POLITECHNIKA WARSZAWSKA

PRZESTRZENNE BAZY DANYCH

PROJEKT

„Zbadanie możliwości przekształceń danych przestrzennych w środowisku Python”

Autor:

Jakub Kuczmarski

Nr indeksu: 297300

Prowadzący:

dr inż. Grzegorz M. Protaziuk

Warszawa 2023/24

1. Temat projektu

Tematem projektu jest zbadanie możliwości przekształceń danych przestrzennych w środowisku Python.

2. Cel Projektu

Celem projektu jest analiza dostępnych w środowisku Python funkcji do przekształcania danych przestrzennych takich jak:

1. upraszczania danych,
2. wyznaczania środków oraz buforów,
3. transformacji afiniczne,
4. przycinania obiektów,
5. tworzenia unii geometrycznych,
6. metod transformacji typów,
7. sąsiedztwo, odległość i przecięcia obiektów.

3. Informacje projektowe

Środowisko wykonawcze: Google Colaboratory
Język: Python
Biblioteki: Pandas, NumPy, GeoPandas, Random, Time, Folium, Matplotlib, Mapclassify, Contextily, Shapely, GeoPy

4. Wstęp teoretyczny

Dane przestrzenne

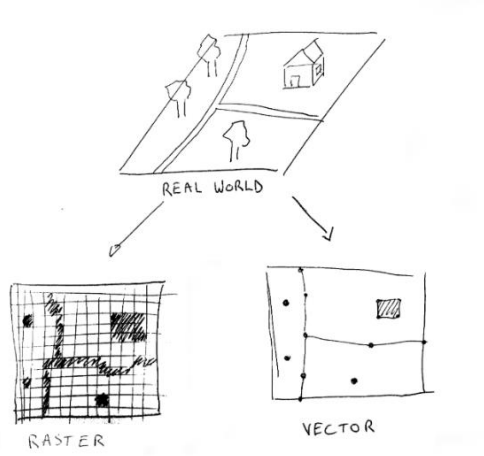
Dane przestrzenne odnoszą się do danych, które są powiązane z określonymi lokalizacjami geograficznymi lub pozycjami w przestrzeni fizycznej. Zawierają informacje o kształcie, rozmiarze i względnym rozmieszczeniu obiektów na danym obszarze. Dane przestrzenne są zazwyczaj reprezentowane za pomocą współrzędnych, takich jak szerokość i długość geograficzna lub system oparty na siatce. Każdy z obiekt przestrzenny/geograficzny składa się z dwóch komponentów:

- **opisu** - zbioru atrybutów alfanumerycznych, typowych dla systemów baz danych, np. nazwa i populacja miasta.
- **komponentu przestrzennego**, który obejmuje:
    - geometrię (lokalizację w przestrzeni geograficznej lub innej przestrzeni, kształt itp.),
    - topologię (relacje przestrzenne istniejące między obiektami, jak styczność).

Istnieją dwa podstawowe typy danych przestrzennych (rys. 1):

- **dane wektorowe** reprezentują cechy przestrzenne jako punkty, linie i wielokąty. Punkty reprezentują określone lokalizacje, takie jak współrzędne geograficzne położenia miasta w przestrzeni, linie zaś reprezentują cechy liniowe, takie jak drogi lub rzeki, a wielokąty zamknięte obszary, takie jak kraje, budynki lub jeziora. Dane wektorowe mogą przechowywać dodatkowe atrybuty lub atrybuty związane z każdą cechą, takie jak populacja miasta lub nazwa budynku.
- **dane rastrowe** dzielą przestrzeń na siatkę komórek lub pikseli, gdzie każda komórka reprezentuje określony obszar. Każda komórka w siatce zawiera wartość, która reprezentuje charakterystykę lub atrybut tej lokalizacji, taki jak temperatura, wysokość lub użytkowanie terenu. Dane rastrowe są często wykorzystywane w teledetekcji, obrazowaniu satelitarnym i analizie terenu.

Dane przestrzenne są szeroko stosowane w różnych dziedzinach, w tym w geografii, urbanistyce, naukach o środowisku, transporcie, rolnictwie i wielu innych. Umożliwiają analizę, wizualizację i modelowanie zjawisk geograficznych, pomagając zrozumieć wzorce, relacje i trendy w świecie fizycznym.



Rys. 1 Dane rastowe i wektorowe.

System Odniesienia Przestrzennego

CRS (Coordinate Reference System) to system odniesienia współrzędnych w danych przestrzennych. Określa on sposób, w jaki współrzędne przestrzenne są zdefiniowane i interpretowane w kontekście konkretnej przestrzeni geograficznej.

CRS obejmuje informacje dotyczące jednostek miary, układu osi, odniesienia geodezyjnego i transformacji współrzędnych. To właśnie CRS zapewnia dokładne odwzorowanie danych przestrzennych na rzeczywistą powierzchnię Ziemi.

Ważne elementy CRS to:

- Jednostki miary: Określa jednostki, w których są wyrażane współrzędne przestrzenne, na przykład stopnie, metry, stopy.

- Układ osi: Określa, które osie reprezentują współrzędne poziome (x, y) i pionowe (z) oraz ich orientację. Na przykład, dla współrzędnych geograficznych, osie zazwyczaj reprezentują szerokość (x), długość (y) i wysokość (z).
- Odniesienie geodezyjne: Określa model matematyczny Ziemi, na podstawie którego są obliczane współrzędne. Może to być elipsoida lub geoida.
- Transformacje współrzędnych: Określa metody transformacji między różnymi CRS, gdy dane przestrzenne muszą być dostosowane do innego systemu odniesienia lub układu osi.

Przy korzystaniu z danych przestrzennych, zrozumienie i poprawne określenie CRS jest niezbędne do interpretacji i analizy danych geograficznych. To umożliwia jednoznaczne ustalenie położenia obiektów na Ziemi i umożliwia integrację danych z różnych źródeł w spójny sposób.

Upraszczanie danych

Funkcja upraszczania danych pozwala na zmniejszenie liczby wierzchołków w obiektach przestrzennych co pozwala na zmniejszenie rozmiaru plików i przyspieszenie operacji w Pythonie. Do upraszczania danych przestrzennych można użyć funkcji simplify() z biblioteki shapely

Zastosowanie: analiza danych i uproszczenie geometrii za pomocą simplify() może pomóc w analizie danych ponieważ mniejsza ilość danych oznacza krótszy czas przetwarzania i mniejszą ilość pamięci potrzebną do przechowywania danych

Wyznaczanie środków oraz buforów

Funkcje te pozwalają na wyznaczanie środków geometrycznych obiektów przestrzennych oraz tworzenie stref buforowych wokół nich. W Pythonie można użyć odpowiednio funkcji centroid() - odpowiada za wyznaczenie środków geometrycznych badanej geometrii oraz buffer() z biblioteki shapely

Zastosowanie:

Wizualizacja danych: Centroid obiektu przestrzennego może być wykorzystany jako punkt odniesienia do umieszczenia etykiet lub symboli wizualizacyjnych na mapie.

Planowanie przestrzenne: Tworzenie buforów wokół obiektów przestrzennych może pomóc w planowaniu przestrzennym, na przykład podczas określenia stref ochronnych wokół źródeł wody, terenów zalewowych lub parków narodowych.

Przycinanie obiektów

Funkcja ta pozwala na przycięcie obiektów przestrzennych do określonego obszaru. W Pythonie można użyć np. funkcji intersection() z biblioteki shapely

Zastosowanie:

Analiza zasięgu działania: Jeśli mamy zdefiniowane obszary działania różnych podmiotów (np. firmy instytucje), funkcja intersection() pozwala na wyznaczenie obszarów na których dwa lub więcej podmiotów mają wspólny zasięg

Transformacje afiniczne

Transformacje afiniczne pozwalają na przekształcanie obiektów przestrzennych za pomocą kombinacji przesunięć, obrotów, skalowania i przekształceń symetrycznych. W Pythonie można użyć funkcji affine_transform() lub określonej transformacji tj. rotate(), scale(), translate() z biblioteki Shapely.

Funkcje:

rotate(): Funkcja ta służy do obracania geometrii obiektów przestrzennych o określony kąt.

Zastosowanie:

Analiza przestrzenna: Obracanie geometrii obiektów przestrzennych może pomóc w analizie ich położenia względem innych obiektów, na przykład w określeniu nachylenia terenu lub orientacji dróg.

scale(): Funkcja ta służy do skalowania geometrii obiektów przestrzennych przez zmianę ich rozmiaru.

Zastosowanie:

Analiza przestrzenna: Skalowanie geometrii obiektów przestrzennych może pomóc w analizie ich wielkości względem innych obiektów, na przykład w określeniu powierzchni zabudowy lub obszarów rolniczych.

translate(): Funkcja ta służy do przesuwania geometrii obiektów przestrzennych o określoną odległość w poziomie i pionie

Zastosowanie:

Analiza przestrzenna: Przesuwanie geometrii obiektów przestrzennych może pomóc w analizie ich położenia względem innych obiektów, na przykład w określeniu odległości między budynkami lub drogami.

Tworzenie unii geometrycznych

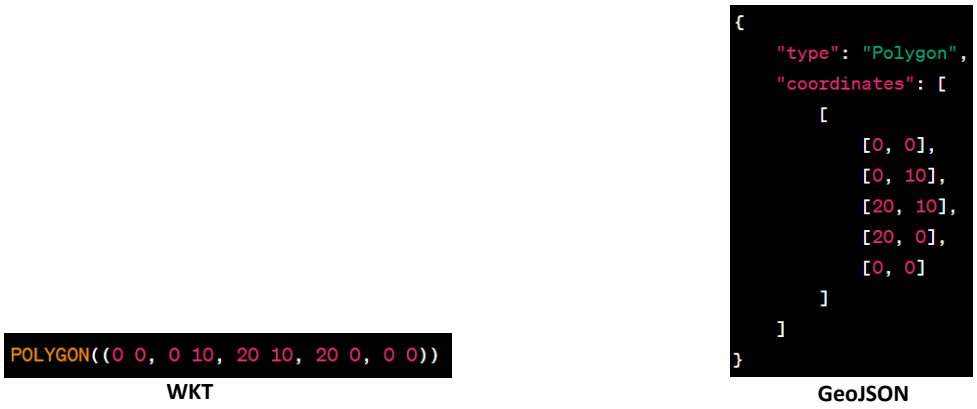
Funkcja ta pozwala na tworzenie unii geometrycznych z kilku obiektów przestrzennych. W Pythonie można użyć funkcji union() lub unary_union() z biblioteki Shapely.

Zastosowanie:

Analiza powierzchni - jeśli mamy wiele obiektów o różnych kształtach i chcemy wyznaczyć powierzchnię całej tej przestrzeni, funkcja union() pozwala na połączenie tych obiektów w jeden, większy obiekt, którego powierzchnię łatwiej obliczyć.

Metody transformacji typów

Metody transformacji typów pozwalają na przekształcanie obiektów przestrzennych pomiędzy różnymi typami reprezentacji, np. pomiędzy formatami WKT i GeoJSON. W Pythonie można użyć funkcji loads() i dumps() z biblioteki Shapely.



Rys. 2 Porównanie formatu WKT i GeoJSON – reprezentacja prostokąta o wymiarach 10 na 20 jednostek

5. Opis zbioru danych

W projekcie wykorzystano następujące zbiory danych:

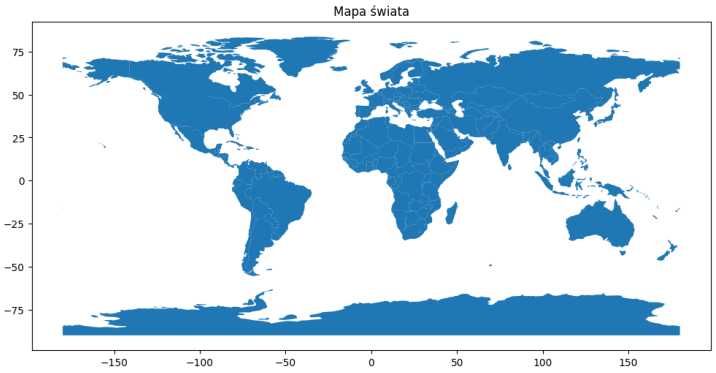
1. Zbiór danych „Mapa Świata”

Zbiór danych „Mapa Świata” zawiera granicę i obszar (poligony i multipologony – obiekty dwuwymiarowe) zajmowany przez kraje zgodnie z ich faktycznym stanem. Informacje te przechowywane są w kolumnie ‘geometry’, która stanowi komponent przestrzenny, zaś pozostałe 5 kolumn (‘iso_a3’ – kod danego kraju, ‘name’ – pełna nazwa kraju, ‘continent’ – kontynent, na którym występuje dany kraj, ‘pop_est’ – szacowana populacja, ‘gpd_md_est’ – szacowana średnia wartość produktu krajowego brutto) zawiera zbiór atrybutów alfanumerycznych służących do opisu tych obiektów geograficznych (rys. 3)

	iso_a3	name	continent	pop_est	gdp_md_est	geometry
0	AFG	Afghanistan	Asia	34124811.0	64080.0	POLYGON ((61.21082 35.65007, 62.23065 35.27066...
1	AGO	Angola	Africa	29310273.0	189000.0	MULTIPOLYGON (((23.90415 -11.72228, 24.07991 -...
2	ALB	Albania	Europe	3047987.0	33900.0	POLYGON ((21.02004 40.84273, 20.99999 40.58000...
3	ARE	United Arab Emirates	Asia	6072475.0	667200.0	POLYGON ((51.57952 24.24550, 51.75744 24.29407...
4	ARG	Argentina	South America	44293293.0	879400.0	MULTIPOLYGON (((-66.95992 -54.89681, -67.56244...

Rys. 3 Przykładowe dane ze zbioru „Mapa świata”

Wizualizacja zbioru danych została zaprezentowana na rys. 4.



Rys. 4 Wizualizacja zbioru danych „Mapa świata”.

