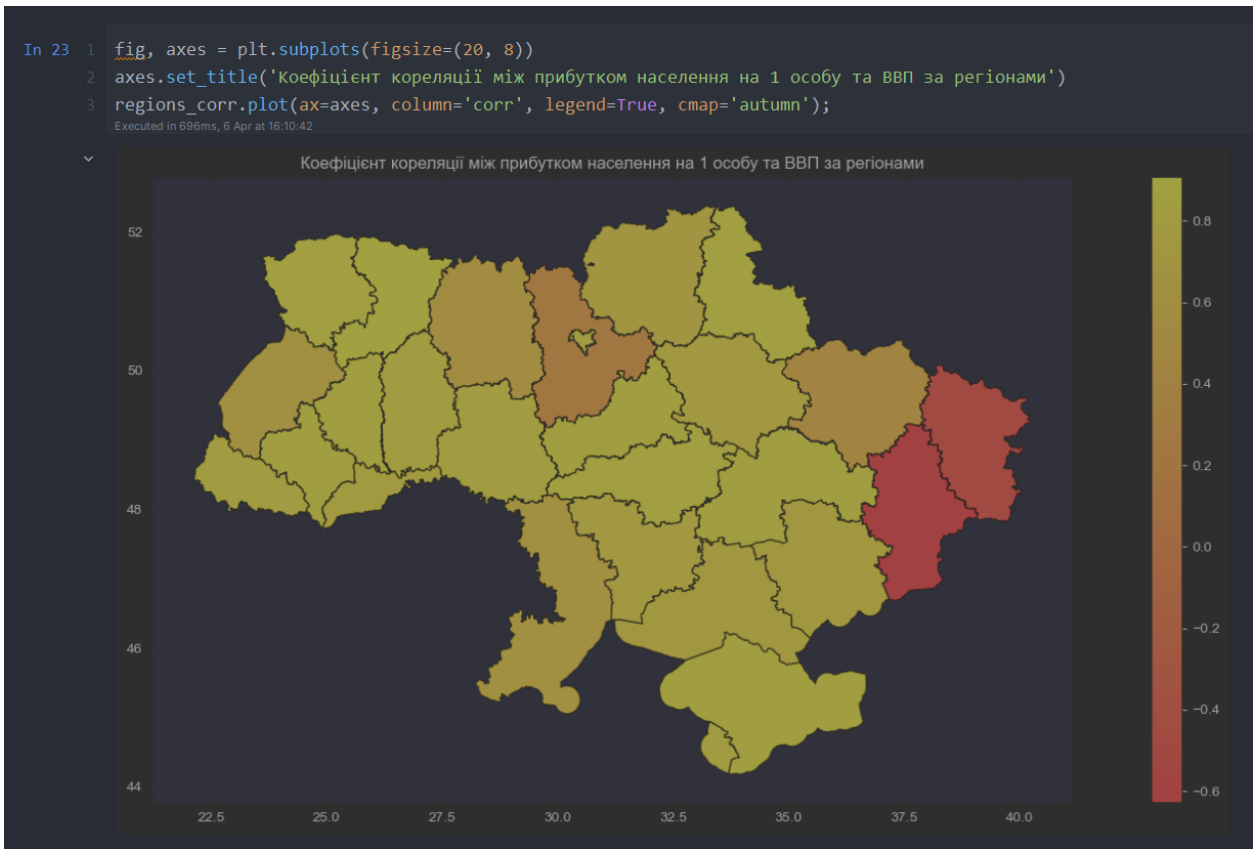

Рис. 5.9 – Відображення даних про кореляцію між прибутком населення та ВВП в регіонах України.

Можемо побачити, що всюди ВВП корелює з зарплатами, окрім Донецької та Луганської області. Спричинено це тим, що по цих регіонах відсутня достатня кількість інформації.

6 ВИСНОВОК

У ході даної лабораторної роботи було досліджено набір даних, що містить інформацію про країни світу. Усі стовпці було перевірено на нормальність розподілу, встановлено, що жоден зі стовпців не розподілений нормально, після цього було перевірено гіпотезу про рівність середнього та медіану, яку знову ж таки було відкинуто для всіх стовпців. Також за допомогою теста Андерсона-Дарлінга було встановлено, що найближчий до нормального розподіл викидів CO2 має регіон Південна Азія. Після проведення цих перевірок було побудовано кругову діаграму, що відображає кількість населення за регіонами.

У наступній частині лабораторної роботи було досліджено кількість населення в регіонах України та відображено JPG картограму з такими даними. Побачили, що найбільше населення в Києві. Також обрахували відстані між містами в кілометрах та пікселях і отримали, що найбільша відстань між Ужгородом та Луганськом – 1227 кілометрів або 764 пікселя.

Після цього було завантажено дані про ВВП та прибуток населення в регіонах України, заповнено пропущені значення за допомогою поліноміальної регресії другого порядку, після чого відображено картограми цих показників за 2016 рік (найвищі вони в Києві), а потім обраховано кореляцію між ВВП та прибутком населення й отримано, що всюди, окрім двох регіонів ці дані корелюють.