System odniesienia przestrzennego dla danego zbioru danych został ukazany na rys. 5.

```
<Geographic 2D CRS: EPSG:4326>
Name: WGS 84
Axis Info [ellipsoidal]:
- Lat[north]: Geodetic latitude (degree)
- Lon[east]: Geodetic longitude (degree)
Area of Use:
- name: World.
- bounds: (-180.0, -90.0, 180.0, 90.0)
Datum: World Geodetic System 1984 ensemble
- Ellipsoid: WGS 84
- Prime Meridian: Greenwich
```

Rys. 5 CRS dla zbioru danych „Mapa świata”

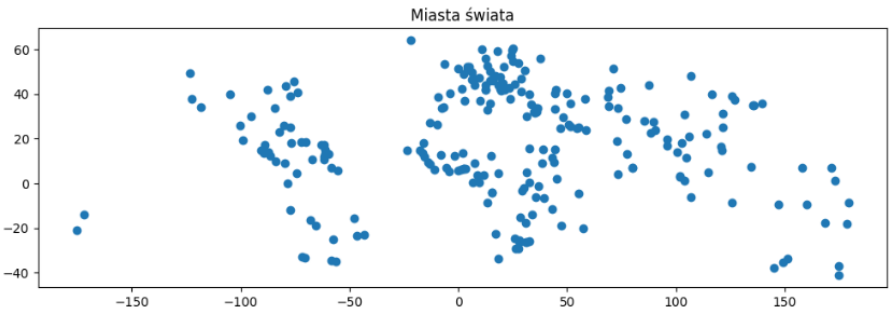
2. Zbiór danych „Miasta Świata”

Zbiór danych „Miasta Świata” zawiera punkty (obiekty zero-wymiarowe) w rozpatrywanej przestrzeni z atrybutami nazw poszczególnych miast. Informacje o geometrii zawiera kolumna ‘geometry’, zaś nazwa danego miasta znajduje się w kolumnie ‘name’ . Przykładowe dane z tego zbioru prezentuje rys. 6.

	name	geometry
0	Vatican City	POINT (12.45339 41.90328)
1	San Marino	POINT (12.44177 43.93610)
2	Vaduz	POINT (9.51667 47.13372)
3	Lobamba	POINT (31.20000 -26.46667)
4	Luxembourg	POINT (6.13000 49.61166)

Rys. 6 Przykładowe dane ze zbioru „Miasta świata”

Wizualizacja zbioru danych została zaprezentowana na rys. 7.



Rys. 7 Wizualizacja zbioru danych „Mapa świata”.

System odniesienia przestrzennego dla danego zbioru danych został ukazany na rys. 8

```
<Geographic 2D CRS: EPSG:4326>
Name: WGS 84
Axis Info [ellipsoidal]:
- Lat[north]: Geodetic latitude (degree)
- Lon[east]: Geodetic longitude (degree)
Area of Use:
- name: World.
- bounds: (-180.0, -90.0, 180.0, 90.0)
Datum: World Geodetic System 1984 ensemble
- Ellipsoid: WGS 84
- Prime Meridian: Greenwich
```

Rys. 8 CRS dla zbioru danych „Miasta świata”

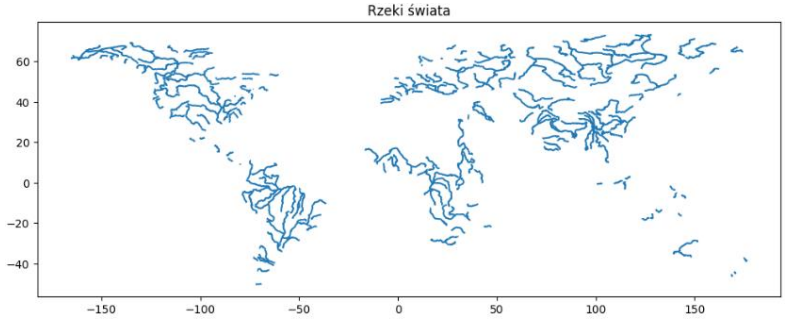
3. Zbiór danych „Rzeki i jeziora na świecie”

Zbiór danych „Rzeki i jeziora na świecie” zawiera informacje na temat geometrii rzecz i jezior, które występują w (przepływają przez) danym kraju. Dane geometryczne są w postaci łamanych (polilini) i znajdują się w kolumnie ‘geometry’. Informacje o nazwie danej rzeki znajdują się w kolumnie ‘name’, zaś informacja, czy mamy do czynienia z rzeką, czy jeziorem w kolumnie ‘featurecla’. Przykładowe dane z tego zbioru prezentuje rys. 9.

	featurecla	name	geometry
0	Lake Centerline	Kama	LINESTRING (51.93713 55.70107, 51.88087 55.686...
1	River	Kama	LINESTRING (53.69385 58.20632, 53.67715 58.273...
2	Lake Centerline	Abay	LINESTRING (37.11301 11.85499, 37.15037 11.893...
3	Lake Centerline	Al Furat	LINESTRING (38.56119 35.86264, 38.36534 35.903...
4	Lake Centerline	Alabama	MULTILINESTRING ((-86.52177 33.03212, -86.5209...

Rys. 9 Przykładowe dane ze zbioru „Rzeki i jeziora na świecie”

Wizualizacja zbioru danych została zaprezentowana na rys. 10.



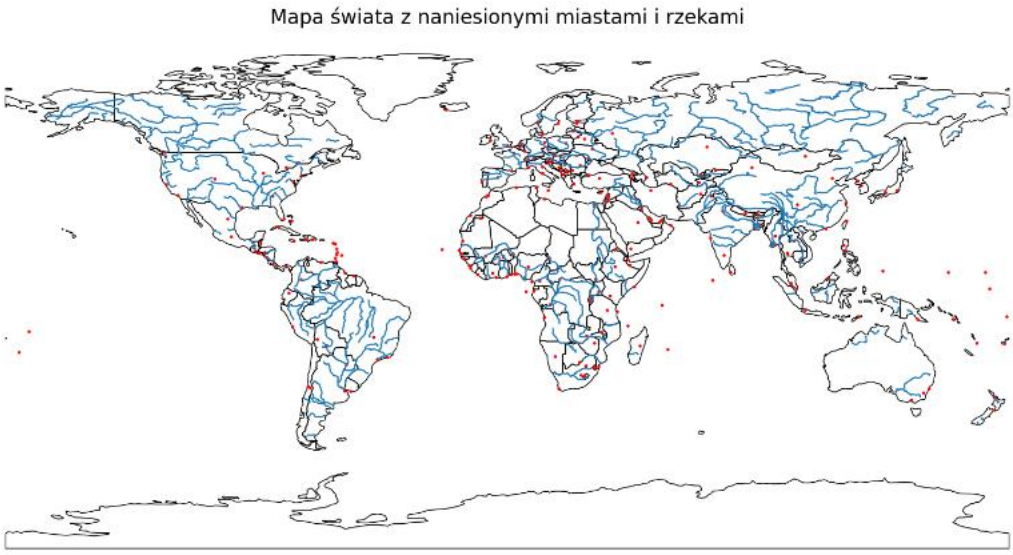
Rys. 10 Wizualizacja zbioru danych „Rzeki i jeziora na świecie”.

System odniesienia przestrzennego dla danego zbioru danych został ukazany na rys. 11

```
<Geographic 2D CRS: EPSG:4326>
Name: WGS 84
Axis Info [ellipsoidal]:
- Lat[north]: Geodetic latitude (degree)
- Lon[east]: Geodetic longitude (degree)
Area of Use:
- name: World.
- bounds: (-180.0, -90.0, 180.0, 90.0)
Datum: World Geodetic System 1984 ensemble
- Ellipsoid: WGS 84
- Prime Meridian: Greenwich
```

Rys. 11 CRS dla zbioru danych „Rzeki i jeziora na świecie”

Wizualizacja wszystkich powyższych trzech zbiorów danych mapy świata na jednym wykresie została ukazana na rys. 12



Rys. 12 Mapa świata z naniesionymi miastami i rzekami

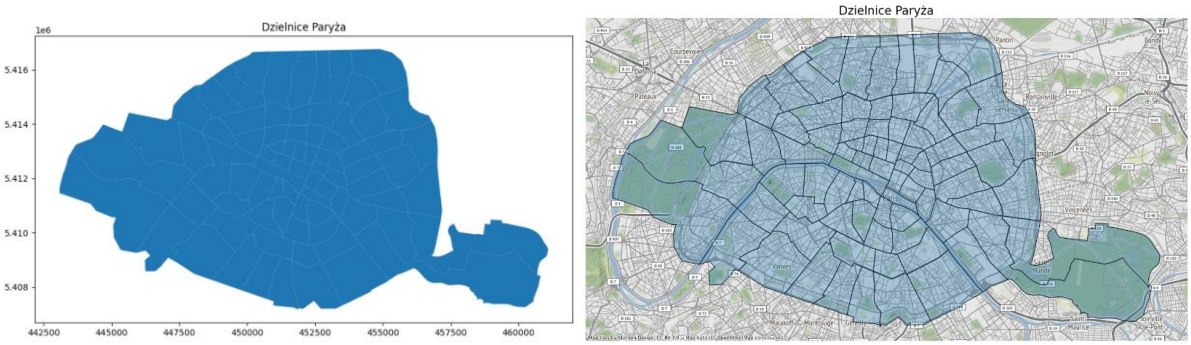
4. Zbiór danych „Dzielnice Paryża”

Zbiór danych „Dzielnice Paryża” zawiera informacje na temat podziału powierzchni Paryża na dzielnice. Zbiór ten zawiera takie informacje jak numer identyfikacyjny danej dzielnucy – kolumna ‘id’, nazwa danej dzielnicy – ‘district_name’, populacja danej dzielnicy – ‘population’ oraz obiekty przestrzenne są w postaci poligonów w kolumnie ‘geometry’. Przykładowe dane z tego zbioru prezentuje rys. 13.

	featurecla	name	geometry
0	Lake Centerline	Kama	LINESTRING (51.93713 55.70107, 51.88087 55.686...
1	River	Kama	LINESTRING (53.69385 58.20632, 53.67715 58.273...
2	Lake Centerline	Abay	LINESTRING (37.11301 11.85499, 37.15037 11.893...
3	Lake Centerline	Al Furat	LINESTRING (38.56119 35.86264, 38.36534 35.903...
4	Lake Centerline	Alabama	MULTILINESTRING ((-86.52177 33.03212, -86.5209...

Rys. 13 Przykładowe dane ze zbioru „Dzielnice Paryża”

Wizualizacja zbioru danych została zaprezentowana na rysunku 14.



Rys. 14 Wizualizacje zbioru danych „Dzielnice Paryża”.

System odniesienia przestrzennego dla danego zbioru danych został ukazany na rys. 15

```
<Projected CRS: EPSG:32631>
Name: WGS 84 / UTM zone 31N
Axis Info [cartesian]:
- E[east]: Easting (metre)
- N[north]: Northing (metre)
Area of Use:
- name: Between 0°E and 6°E, northern hemisphere between equator and 84°N, onshore and offshore. Algeria.
Andorra. Belgium. Benin. Burkina Faso. Denmark - North Sea. France. Germany - North Sea. Ghana. Luxembourg.
Mali. Netherlands. Niger. Nigeria. Norway. Spain. Togo. United Kingdom (UK) - North Sea.
- bounds: (0.0, 0.0, 6.0, 84.0)
Coordinate Operation:
- name: UTM zone 31N
- method: Transverse Mercator
Datum: World Geodetic System 1984 ensemble
- Ellipsoid: WGS 84
- Prime Meridian: Greenwich
```

Rys. 15 CRS dla zbioru danych „Dzielnice Paryża”

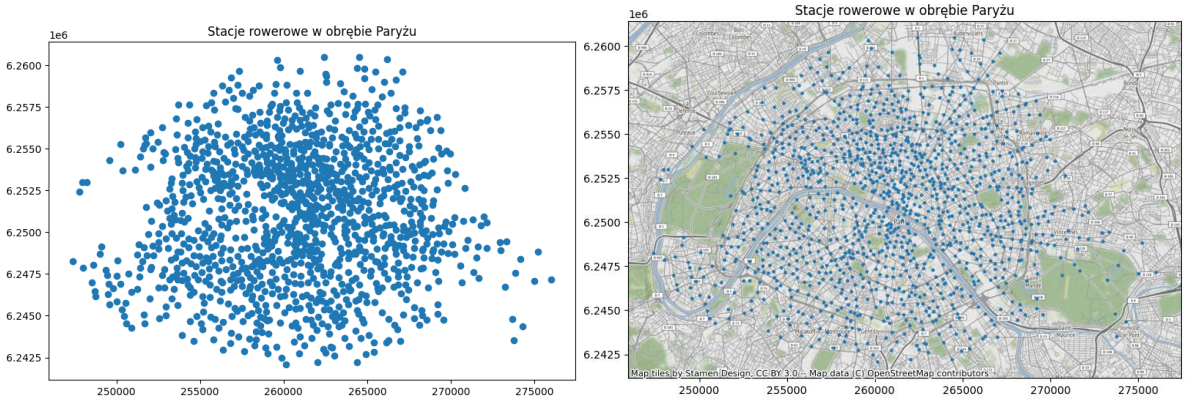
5. Zbiór danych „Stacje rowerowe w Paryżu”

Zbiór danych „Stacje rowerowe w Paryżu” zawiera informacje na temat rozmieszczenia stacji rowerowych w obrębie Paryża i jego na jego obrzeżach. Zbiór ten zawiera takie informacje jak nazwa stacji – kolumna ‘name’, liczba stanowisk na rowery – ‘bike_stands’, liczba dostępnych rowerów na stacji (w chwili zbierania danych) – ‘available_bikes’ oraz położenie stacji na mapie, które znajduje się w kolumnie ‘geometry’ i jest postaci pojedynczego punktu. Przykładowe dane z tego zbioru prezentuje rys. 16.

	name	bike_stands	available_bikes	geometry
0	14002 - RASPAIL QUINET	44	4	POINT (259324.887 6247620.771)
1	20503 - COURS DE VINCENNES PYRÉNÉES	21	3	POINT (267824.377 6249062.894)
2	20011 - PYRÉNÉES-DAGORNO	21	0	POINT (267742.135 6250378.469)
3	31008 - VINCENNES (MONTREUIL)	56	0	POINT (271326.638 6250750.824)
4	43006 - MINIMES (VINCENNES)	28	27	POINT (270594.689 6248007.705)

Rys. 16 Przykładowe dane ze zbioru „Stacje rowerowe w Paryżu”

Wizualizacja zbioru danych została zaprezentowana na rysunku 17.



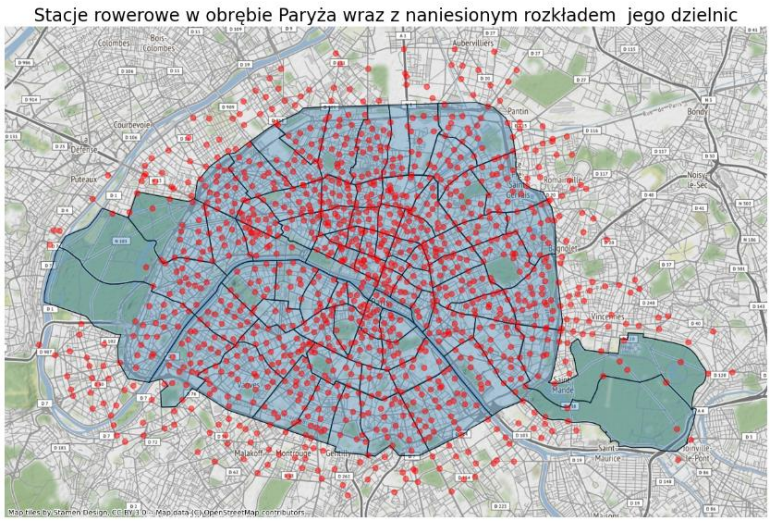
Rys. 17 Wizualizacja zbioru danych „Stacje rowerowe w Paryżu”.

System odniesienia przestrzennego dla danego zbioru danych został ukazany na rys. 18

```
<Projected CRS: EPSG:3857>
Name: WGS 84 / Pseudo-Mercator
Axis Info [cartesian]:
- X[east]: Easting (metre)
- Y[north]: Northing (metre)
Area of Use:
- name: World between 85.06°S and 85.06°N.
- bounds: (-180.0, -85.06, 180.0, 85.06)
Coordinate Operation:
- name: Popular Visualisation Pseudo-Mercator
- method: Popular Visualisation Pseudo Mercator
Datum: World Geodetic System 1984 ensemble
- Ellipsoid: WGS 84
- Prime Meridian: Greenwich
```

Rys. 18 CRS dla zbioru danych „Stacje rowerowe w Paryżu”

Wizualizacja obu powyższych dwóch zbiorów danych dotyczących Paryża na jednym wykresie została ukazana na rys. 19



Rys. 19 Stacje rowerowe w obrębie Paryża wraz z naniesionym rozkładem jego dzielnic.

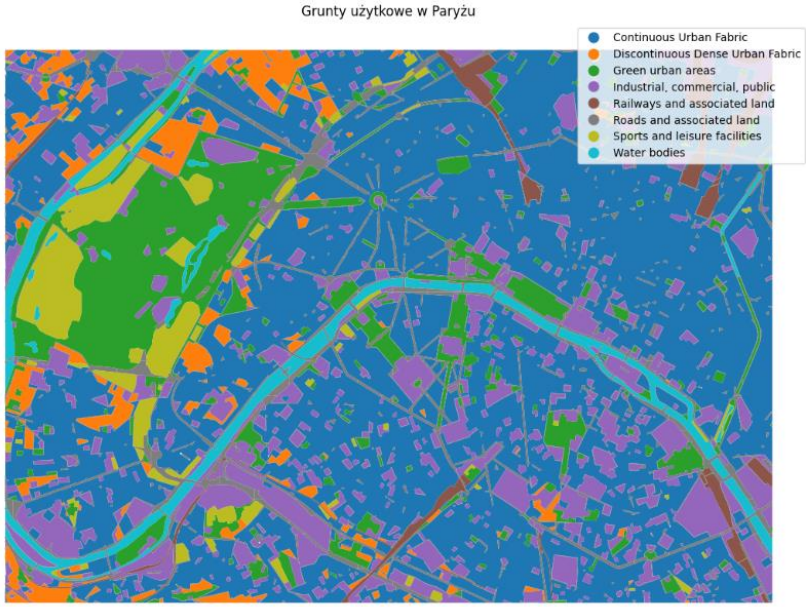
6. Zbiór danych „Tereny użytkowe w Paryżu”

Zbiór danych „Tereny użytkowe w Paryżu” zawiera informacje na temat rodzaju działalności do jakiej wykorzystywany jest określony obszar (np. obszar miejski lub rekreacyjny). Jest to zestaw danych wielokątów (kolumna ‘geometry’) z etykietą reprezentującą klasę użytkowania gruntów dla różnych obszarów Paryża (kolumna ‘class’). Przykładowe dane z tego zbioru prezentuje rys. 20.

	class	geometry
0	Water bodies	POLYGON ((3751386.281 2890064.323, 3751395.345...
1	Roads and associated land	POLYGON ((3751390.345 2886000.000, 3751390.345...
2	Roads and associated land	POLYGON ((3751390.345 2886898.192, 3751390.370...
3	Roads and associated land	POLYGON ((3751390.345 2887500.000, 3751390.345...
4	Roads and associated land	POLYGON ((3751390.345 2888647.357, 3751390.370...

Rys. 20 Przykładowe dane ze zbioru „Tereny użytkowe w Paryżu”

Wizualizacja zbioru danych została zaprezentowana na rys. 21.



Rys. 21 Wizualizacja zbioru danych „Tereny użytkowe w Paryżu”.

System odniesienia przestrzennego dla danego zbioru danych został ukazany na rys. 22.

```
<Projected CRS: EPSG:3857>
Name: WGS 84 / Pseudo-Mercator
Axis Info [cartesian]:
- X[east]: Easting (metre)
- Y[north]: Northing (metre)
Area of Use:
- name: World between 85.06°S and 85.06°N.
- bounds: (-180.0, -85.06, 180.0, 85.06)
Coordinate Operation:
- name: Popular Visualisation Pseudo-Mercator
- method: Popular Visualisation Pseudo Mercator
Datum: World Geodetic System 1984 ensemble
- Ellipsoid: WGS 84
- Prime Meridian: Greenwich
```

Rys. 22 CRS dla zbioru danych „Tereny użytkowe w Paryżu”

7. Własnoręcznie stworzone liniowe obiekty przestrzenne tj. łamana.

Zestawienie informacji o zbiorach danych

Tabela 1 . Zestawienie najważniejszych parametrów zbiorów danych

Nazwa zbioru	Nazwa pliku	Liczba wartości	Liczba kolumn	CRS
Mapa świata	ne_110m_admin_0_countries.zip	177	6	ESPG: 4326 WGS 84
Miasta świata	ne_110m_populated_places.zip	243	2	ESPG: 4326 WGS 84
Rzeki i jeziora na świecie	ne_50m_rivers_lake_centerlines.zip	461	3	ESPG: 4326 WGS 84
Dzielnice Paryża	paris_districts.geojson	80	4	ESPG: 32631 WGS 84 / UTM zone 31N
Stacje rowerowe w Paryżu	paris_bike_stations.gpkg	1226	4	ESPG: 3857 WGS 84 / Pseudo-Mercator
Tereny użytkowe w Paryżu	paris_land_use.zip	3243	2	PROJCS["Lambert_Azimuthal_Equal_Area",GEOGCS["GCS_... Lambert_Azimuthal_Equal_Area

8. Przekształcenia danych przestrzennych

Upraszczanie danych

Do realizacji tej części projektowej została wykorzystana funkcja simplify() z biblioteki Shapely, która pozwala na uproszczenie geometrii linii (LineString) lub wielokąta (Polygon) zachowując przy tym ogólny kształt oryginalnej geometrii poprzez redukcję zbędnych punktów. Parametr 'tolerance' jest odpowiedzialny za kontrolę stopnia uproszczenia poprzez określenie maksymalnej odległość między pierwotną geometrią, a uproszczoną. Im jego wartość większa tym większe zastosowane uproszczenie.

Wykorzystane zbiory danych: własnoręcznie zdefiniowane obiekty przestrzenne – łamane.

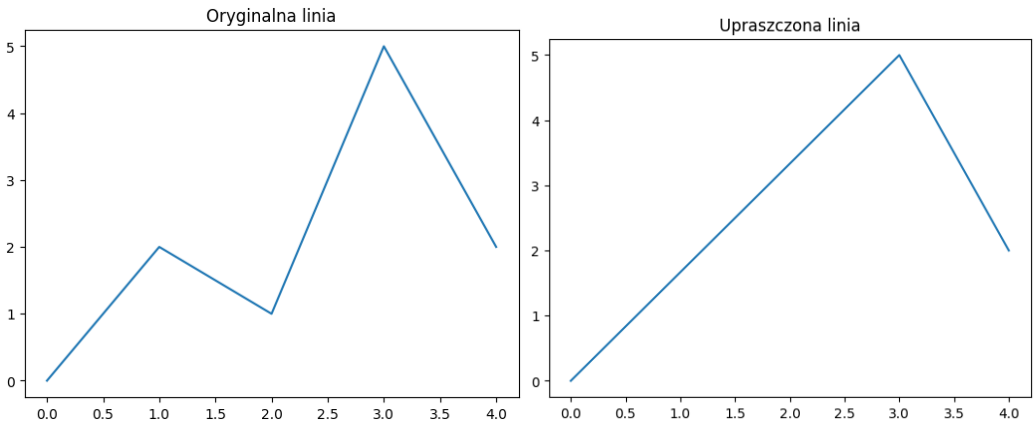
W tej części projektowej przeprowadzono następujące eksperymenty:

Eksperyment 1

Składał się z dwóch przykładów prezentujących podstawowe działanie funkcji simplify().

Przykład 1 – różnica długości łamanej przed i po uproszczeniu

W tym przykładzie zdefiniowano arbitralnie łamaną o punktach [(0, 0) , (1, 2) , (2, 1) , (3, 5) , (4, 2)], a następnie dokonano jej uproszczenia dla parametru 'tolerance' = 2.0 (ryz. 23).



Rys. 23 Uproszczenie łamanej przy pomocy funkcji simplify() o tolerancji równej 2.

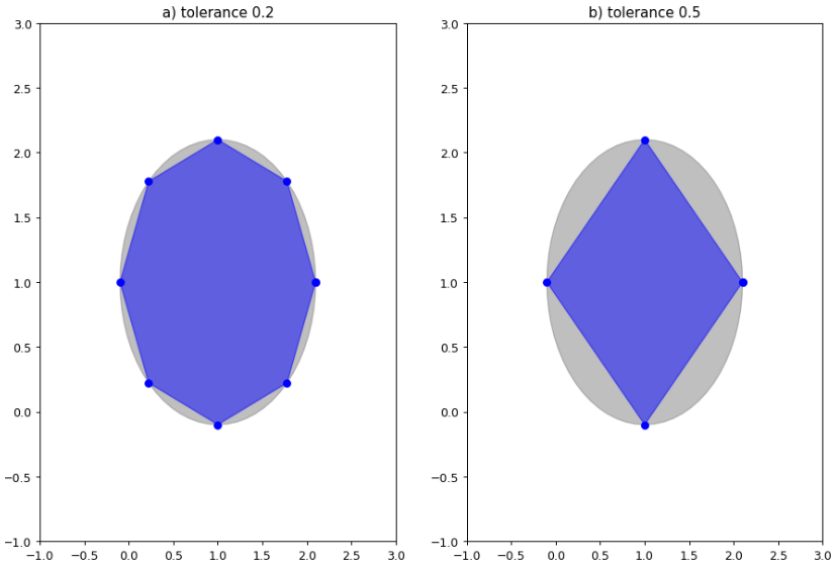
Dodatkowo zbadano zmianę długości łamanej przed i po uproszeniu. Otrzymane wyniki zawiera Tabela 2.

Tabela 2. Długość łamanej przed i po uproszczeniu

Upraszczanie	Punkty łamanej	Długość łamanej
Nie	(0 0, 1 2, 2 1, 3 5, 4 2)	(0 0, 3 5, 4 2)
Tak	10.935664825658925	8.99322955501368

Przykład 2 – różnica powierzchni wieloboku przed i po uproszczeniu

W tym przykładzie zdefiniowano arbitralnie bufor dookoła punktu o współrzędnych (1,1) – kolor szary na rys. 24, a następnie dokonano jego uproszczenia, kolejno dla wartości parametru ‘tolerance’ równej 0.2, a następnie 0.5.



Rys. 24 Wpływ wielkości parametru tolerancji na wielkość uproszczenia.

Dodatkowo zbadano zmianę powierzchni przed i po każdym z uproszeń. Otrzymane wyniki prezentuje tabela 3.

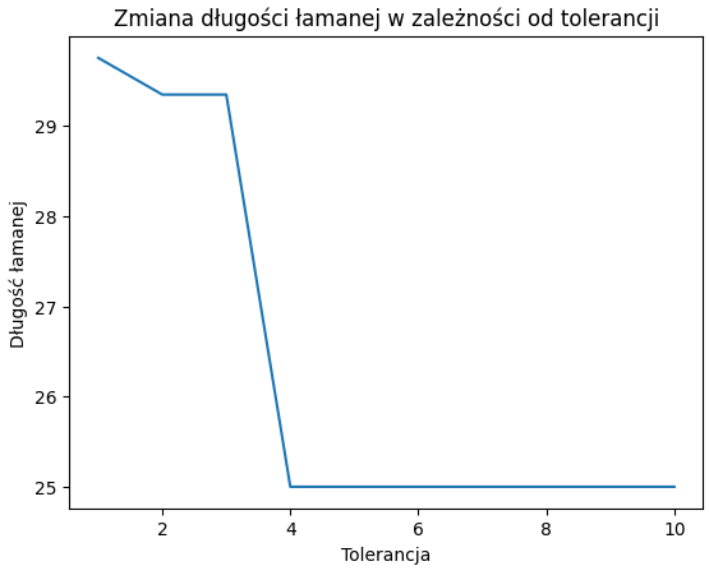
Tabela 3. Powierzchnia wielokąta przed i po uproszczeniu

Upraszczanie	Tolerancja	Powierzchnia
Nie	-	3.7952236735605873
Tak	0.2	3.4223968209428905
	0.5	2.4200000000000004

Eksperyment 2

Eksperyment drugi polegał na zbadaniu wpływ parametru 'tolerance' na długość upraszczanej łamanej. Kształt łamanej w tej części eksperymentalnej została zdefiniowana z góry i nie ulegał zmianie do końca trwania eksperymentu. Stworzona łamaną zawierała następujące punkty: [(0, 0), (1, 5) , (0, 10) , (-2, 15) , (4, 20) , (0, 25)]. Testy przeprowadzono dla tolerancji w przedziale wartości od 1 do 11.

Otrzymane wyniki zostały przedstawione na rys. 25

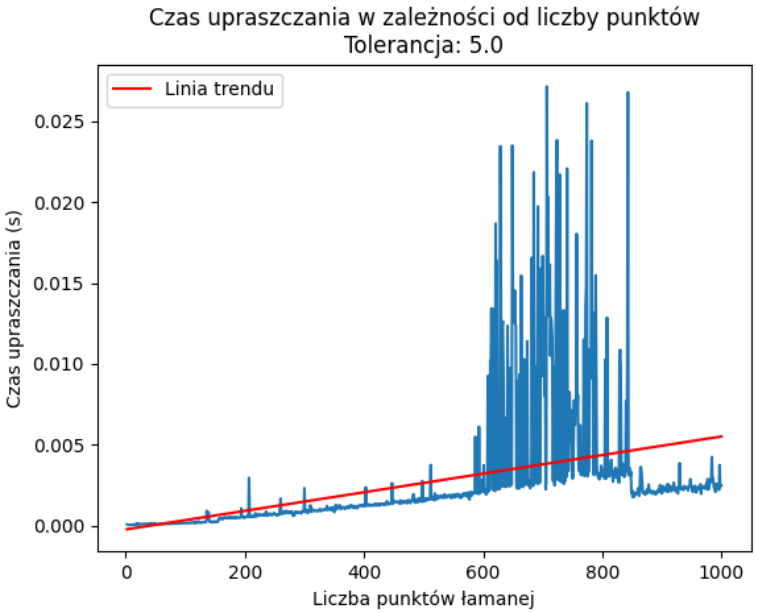


Rys. 25 Wpływ tolerancji na długość krzywej.

Eksperyment 3

W tej części badany był czas upraszczania w zależności od liczby punktów należących do łamanej. Przyjęta do testów wartość tolerancji wynosiła 5, zaś liczba punktów tworzących łamaną zmieniła się w przedziale [2, 1000]. Wartości współrzędnych na osi x zawierały się w przedziale [0, aktualna liczba punktów tworzących łamaną-1], zaś na osi y były wybierane w sposób losowy z przedziału [-5,5]. Wizualizacja otrzymanych wyników została przedstawiona na rys. 26. Posiada on również linię trendu

(kolor czerwony) informującą jaki trend miała zmiana długości czasu upraszczania łamanej w zależności od liczy tworzących ją punktów.



Rys. 26 Wpływ liczy punktów łamanej na czas jej upraszczania.

Wnioski

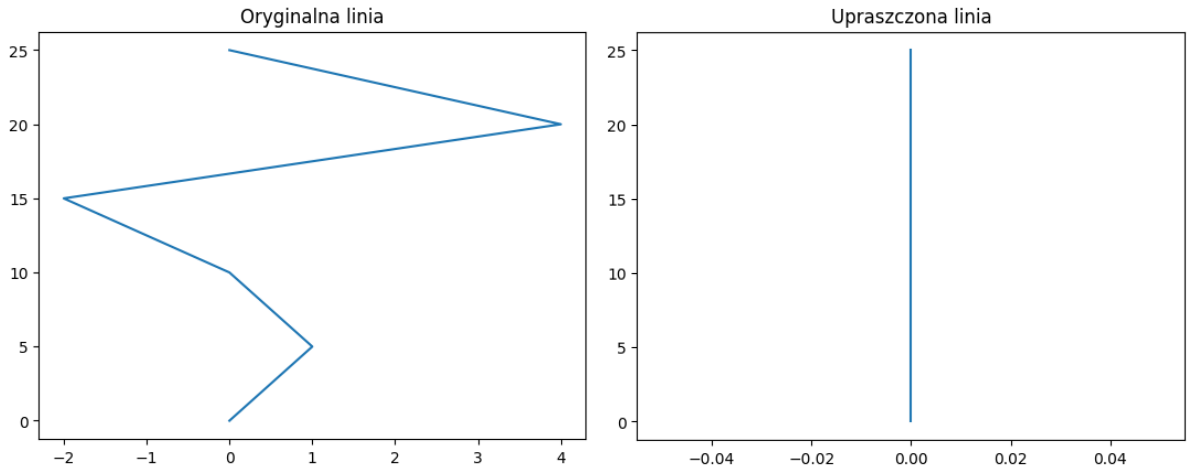
Na bazie wiedzy wyciągniętej podczas eksperymentów i analizy otrzymanych wykresów można dość do następujących wniosków:

- im większa wartość parametru ‘tolerance’ tym długość łamanej jest mniejsza. Zmian długości zachodzi do momentu, gdy łamana zawiera jedynie dwa punkty ja tworzące, dlatego też dalszy wzrost wartości parametry tolerancji jest bezcelowy.

W naszym przypadku wartość graniczna tolerancji upraszczania wynosi 4 - rys. 27.

- wraz ze wzrostem liczby punktów składających się na łamaną, czas potrzebny na jej uproszczenie również wzrasta.

W naszym przypadku widoczne są znaczne wzrosty czasu potrzebnego na wykonanie niezbędnych operacji przez funkcje simplify() dla liczby punktów z przedziału [600,820]. Wynika to z faktu, że wartości punktów względem osi y są wybierane w sposób losowy, dlatego, też kształty powstałych łamanych różni się między sobą (jedna może być bardziej skomplikowana od drugiej), co wpływa na czas potrzebny na jej uproszczenie.



Rys. 27 Graniczna wartość toleracji równa 4.

Wyznaczanie środków oraz buforów

Do realizacji tej części projektowej została wykorzystana funkcja centroid() oraz buffer() z biblioteki GeoPandas. Pierwsza z nich określa położenie środka geometrycznego danego obiektu przestrzennego, druga zaś pozwala stworzyć strefę buforową wokół obiektu.

Wykorzystane zbiory danych: „Dzielnice Paryża”, „Mapa świata”, własnoręcznie zdefiniowana geometria rzeki Sekwany.

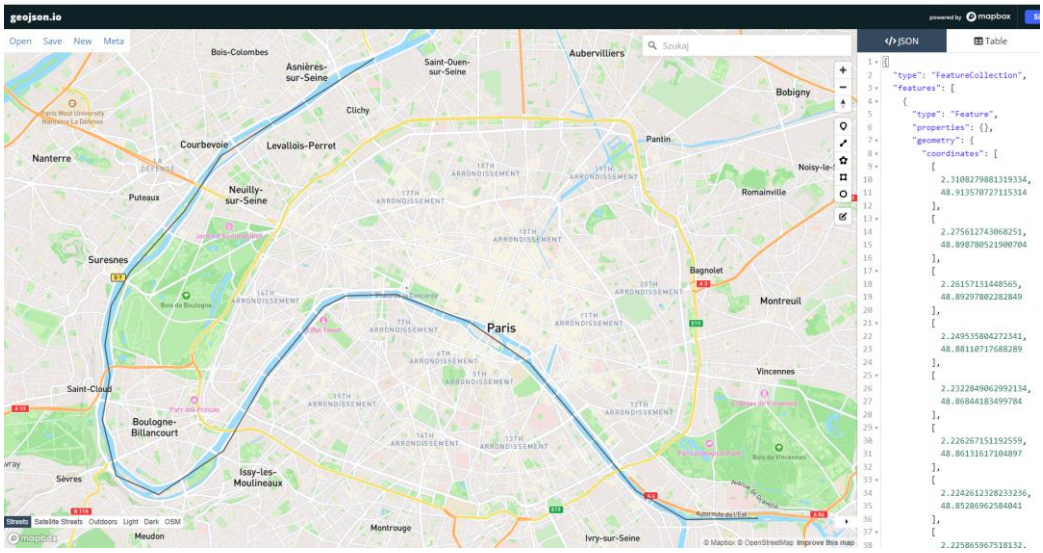
W tej części projektowej przeprowadzono następujące eksperymenty:

Bufor

Eksperyment 1

Polegał na sprawdzeniu, które z dzielnic Paryża mogą być zagrożone w sytuacji wzrostu poziomu wody w Sekwanie. W tej części wykorzystano własnoręcznie zdefiniowaną geometrię rzeki Sekwany oraz zbiór danych zawierający dzielnicę Paryża.

Pierwszy etap eksperymentu polegał na zdefiniowaniu geometrii Sekwany. W tym celu skorzystano ze strony internetowej geojson.io, która pozwala na ręczne określenie kształtu danego obiektu geometrycznego, a następnie na tej podstawie generowany jest koda w formacie GeoJSON. Efekt działania strony prezentuje rys. 28.

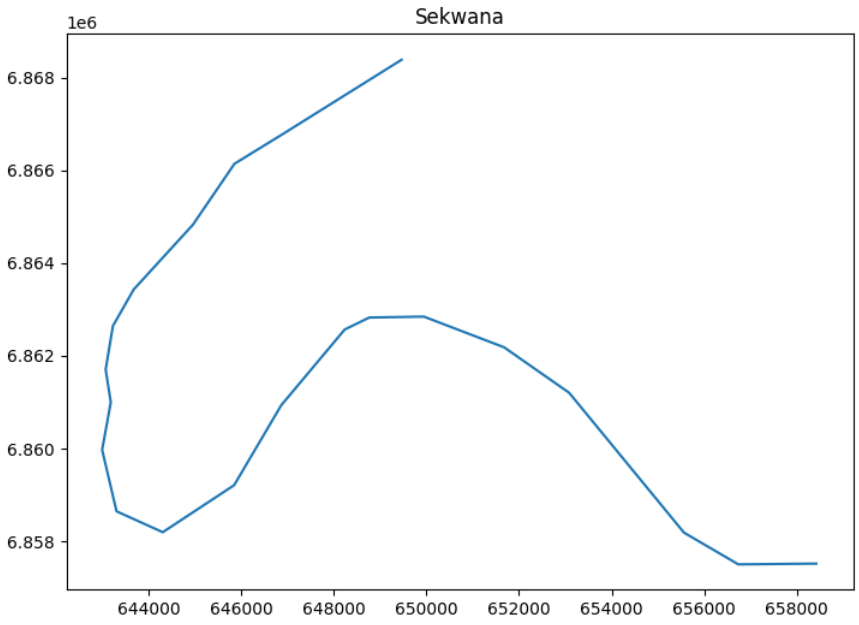


Rys. 28 Zdefiniowana geometria Sekwany przy pomocy strony geojson.io.

Otrzymana geometria prezentuje się następująco – rys. 29:


```
{
 "type": "FeatureCollection",
 "features": [
  {
   "type": "Feature",
   "properties": {},
   "geometry": {
    "coordinates": [
     [ 2.3108279881319334, 48.913570727115314],
     [2.275612743068251, 48.898780521900704],
     [2.26157131448565, 48.89297802282849],
     [2.249535804272341, 48.88110717688289],
     [2.2322849062992134, 48.86844183499784],
     [2.226267151192559, 48.86131617104897],
     [2.2242612328233236, 48.85286962584041],
     [2.225865967518132, 48.846533781610134],
     [2.223458865475436, 48.83729257163333],
     [2.227871885887339, 48.825408510413496],
     [2.2415121307964228, 48.821446530265405],
```

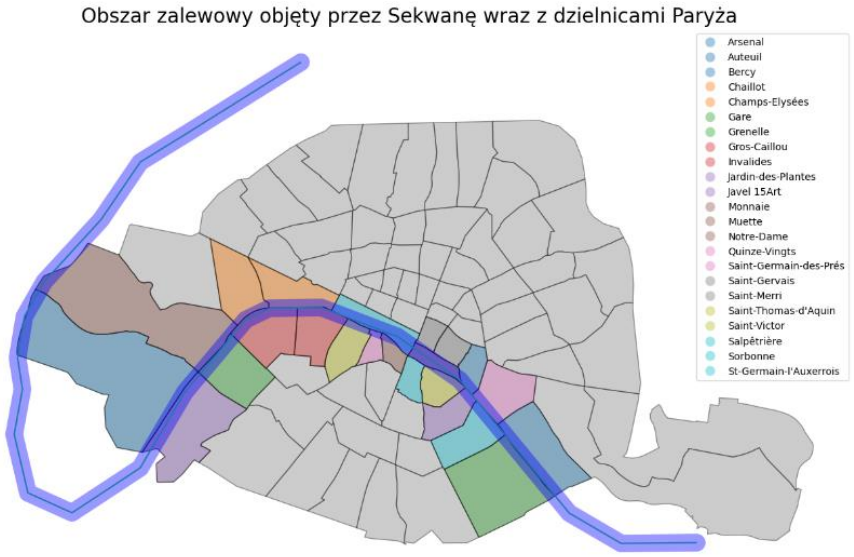
```
        [2.2623736818335374, 48.83069066339962],
        [2.2760139267426496, 48.84626977070218],
        [2.294468375737125, 48.861052238084625],
        [2.301689681865099, 48.86342758466097],
        [2.317737028816879, 48.86369150509984],
        [2.341406865570093, 48.85788493394887],
        [2.3606636819124276, 48.849173814137345],
        [2.3947642941842275, 48.82223895135195],
        [2.4108116411360356, 48.816163402845206],
        [2.4340802942157893, 48.81642757244077]],
      "type": "LineString"
    }
   }
 ]
}
```

Rys. 29 Wizualizacja wygenerowanego kodu w formacie GeoJSON.

Następnie został zdefiniowany bufor określający w jakim otoczeniu wokół rzeki może dojść do powodzi (w projekcie przyjęto, że będzie to 200 m).

Na koniec sprawdzono jaki obszar, które z dzielnic Paryża są narażone w przypadku gwałtownego wzrostu poziomu wody w Sekwanie. W tym celu wykorzystano funkcję intersects() z biblioteki Shaply, która sprawdza czy dane dwa obiekty geometryczne maja wspólne punkty lub czy jedne z obiektów przecina drugi. W efekcie otrzymana została wizualizacja z zaznaczonymi dzielnicami oraz legendą z ich nazwami (rys. 30).



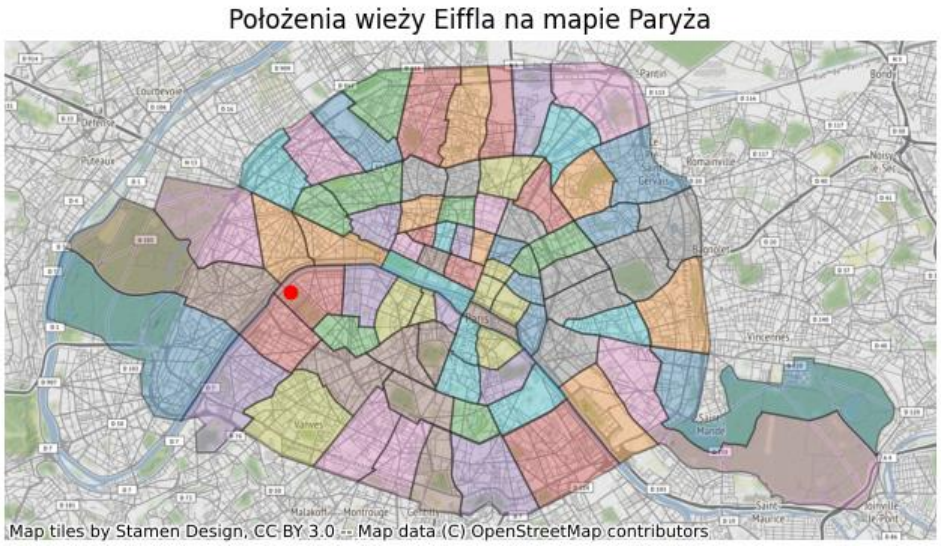
Rys. 30 Dzielnice Paryża narażone na zalanie.

Eksperyment 2

W tym eksperymencie sprawdzono zasięg zniszczeń sytuacji hipotetycznej - detonacji bomby jądrowej wycelowanej w wieżę Eiffla. Obszar zniszczeń został podzielony na kilka stref w kształcie okręgów wokół epicentrum wybuchu, różniących się między sobą zasięgiem i rodzajem zniszczenia. Opis zniszczeń dla poszczególnych stref, zaczynając od najmniejszych powierzchniowo prezentuje się następująco:

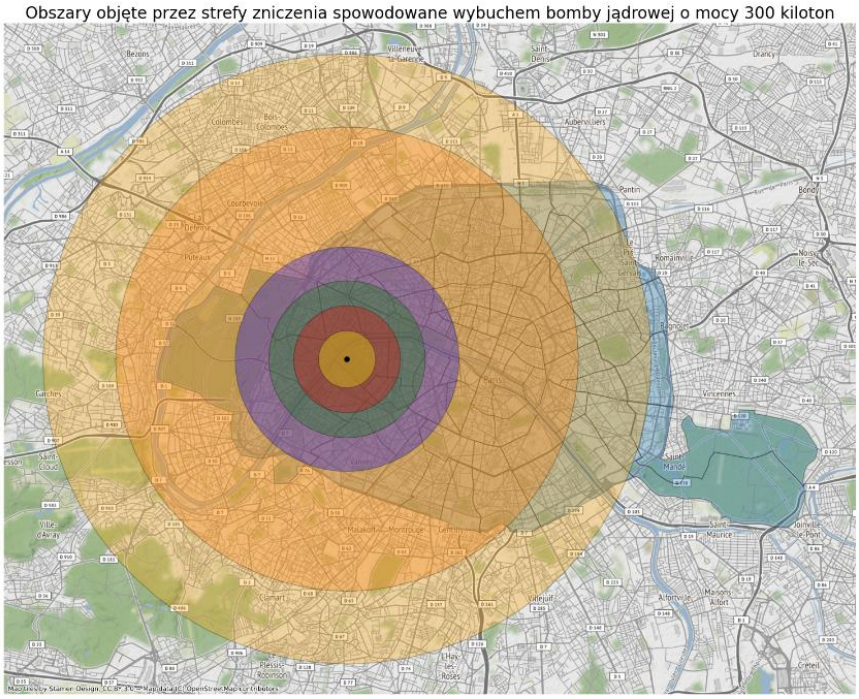
1. **Żółta strefa**: promień 0,78 km – tak duża będzie kula ognia, która dosłownie strawi wieżę Eiffla. Nikt w tej części nie będzie miał żadnych szans na przeżycie.
2. **Czerwona strefa**: promień 1,46 km – ekstremalnie silna fala uderzeniowa, niszcząca niemal całą infrastrukturę naziemną. Szanse na przeżycie zerowe.
3. **Zielona strefa**: promień 2,14 km – zasięg promieniowania o dawce 5 siwertów (śmiertelność 50% populacji).
4. **Niebieska strefa**: promień 3,06 km – fala uderzeniowa niszcząca większość budynków mieszkalnych, ale szanse na przeżycie większe od zera i zależą od schronienia (duża przeżywalność w schronach i podziemnych pomieszczeniach).
5. **Pomarańczowa strefa**: promień 6,33 km – fala ciepła, powodująca głębokie oparzenia 3 stopnia. Potencjalnie śmiertelna, ilość obrażeń zależy od miejsca pobytu i zastosowanej osłony (grube mury, zagłębienia terenu, schrony i piwnice).
6. **Jasnopomarańczowa strefa**: promień 8,32 km – fala ciepła, powodująca oparzenia 2 stopnia, w większości przypadków niezagrażające życiu.

Pierwsza część eksperymentów polegała na zdefiniowaniu geometrii wieży Eiffla – punkt o współrzędnych (648237.3 , 6862271.9) i wizualizacji jej na tle dzielnic Paryża (czerwony punkt na rys. 31)



Rys. 31 Położenie wieży Eiffla na tle dzielnic Paryża

Następnie zdefiniowany zasięg poszczególnych stref rażenia i zwizualizowano je na mapie Paryża (rys. 32)



Rys. 32 Zasięg rażenia bomby jądrowej o masie 300 kiloton

W projekcie zbadano, które z dzielnic zostaną objęte jedynie przez strefę żółtą rys. 33 (reszta stref analogicznie). Otrzymane wyniki zaprezentowano na rys. 34.

Rys. 33 Zasięg strefy żółtej na tle Paryża.

```
Strefa żółta obejmuje następujące dzielnice:
27      Gros-Caillou
28    Champs-Elysées
58          Grenelle
61            Muette
62    Porte-Dauphine
63          Chaillot
Name: district_name, dtype: object
Łączna liczba objetych tym obszarem dzielnic: 6
```

Rys. 34 Dzielnice objęte oddziaływaniem strefy żółtej.

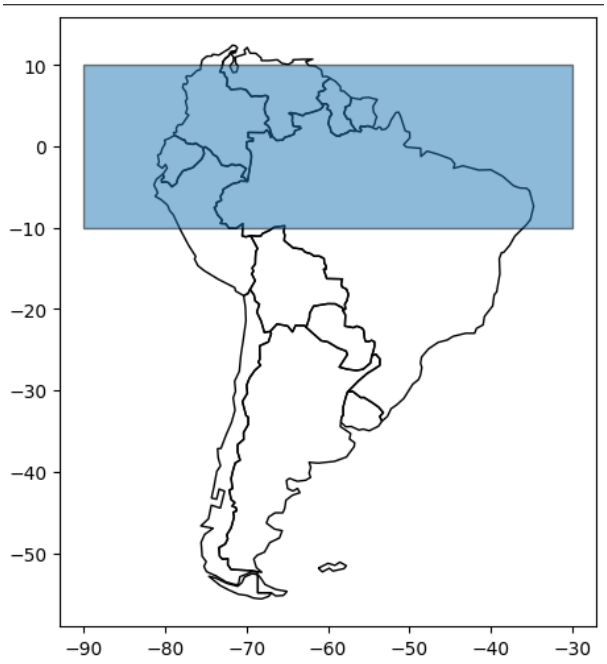
Na koniec zwizualizowano różnice między obszarem Paryża, a oddziaływaniem strefy żółtej – rys. 35.

Rys. 35 Zasięg strefy żółtej na tle Paryża

Eksperyment 3

W tej części projektu wykorzystano bufor prostokątny, do wycięcia określonego obszaru na mapie. W tym celu skorzystano z danych zawierających mapę świata, a dokładnie z geometrii Ameryki Południowej. Następnie skorzystano z funkcji intersection() służącej do obliczenia części wspólnej (przecięcia) miedzy dwoma obiektami geometrycznymi. Wyznaczono interesujący nasz fragment Ameryki Południowej – rys. 36.

Przykładowe wyniki przecięcia tych dwóch powierzchni prezentuje rys. 37.

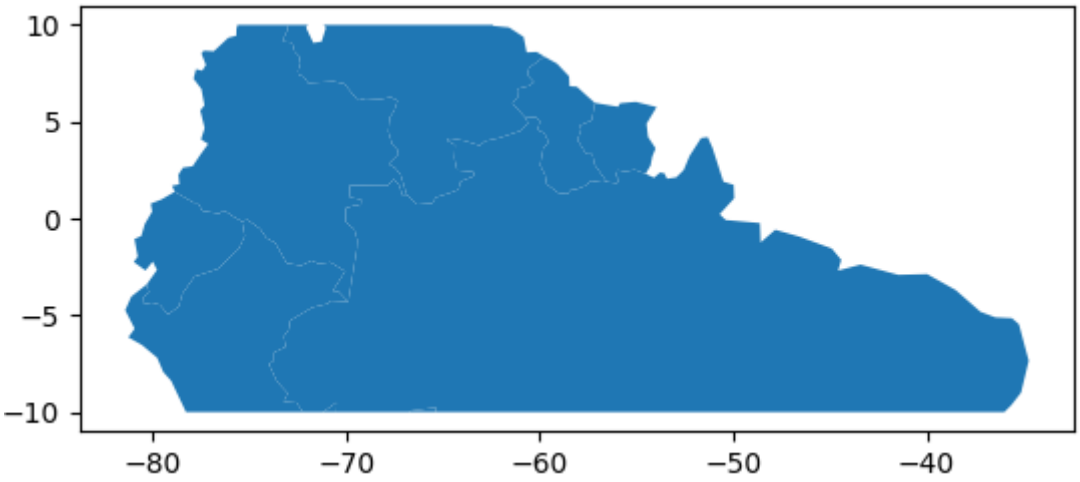


Rys.37 Ameryka Południowa i bufor na jednym wykresie.

```
                                        POLYGON EMPTY
POLYGON ((-66.64691 -9.93133, -65.33844 -9.761...
POLYGON ((-65.33844 -9.76199, -66.64691 -9.931...
                                        POLYGON EMPTY
POLYGON ((-67.06505 1.13011, -67.26000 1.72000...
```

Rys. 36 Przykładowe wyniki wywołania funkcji intersection()

Widać, że część ze zwróconych geometrii przyjmuje wartość ‘POLYGON EMPTY’. Oznacza to, że dany kraj nie przecina się z prostokątem wyznaczonym przez bufor. Takie wartości należy usunąć. W efekcie otrzymujemy następujący wynik przecięcia rys. 38.



Rys. 38 Końcowa wizualizacja wyników wywołania funkcji intersection()

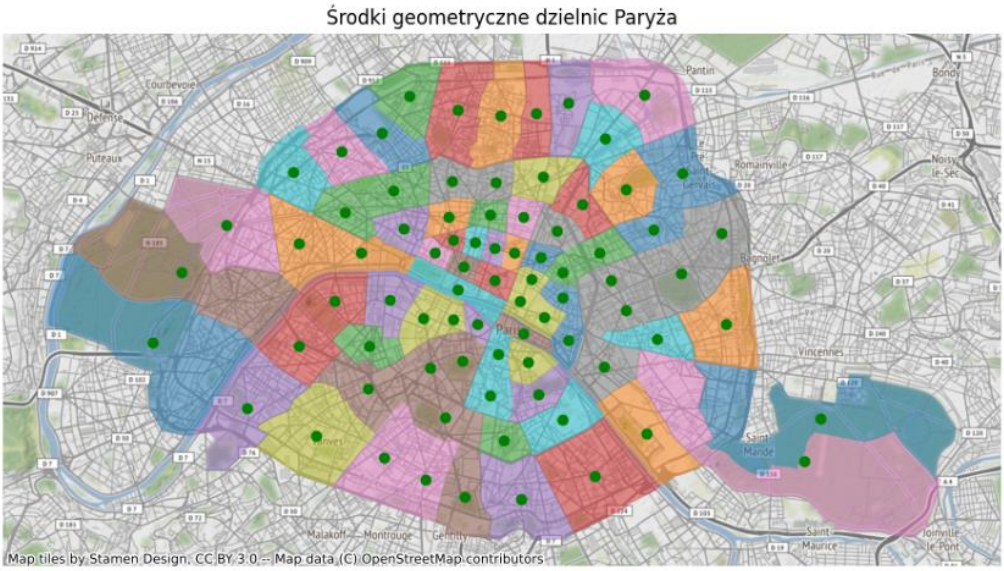
Wnioski

Przy pomocy bufora jesteśmy w stanie sprawdzić zarówno zasięg oddziaływania jakiegoś czynnika na inne, jak i wyciąć interesujący nasz obszar oraz sprawdzić, czy dane dwie geometrie się przycinają.

Wyznaczanie środków

Eksperyment 1

W eksperymencie tym wyznaczono środki geometryczne (zielone punkty na wizualizacji) dzielnic Paryża przy pomocy funkcji centroid(). Efekt wywołania tej metody został zaprezentowany na rys. 39.



Rys. 39 Centroidy dzielnic Paryża.

Z funkcji wyznaczania środków skorzystano również w ostatnim punkcie projektu podczas określania odległości wieży Eiffla od środków dzielnic Paryża.

Przycinania obiektów

Do realizacji tej części projektowej została wykorzystana funkcja intersection() z biblioteki Geopandas. Była ona zastosowana i opisana już w punkcie poprzednim, przy wyznaczaniu części wspólnej prostokątnego bufora i Ameryki Południowej. Również w rozdziale „Wyznaczanie środka i buforów" wykorzystano funkcję difference() do wycięcia z mapy Paryża obszaru jaki ulegnie zniszczeniu po wybuchu bomby jądrowej (obszar obejmujący strefę żółta).

Wykorzystane zbiory danych: „Dzielnice Paryża", „Mapa świata", „Tereny użytkowe w Paryżu"

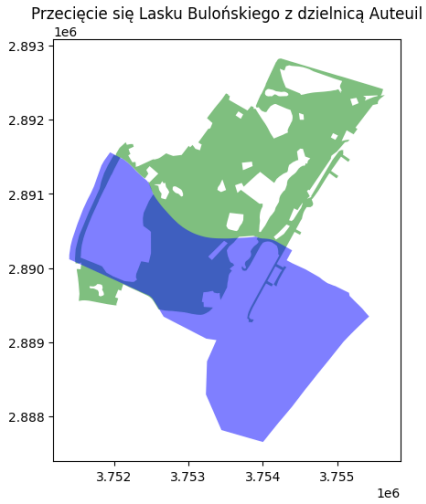
W tej części projektowej poza powyższym przykładem przecinania obiektów przeprowadzono następujące eksperymenty:

Eksperyment 1

W tym eksperymencie wykorzystano po raz pierwszy zbiór danych „Tereny użytkowe w Paryżu" w celu zbadania przecinania się powierzchni dzielnicy Auteuil z Lasem Bulońskim (rys. 40) oraz innymi terenami użytkowymi. W tym celu konieczne było wyodrębnienie dzielnicy Auteuil oraz wyznaczenie powierzchni każdego z gruntów – rys. 41. Następnie przy pomocy funkcji intersection() wyznaczono część wspólną obu terenów – rys. 42.

Railways and associated land	1.935793
Water bodies	3.189706
Sports and leisure facilities	3.578509
Discontinuous Dense Urban Fabric	3.657343
Roads and associated land	7.401574
Green urban areas	9.858438
Industrial, commercial, public	13.295042
Continuous Urban Fabric	45.943090

Rys. 41 Całkowita powierzchnia gruntów użytkowych danej klasy



Rys. 40 Wizualizacja położenia dzielnicy Auteuil względem Lasu Bulońskiego

Rys. 42 Część wspólna Lasku Bulońskiego z dzielnicą Auteuil

Stosunek powierzchni przecięcia dwóch obszarów geometrycznych (rys. 42) do powierzchni całego obszaru dzielnicy (Auteuil) wynosi około 0.23.

W dalszej części sprawdzono jaki procent całego obszaru dzielnicy Auteuil stanowi każda z klas terenów użytkowych. W tym celu również skorzystano z funkcji intersection() jednak w tym przypadku dla wszystkich klas, a nie tylko obszarów zielonych tak jak to było w poprzedniej części. Wizualizacja rozkładu terenów użytkowych w dzielnicy Auteuil zaprezentowano na rys. 43, zaś procentowy rozkład powierzchni zajmowanej przez daną klasę na rys. 44.

Rys. 43 Rozkład powierzchni zajmowanej przez daną klasę terenu w dzielnicy Auteuil

Discontinuous Dense Urban Fabric	4.639087
Water bodies	5.396698
Industrial, commercial, public	6.567202
Roads and associated land	7.615233
Continuous Urban Fabric	23.426005
Green urban areas	24.698847
Sports and leisure facilities	27.656929

Rys. 44 Procentowy rozkład powierzchni zajmowanej przez daną klasę w dzielnicy Auteuil

Wnioski

Przy pomocy funkcji służących do przycinania obiektów jesteśmy w stanie w szybki i łatwy sposób wyodrębnić interesujący nas fragment danych, który następnie może zostać poddany dalszej obróbce i przekształceniom.

W bibliotece Geopandas występują dwie bardzo podobne pod względem nazwy funkcje, mianowicie jest to metoda intersection() i intersects(). Mimo zbliżonej nazwy zwracają one różne wartości.

- Funkcja intersection() zwraca geometrię, która jest wynikiem przecięcia dwóch obiektów geometrycznych. Może to być przecięcie dwóch linii, poligonów, punktu i linii, itp. Na przykład, jeśli mamy dwa poligony, funkcja intersection() zwróci obszar, w którym poligony się przecinają.
- Funkcja intersects() jest funkcją logiczną, która sprawdza, czy dwa obiekty geometryczne mają przecięcie. Zwraca wartość logiczną: True, jeśli obiekty się przecinają, i False, jeśli nie ma przecięcia. Ta funkcja nie zwraca dokładnej geometrii przecięcia, ale tylko informację o tym, czy przecięcie istnieje.

Podsumowując, intersection() służy do uzyskania geometrycznego przecięcia dwóch obiektów, podczas gdy intersects() informuje o tym, czy dwa obiekty mają jakiekolwiek przecięcie.

Tworzenie unii geometrycznej

Do realizacji tej części projektowej została wykorzystana funkcja union() oraz unary_union() z biblioteki Geopandas, pozwalają one na wykonanie operacji unii na zbiorze obiektów geometrycznych.

Wykorzystane zbiory danych: „Mapa świata”

W tej części projektowej przeprowadzono następujące eksperymenty:

Eksperyment 1

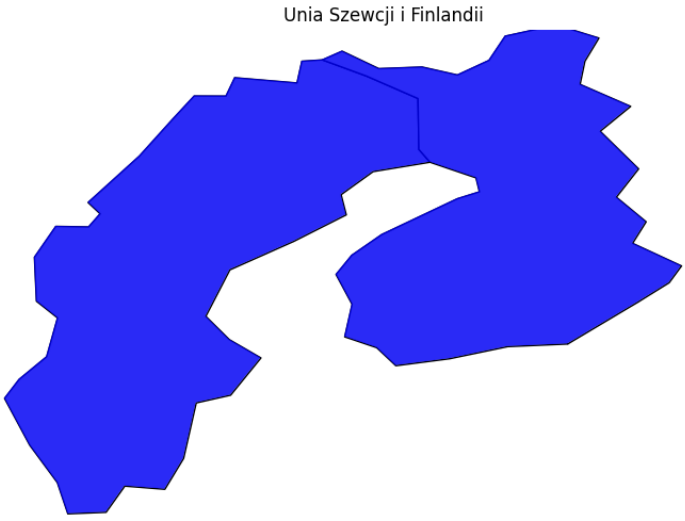
Polegał na połączeniu geometrii dwóch państw – Norwegii i Finlandii w jedną. W tym celu na początku należało wyodrębnić geometrię poszczególnych państw przy pomocy operacji unary_union – rys 45.

	iso_a3	name	continent	pop_est	gdp_md_est	geometry
151	SWE	Sweden	Europe	9960487.0	498100.0	POLYGON ((20.64559 69.10625, 21.97853 68.61685...

	iso_a3	name	continent	pop_est	gdp_md_est	geometry
52	FIN	Finland	Europe	5491218.0	224137.0	POLYGON ((23.90338 66.00693, 23.56588 66.39605...

Rys. 45 Geometria Szwecji i Finlandii

Następnie dokonano połączenia powierzchni Szwecji i Finlandii w jeden obiekt przy pomocy funkcji union(). Efekt tego połączenia jest poligon, który prezentuje rys. 46.



Rys. 46 unia Szwecji i Finlandii przy pomocy funkcji union()

Eksperyment 2

Eksperyment ten był analogiczny do poprzedniego, z tą różnicą, że dokonano unii państw Bliskiego Wschodu (Arabii Saudyjskiej, Zjednoczonych Emiratów Arabskich, Egiptu, Omanu, Iranu, Iraku, Syrii, Izraela, Jordanii, Palestyny, Kataru, Jemenu, Kuwejtu, Turcji i Libanu). Rezultat tego połączeni przedstawia rys. 47.

Rys. 47 Unia państw Bliskiego Wschodu

W efekcie otrzymano multipologon.

Wnioski:

W bibliotece Geopandas istnieją dwie funkcje, union() i unary_union(), które są używane do łączenia geometrii. Wynik końcowy mają taki sam – tworzą unię geometrii, jednak różnią się pod względem obiektów, z których tą geometrię tworzą. Oto różnica między tymi dwiema funkcjami:

- Funkcja union() służy do łączenia dwóch lub więcej obiektów geometrii w jeden obiekt geometryczny. Wynikiem jest nowa geometria, która zawiera obszar obejmujący wszystkie składowe obiekty. Na przykład, jeśli połączymy trzy poligony za pomocą funkcji union(), otrzymamy nowy poligon obejmujący cały obszar wszystkich trzech poligonów.
- Funkcja unary_union() działa na pojedynczym zbiorze geometrii i łączy je w jeden obiekt geometryczny. Wynikiem jest nowa geometria, która zawiera obszar obejmujący wszystkie składowe geometrie w danym zbiorze. Może to być przydatne, gdy mamy wiele geometrii w jednym zbiorze danych i chcemy uzyskać jedną geometrię obejmującą cały obszar tych geometrii.

Podsumowując, union() jest używane do łączenia dwóch lub więcej obiektów geometrii, tworząc nową geometrię obejmującą wszystkie te obiekty. Z kolei unary_union() działa na pojedynczym zbiorze geometrii, łącząc je w jedną geometrię obejmującą cały obszar.

Transformacje afiniczne

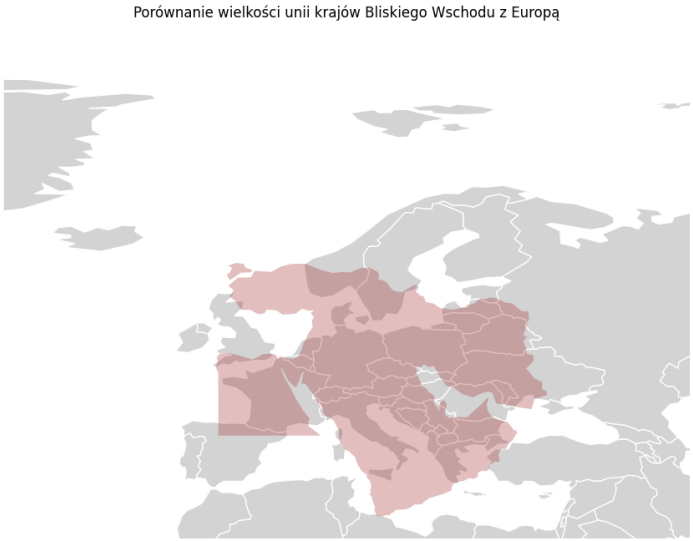
Do realizacji tej części projektowej została wykorzystana funkcja translate () – do przesuwania geometrii wzdłuż osi (x,y), scale() – do skalowania i zmiany rozmiaru obiektów geometrycznych oraz rotate() – do obrodu obiektów o zadany kąt. Wszystkie te funkcje pochodzą z biblioteki Shapely.

Wykorzystane zbiory danych: „Mapa świata”

W tej części projektowej przeprowadzono następujące eksperymenty:

Eksperyment 1

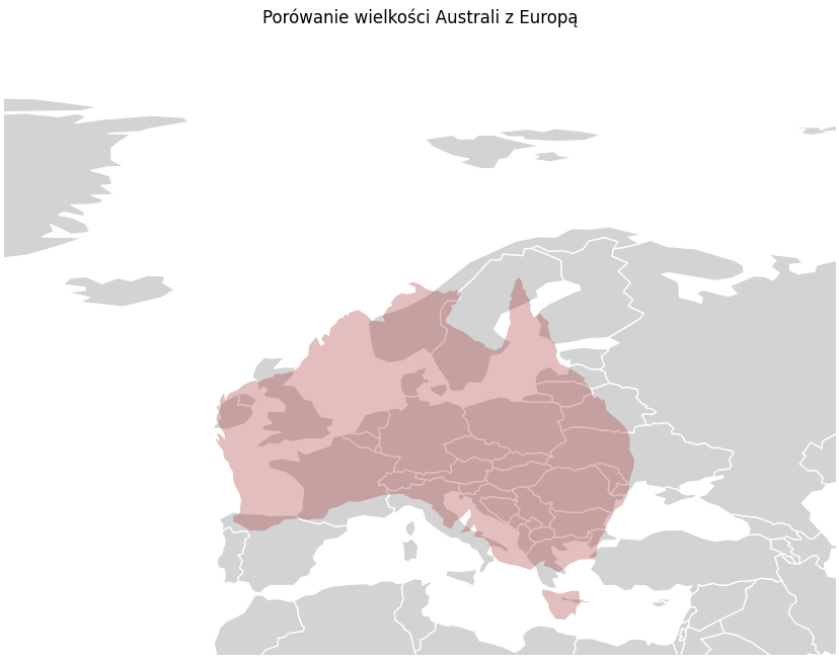
Eksperyment ten bazuje na unii państwa Bliskiego Wschodu utworzonej w poprzednim podpunkcie. Utworzona unia przesuwana jest przy pomocy funkcji translate() w taki sposób, aby porównać jej wielkość z wielkością kontynentu Europy. Uzyskany wynik przesunięcia wzdłuż osi x i y prezentuje rys. 48.



Rys. 48 Porównanie wielkości unii krajów Bliskiego wschodu z Europą

Eksperyment 2

Eksperyment ten był analogiczny do poprzedniego z tym, że translacji poddano Australię, której powierzchnie porównano z Europą. Wyniki przedstawia rys. 49.



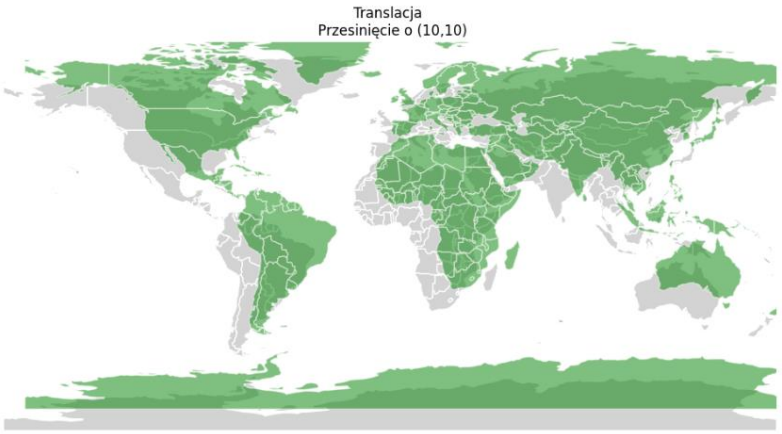
Rys. 49 Porównanie wielkości Australii z Europa

Eksperyment 3

Polegał na przedstawieniu efektu działania poszczególnych funkcji afinicznych na mapę świata.

Translacja

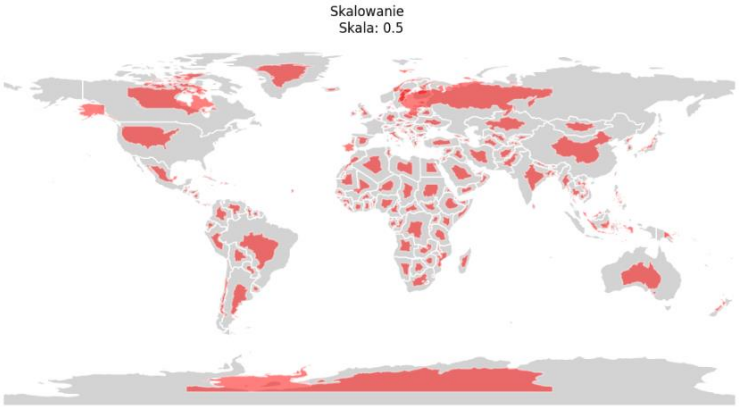
Translacji o (10,10) wszystkich państwa świata – rys. 50.



Rys. 50 Translacja

Skalowanie

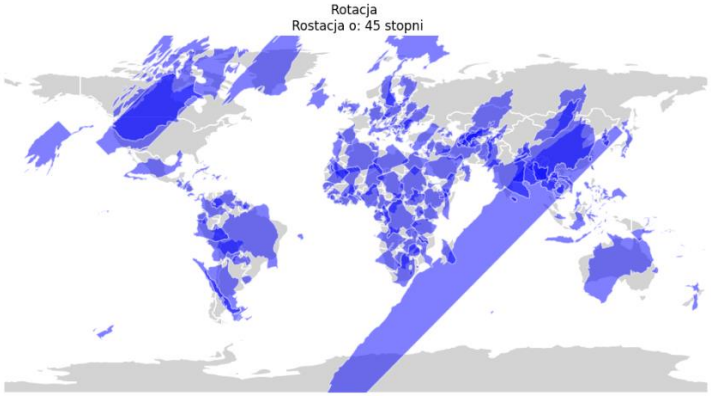
Skalowanie o połowę wszystkich państwa świata względem obu osi współrzędnych– rys. 51.



Rys. 51 Skalowanie

Rotacja

Rotacja o kąt 45° wszystkich państwa świata – rys. 52.



Rys. 52 Rotacja

Wnioski:

Operacje afiniczne, takie jak translacja, rotacja i skalowanie, są przydatne w przetwarzaniu danych geoprzestrzennych za pomocą biblioteki Geopandas. Umożliwiają one manipulację geometrią obiektów geoprzestrzennych. Translacja pozwala na przesunięcie obiektów w przestrzeni, rotacja umożliwia obrót obiektów, a skalowanie pozwala na zmianę rozmiaru. Operacje te wykorzystywane są często w wielu scenariuszach analizy przestrzennej, np. do porównywania obiektów.

Metody transformacji typów

Do realizacji tej części projektowej została wykorzystana funkcja to_wkt () – konwersja do formatu WKT, to_json() – do formatu GeoJSON, from_wkt() – wykorzystana przy transformacji z formatu WKT do GeoJSON oraz from_file() – wykorzystana przy transformacji z formatu GeoJSON do WKT.

Wykorzystane dane: punkt o współrzędnych (X,Y, Z) stworzony w przestrzeni 3D.

W tej części projektowej przeprowadzono następujące eksperymenty:

Eksperyment 1

Eksperyment ten polegał na sprawdzaniu i porównuj różnic między formatem WKT, a GeoJSON. W tym celu stworzono pojedynczy, trójwymiarowy punkt (współrzędne punktu - (1.656, 2.5, 10.123456712345171) i umieszczono go w GeoDataFrame. Następnie dokonano konwersji tego punktu z formatu WKT do GeoJSON oraz z formatu GeoJSON do WKT. Na koniec dokonano porównania różnic między oryginalnym punktem, reprezentacjami WKT i GeoJSON oraz wynikami transformacji.

Otrzymane wyniki transformacji typów, po uruchomieniu kodu, prezentują się następująco:

Różnice między formatami WKT i GeoJSON:

Różnica 1: Prezentacja danych
Oryginalny punkt (3D):
geometry
0 POINT Z (1.65600 2.50000 10.12346)

Prezentacja i precyzja zapisu liczbowego danych w formacie WKT:
0 POINT Z (1.656 2.5 10.123457)
dtype: object

Prezentacja i precyzja zapisu liczbowego danych w formacie GeoJSON:
{"type": "FeatureCollection", "features": [{"id": "0", "type": "Feature", "properties": {}, "geometry": {"type": "Point", "coordinates": [1.656, 2.5, 10.12345671234517]}, "bbox": [1.656, 2.5, 1.656, 2.5]}], "bbox": [1.656, 2.5, 1.656, 2.5]}

Różnica 2: Konwersja formatów
Konwersja formatu WKT do GeoJSON:
geometry
0 POINT Z (1.65600 2.50000 10.12346)

Konwersja formatu GeoJSON do WKT:
id geometry
0 0 POINT Z (1.65600 2.50000 10.12346)

Wnioski:

Różnica 1: Sposób prezentacji i struktura obiektu

Pierwszą widoczną różnicę między formatami danych można dostrzec już w sposobie prezentacji danych:

Oryginalny punkt (3D): POINT (1.65600 2.50000 10.12346)
Format WKT: POINT (1.656 2.5 10.123457)
Format GeoJSON: {"type": "FeatureCollection", "features": [{"id": "0", "type": "Feature", "properties": {}, "geometry": {"type": "Point", "coordinates": [1.656, 2.5, 10.12345671234517]}, "bbox": [1.656, 2.5, 1.656, 2.5]}], "bbox": [1.656, 2.5, 1.656, 2.5]}

Oba formaty przechowują informacje o geometrii, ale posiadają różne struktury danych i dodatkowe właściwości. W formacie WKT pojedyncza geometria jest zwykle przedstawiana jako pojedynczy ciąg znaków, bez dodatkowej struktury obiektowej, zaś w formacie GeoJSON geometria jest przedstawiana jako obiekt JSON, który może zawierać dodatkowe właściwości takie jak: informacja o typie geometrii (w naszym przypadku jest to punkt), ewentualnych atrybutach obiektu oraz granicach (bbox) geometrii.

Sama notacja punkty jest też różna. Punkt w formacie WKT jest reprezentowany jako "POINT (x y)" lub "POINT Z (x y z)", gdzie x, y i z to współrzędne punktu, zaś punkt w formacie GeoJSON jest reprezentowany jako obiekt JSON z właściwością "type" ustawioną na "Point" i tablicą współrzędnych, np. "coordinates": [x, y] lub "coordinates": [x, y, z].

Różnica 2: Precyzja zapisu liczbowego

Na podstawie powyższego zestawiania sposobu prezentacji i precyzji zapisu liczbowego danych w zależności od formatu, widać, że dokładność (liczba miejsc po przecinku) oraz sposób zaokrąglenia zdefiniowanych wartości geometrii punkt 3D różni się między sobą. W przypadku prezentacji oryginalnego punktu wszystkie współrzędne (x,y,z) mają jednakową liczbę cyfr po przecinku (pierwotnie każdy miał inną), a współrzędna wysokości została skrócona do 5 miejsc i zaokrąglona w górę. W przypadku formatu WKT zaokrąglona w górę została 6 pozycja, jedynie format GeoJSON zachowuje pełną wartość liczbową wysokości punktu zdefiniowaną przez użytkownika.

Różnica 3: Wielkość liter

Inną kluczową różnicą między WKT i GeoJSON jest to, że GeoJSON (oparty na JSON) uwzględnia wielkość liter, podczas gdy WKT nie.

Różnica 4: Obsługa wielu obiektów

Format WKT obsługuje pojedynczą geometrię. Jeśli chcemy reprezentować kolekcję obiektów, musisz użyć wielokrotnych ciągów WKT lub zagnieżdżonych struktur. W formacie GeoJSON obsługiwane są zarówno pojedyncze obiekty, jak i kolekcje obiektów za pomocą struktury "Feature" lub "FeatureCollection".

Różnica 5: Konwersja między formatami

Podczas konwersji między formatami - format GeoJSON obsługuje dodatkowe właściwości (np. atrybuty) obok geometrii, które nie są obsługiwane przez format WKT. Przy konwersji z GeoJSON do WKT te dodatkowe właściwości mogą zostać utracone. Dodatkowo, w związku z tym, że precyzja zapisu liczbowego w obu formatach jest różna - format GeoJSON używa standardu IEEE 754 dla zapisu liczb

zmiennoprzecinkowych, który jest bardziej dokładny od standardu stosowanego w formacie WKT może to prowadzić do nieznacznych różnic w precyzji liczb (współrzędnych obiektu), a tym samym w położeniu punktów w przestrzeni.

Sąsiedztwo, odległość i przecinanie obiektów

Wykorzystane zbiory danych: „Mapa świata", „Rzeki i jeziora na świecie", „Dzielnice Paryża", „Stacje rowerowe w Paryżu"

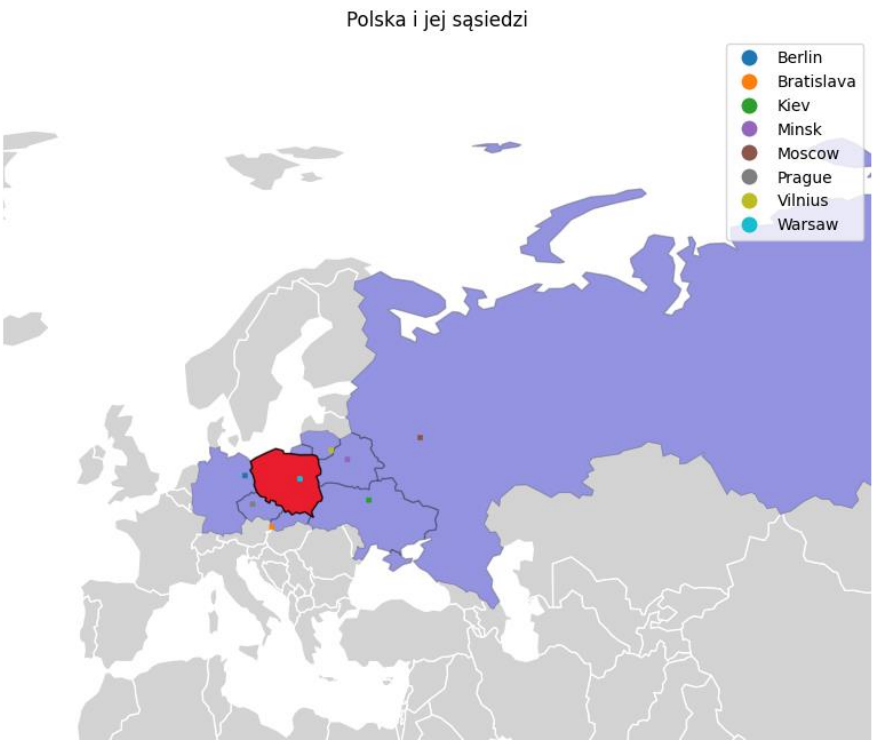
Sąsiedztwo

W tej części projektowej wyznaczono państwa sąsiadujące z Polską. W tym celu skorzystano z funkcji sjoin() z biblioteki GeoPandas, która jest używana do łączenia obiektów typu GeoDataFrame na podstawie określonego predykatu przestrzennego (w naszym przypadku były to obiekty 'GeoDataFrame' reprezentujące zbiór danych „Mapa świata" i wcześniej wyznaczone państwo, którego sąsiadów szukamy – 'target_country') i zwraca rekordy, które spełniają ten warunek.

Parametry funkcji sjonin() wykorzystane w tym wywołaniu to:

- how='inner' określa, że zostaną zwrócone tylko rekordy, które mają przecięcie przestrzenne z 'target_country'.
- predicate='intersects' określa, że jako predykat przestrzenny zostanie użyta operacja intersects, czyli przecięcie przestrzenne. Oznacza to, że zostaną zwrócone tylko rekordy, które mają przecięcie przestrzenne z 'target_country'.

W rezultacie funkcja sjoin() zwraca nowy obiekt typu GeoDataFrame, który zawiera rekordy z countries, które przecinają się z target_country. Prezentuje to rys. 53.



Rys. 53 Polska i jej sąsiedzi

W projekcie zaprezentowano również sposób wyznaczanie sąsiadów każdego z państw ze zbioru „Mapa świata". W tym celu stworzono metodę o nazwie 'find_neightbours', która iteruje po wszystkich wierszach. Do wyznaczenia sąsiedztwa wykorzystuje funkcję disjoint(), która sprawdza, czy dwa obiekty geometryczne są rozłączne, czyli czy nie mają żadnych punktów wspólnych (w naszym przypadku jest ona zaprzeczona za pomocą ~, ponieważ szukamy obiektów mających część wspólną – granicę między państwami). Przykładowe wyniki otrzymane po wywołaniu tej metody prezentuj Tabela 4.

Tabela 4 Sąsiedzi danego kraju

	name	neighbors
0	Afghanistan	[Afghanistan, China, Iran, Pakistan, Tajikista...
1	Angola	[Angola, Dem. Rep. Congo, Congo, Namibia, Zambia]
2	Albania	[Albania, Greece, Kosovo, Macedonia, Montenegro]
3	United Arab Emirates	[United Arab Emirates, Oman, Saudi Arabia]
4	Argentina	[Argentina, Bolivia, Brazil, Chile, Paraguay, ...
...	...	...
172	Vanuatu	[Vanuatu]
173	Yemen	[Oman, Saudi Arabia, Yemen]
174	South Africa	[Botswana, Lesotho, Mozambique, Namibia, Swazi...
175	Zambia	[Angola, Botswana, Dem. Rep. Congo, Mozambique...
176	Zimbabwe	[Botswana, Mozambique, South Africa, Zambia, Z...

Wnioski:

Do wyznaczenia sąsiadów danego obiektu wykorzystujemy w projekcie funkcje pozwalające na łączenie obiektów typu ‘GeoDataFrame’. Tego typu zabieg pozwala nam na stworzenie nowej tabeli zawierającej większą ilość danych, co może nam przydać podczas dalszej analizy lub obróbki tego zbioru.

Odległość

W tej części projektu wyznaczana była odległość między punktami przy pomocy funkcji distance() z biblioteki Shapely i geopy.distance z biblioteki Geopy.

Eksperyment 1

Jest to kontynuacja zadania poprzedniego, w której wyznaczana jest odległość między stolica Polski (Warszawą), a stolicami jej sąsiadów. Następnie zwracana jest informacja o dystansie dzielącym te miasta kilometrach. Otrzymane wyniki prezentuje rys. 54.

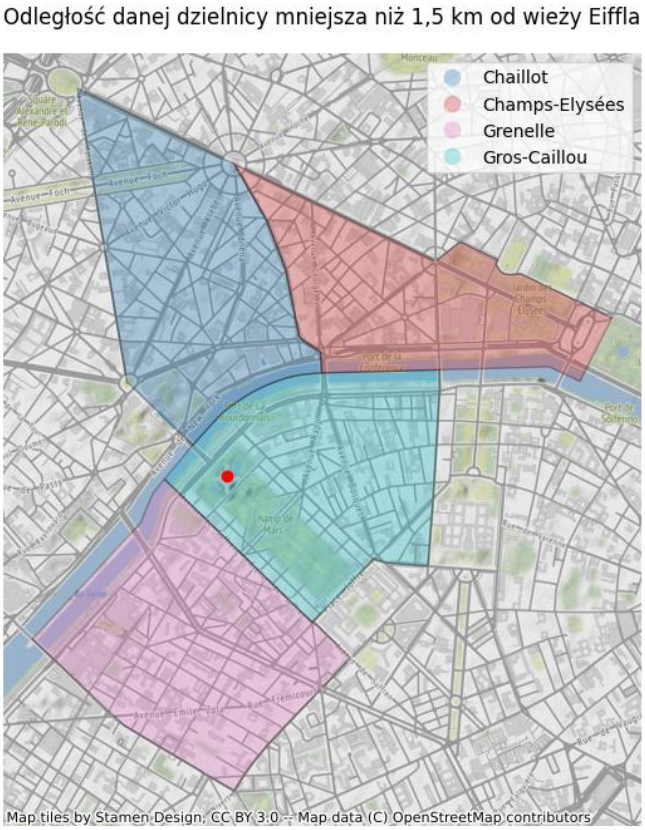
```
Dystans między Vilnius a Warszawą: 394.174221638932 kilometrów
Dystans między Minsk a Warszawą: 476.62179108544797 kilometrów
Dystans między Prague a Warszawą: 516.4121920896865 kilometrów
Dystans między Berlin a Warszawą: 517.967447337985 kilometrów
Dystans między Bratislava a Warszawą: 533.6392318173173 kilometrów
Dystans między Kiev a Warszawą: 692.6191653981521 kilometrów
Dystans między Moscow a Warszawą: 1153.7043173100435 kilometrów

Najbliższa destynacja to Vilnius (394.174221638932 km)
Najdalsza destynacja to Moscow (1153.7043173100435 km)
```

Rys. 54 Odległość z Warszawy do innych stolic sąsiadów polski

Eksperyment 2

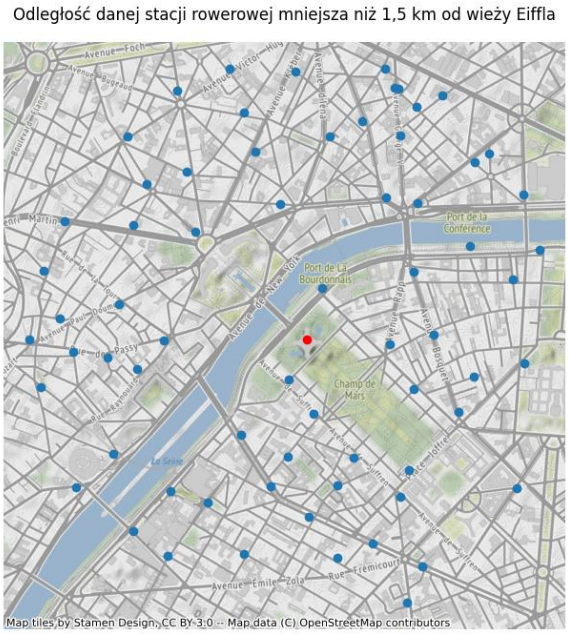
Eksperyment ten polegał na wyznaczeniu, które z dzielnic Paryża znajdują się w określonej zakresie odległości od wieży Eiffla (w projekcie jest to < 1,5 km). Jako punkt orientacyjny służący do wyznaczenia tej odległości wykorzystano środki geometryczne dzielnic (centroidy) oraz współrzędne wieży Eiffla. Wyniki prezentuje rys. 55.



Rys. 55 Dzielnice Paryża, których środki geometryczne są w odległości mniejszej niż 1,5 km od wieży Eiffla.

Eksperyment 2

Eksperyment ten był analogiczny do poprzedniego z tym, że badana była odległość stacji rowerowych od wieży Eiffla. Otrzymane wyniki prezentuje rys. 56.



Rys. 56 Stacje rowerowe, których odległość od wieży Eiffla jest mniejszej niż 1,5 km

Wnioski:

Podstawowa różnica między funkcją distance() w geopandas a funkcją distance() z geopy.distance polega na tym, że geopandas jest biblioteką do analizy przestrzennej, która operuje na obiektach geometrycznych, takich jak punkty, linie, wielokąty, itp. Funkcja distance() w geopandas oblicza odległość na podstawie geometrii tych obiektów i działa na różnych typach geometrii.

Z kolei geopy.distance jest oddzielną biblioteką, która specjalizuje się w obliczaniu odległości geograficznych między dwoma punktami na Ziemi. Funkcja distance() z geopy.distance używa różnych metryk geograficznych i operuje na punktach, określanych jako krotki współrzędnych (długość, szerokość geograficzna).

Wybór odpowiedniej funkcji do obliczania odległości zależy od kontekstu i rodzaju danych, z którymi pracujemy. Jeśli mamy dane przestrzenne w postaci obiektów geometrycznych, geopandas może być lepszym wyborem, zaś jeśli mamy do czynienia jedynie ze współrzędnymi geograficznymi punktów, to zastosowanie geopy.distance może być bardziej odpowiednie wyborem.

Przecinanie obiektów

Do realizacji tej części projektowej została wykorzystana funkcja intersects() z biblioteki GeoPandas. Była ona już zastosowana i opisana w punkcie „Wyznaczanie środków oraz buforów”, podczas sprawdzania przez jakie dzielnice przepływa Sekwana oraz z jakim dystryktami Paryża przecina się dana strefa zniszczenia spowodowanego wybuchem bomby jądrowej.

W tej części projektowej poza powyższym przykładem przecinania obiektów przeprowadzono następujące eksperymenty:

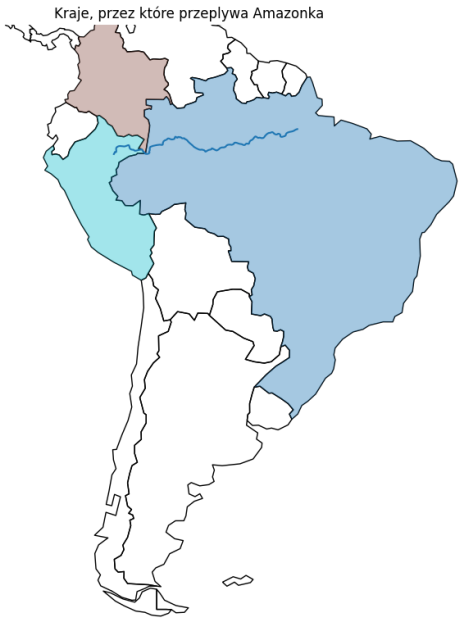
Eksperyment 1

Polegał na sprawdzeniu przez jakie kraje przepływa rzeka Amazonka. W przeciwieństwie do dwóch poprzednich przykładów w tym eksperymencie przecięcie miedzy obiektami zostanie wyznaczone za pomocą funkcji crosses() z biblioteki GeoPandas, która pozwala na sprawdzenie, czy dana geometria przecina inną geometrię.

W tym celu konieczne było zdefiniowanie geometrii Amazonki, a następnie sprawdzanie przez jakie kraje przepływa za pomocą funkcji crosses(). Efekt wywołania tej funkcji prezentuje rys. 56 i 57.

	iso_a3	name	continent	pop_est	gdp_md_est	geometry
22	BRA	Brazil	South America	207353391.0	3081000.0	POLYGON ((-57.62513 -30.21629, -56.29090 -28.8...
35	COL	Colombia	South America	47698524.0	688000.0	POLYGON ((-66.87633 1.25336, -67.06505 1.13011...
124	PER	Peru	South America	31036656.0	410400.0	POLYGON ((-69.52968 -10.95173, -68.66508 -12.5...

Rys. 56 Tabela zwierająca kraje, przez które przepływa Amazonka.



Rys. 57 Wizualizacja rozwiązania wywołania funkcji crosses().

9. Wnioski ogólne

W środowisku Python istnieje wiele bibliotek (tj. GeoPandas, Shapely) pozwalających na łatwe i szybkie wykonywanie różnego rodzaju przekształceń na danych przestrzennych. Każda z nich posiada bogatą dokumentację zawiera szczegółowe informacje na temat dostępnych funkcji, ich argumentów, przykładów użycia i innych przydatnych informacji.

Ważnym aspektem przed przystąpieniem do wykonywania przekształceń na danych przestrzennych, jest to, aby przeanalizować dane wejściowe - ich format i strukturę. Bardzo istotne jest, aby upewnić się, że dane są poprawnie wczytane do odpowiednich struktur danych i posiadają formaty zgodne i

obsługiwane przez konkretne biblioteki (np. w przypadku bibliotek GeoPandas i Shapely dane powinny być reprezentowane jako obiekty GeoDataFrame lub Geometry). Pozwala to unikać niepotrzebnych problemów.

Podczas analizy danych warto zbadać ich właściwości, takie jak układ współrzędnych, jednostki miary, dokładność geometrii itp. Niektóre funkcje przekształceń mogą wymagać ujednolicenia niektórych właściwości danych, na przykład wymagają jednakowego układu współrzędnych.

Podczas pracy z danymi przestrzennymi często mamy do czynienia z dużymi zbiorami danych, co może wpływać na wydłużenie czasu i zwiększenie ilości potrzebnych zasobów obliczeniowych do wykonywania różnego rodzaju przekształceń. W związku z tym ważna jest, aby zoptymalizować wydajność obliczeniową poprzez, np. wykonywanie operacji na partycjach danych.

Kolejnym istotnym aspektem podczas pracy z danymi wszelkiego rodzaju jest walidacja wyników w celu ich weryfikacji i sprawdzenia, czy są możliwe (posiadają realne wartości) lub zgodne z oczekiwanymi . Pomóc w tym może wizualizacja otrzymanych wyników lub porównanie ich z innymi zbliżonymi danymi, aby upewnić się, że przekształcenia zostały wykonane poprawnie. Podczas tego typu testów ważne jest, aby zachować kontekst geoprzestrzenny, czyli należy zwrócić uwagę na odpowiednie jednostki miary, skalowanie, odwzorowanie i inne aspekty związane z analizą danych przestrzennych.

Bardzo ważnym aspektem przy pracy z danymi przestrzennymi jest dobór odpowiedniego układu odniesienia przestrzennego danych wejściowych - CRS (ang. Coordinate Reference System). W zależności od jego rodzaju otrzymane wyniki dla tych samych danych mogą się różnić między sobą. Przykładowo, jeśli dane wejściowe używają układu odniesienia EPSG:4326, to wywołanie funkcji ‘area’ na obiekcie zwróci powierzchnię wyrażoną w stopniach kwadratowych, zaś dla EPSG:3857, powierzchnia będzie wyrażona w metrach kwadratowych.

10. Bibliografia

1. Zbiór danych „Mapa Świata” - https://www.naturalearthdata.com/downloads/110m-cultural-vectors/110m-admin-0-countries/
2. Zbiór danych „Miasta Świata” - https://www.naturalearthdata.com/downloads/110m-cultural-vectors/110m-populated-places/
3. Zbiór danych „Rzeki i jeziora na świecie” - https://www.naturalearthdata.com/downloads/50m-physical-vectors/50m-rivers-lake-centerlines/
4. Zbiór danych „Dzielnice Paryża” - https://github.com/blackmad/neighborhoods/blob/master/paris.geojson
5. Zbiór danych „Stacje rowerowe w Paryżu” – https://opendata.paris.fr/pages/home/
6. Zbiór danych „Tereny użytkowe w Paryżu” - https://land.copernicus.eu/local/urban-atlas
7. Geometria rzeki Sekwany - http://geojson.io/#map=2/0/20
8. Dokumentacja GeoPandas - https://geopandas.org/en/stable/index.html
9. Dokumentacja Shapely - https://shapely.readthedocs.io/en/stable/manual.html#shapely.affinity.affine_transform
10. Powierzchniowa detonacja ładunku jądrowego o mocy 300 kiloton w centrum Warszawy - https://urbanrat.pl/symulacja-bomby-atomowej/
11. Materiały z zajęć z przedmiotu Przestrzenne Bazy Danych.