## Analiza danych w naukach o Ziemi – Geoinformatyka II st. – prowadzący Jakub Staszel

# Projekt 2 – Location Intelligence

Spis treści

[Analiza danych w naukach o Ziemi – Geoinformatyka II st. – prowadzący Jakub Staszel 1](#_Toc938944925)

[Projekt 2 – Location Intelligence 1](#_Toc1162041187)

[Wstęp 1](#_Toc1750047089)

[Wymagania i założenia 1](#_Toc1317024932)

[Cele projektu 2](#_Toc1763248392)

[Kolejne etapy projektu i minimalne wymagania 2](#_Toc786405196)

[Stworzenie siatek H3 dla Krakowa i Amsterdamu 2](#_Toc403055405)

[Obliczenie długości ścieżek rowerowych wewnątrz każdego heksagonu 2](#_Toc2061790095)

[Dodanie kolejnych argumentów 3](#_Toc1045485005)

[Trenowanie modeli 3](#_Toc1428004661)

[Wykorzystanie najlepszego modelu na siatce dla Krakowa 4](#_Toc615430400)

## Wstęp

Location Intelligence (LI) to dziedzina analityki, która łączy dane przestrzenne (lokalizacyjne) z danymi biznesowymi, aby pomóc organizacjom w podejmowaniu decyzji opartych na kontekście geograficznym. Wykorzystuje technologie GIS (Geographic Information Systems), wizualizację danych i zaawansowane metody analizy przestrzennej, aby identyfikować wzorce i zależności widoczne tylko przez pryzmat lokalizacji. Dzięki LI firmy mogą optymalizować procesy, planować rozwój, zarządzać ryzykiem i lepiej docierać do klientów, opierając swoje działania na dokładnej znajomości warunków terenowych, dystrybucji zasobów czy zachowań konsumentów w różnych regionach.

Rozwój infrastruktury rowerowej w miastach stał się kluczowym elementem planowania miejskiego, sprzyjającym zrównoważonemu rozwojowi i poprawie jakości życia mieszkańców. W analizie tej, zbadamy długość ścieżek rowerowych w Amsterdamie, wykorzystując do tego celu podejście oparte na heksagonalnej siatce H3. Siatka ta pozwoli na precyzyjne i jednolite porównanie różnych obszarów miasta, uwzględniając dodatkowe zmienne takie jak np. gęstość zaludnienia, ilość punktów usługowych, które mogą wpływać na potrzeby oraz intensywność użytkowania infrastruktury rowerowej. Dzięki zastosowaniu technik modelowania przestrzennego, możliwe będzie nie tylko zrozumienie obecnych potrzeb, ale także przewidywanie przyszłych wymagań i planowanie odpowiednich inwestycji infrastrukturalnych.

## Wymagania i założenia

1. Akademickie
   1. Projekt można wykonać samemu lub w grupach maksymalnie 3-osobowych.
   2. Składy zespołów oraz linki do repozytoriów należy wysłać w 1 tygodniu do prowadzącego.
   3. Projekt przewidziany jest na 4 lub 5 zajęć, jednak możliwe są modyfikacje – proszę o informację, jeśli potrzebne będzie więcej lub mniej czasu. Chciałbym respektować nakład pracy studenta określony w sylabusie, stąd też zachęcam do kontaktu, jeśli będzie to potrzebne.
   4. Weryfikacja postępów w projekcie odbywa się w cyklach tygodniowych, w każdym tygodniu oczekiwany jest postęp w budowaniu analizy i repozytorium (podlegać to będzie weryfikacji).
   5. Brak postępów czy opublikowanie całości kodu przed samym oddaniem projektu będzie wpływać negatywnie na ocenę.
   6. Oprócz repozytorium na koniec należy oddać sprawozdanie opisujące wszystkie etapy przeprowadzonej analizy. Nie ma konieczności produkowania zbędnego tekstu, liczą się opis każdego elementu analizy oraz opis wyników.
2. Techniczne
   1. Organizacja samego repozytorium ma być logiczna, ale występuje tutaj dowolność.
   2. Wykorzystywanie Jupyter Notebooks jest wskazane, szczególnie na etapie developmentu.
   3. W pełni należy korzystać z możliwości rozproszonego systemu kontroli wersji (GIT). Proszę oznaczać mnie jako reviewera w swoich PRach do głównej gałęzi, to tutaj będzie prowadzona weryfikacja postępów oraz jakości kodu. Dodatkowo, notatniki, pliki z danymi oraz wynikami nie powinny być przechowywane w repozytorium, to jest miejsce na sam kod – pomocne tutaj będzie wykorzystanie .gitignore.
   4. Analizy mają być w pełni reprodukowalne. Oznacza to, że osoba, która wchodzi do repozytorium ma możliwość przeprowadzenia wszystkich analiz podążając za instrukcjami w pliku README.
   5. W pliku README mają się znaleźć informacje dot. projektu. Do tworzenia i zarządzania środowiskiem polecam korzystać z conda-lock.
   6. Każdy kolejny etap ma sprowadzać się do pojedynczych funkcji, których parametry oraz działanie jest opisane w komentarzach.
   7. Eksperymenty należy przeprowadzać w środowisku MLFlow, konieczne jest dokumentowanie wyników dla różnych modeli w zależności od zastosowanych hiperparametrów.
3. Tematyczne
   1. Wielkość heksów można dostosować samodzielnie.
   2. Pochodzenie i wybór pozostałych argumentów jest dowolny – same dane pochodzą z OSM, w podobny sposób uzyskać można sporo innych danych (np. ilość punktów usługowych, budynków mieszkalnych, odległość do najbliższego terenu zielonego czy powierzchnia terenów zielonych w danym heksie),
   3. W projekcie należy używać środowiska MLFlow.
   4. W projekcie można przetestować efektywność m.in. modeli regresyjnych, drzew decyzyjnych, lasów losowych, GBM czy sieci neuronowych.
   5. Przed przystąpieniem do tworzenia modeli, należy podsumować wykorzystywany zestaw danych (chociażby poprzez opis atrybutów i przedstawienie korelacji pomiędzy atrybutami).

## Cele projektu

Głównym zadaniem tego projektu jest stworzenie modelu, który w najlepszym stopniu przewidywać będzie długość ścieżek rowerowych w danym heksie na podstawie pozostałych argumentów dla miasta Amsterdam.

Ostatnim elementem będzie przeniesienie modelu na dane dla Krakowa (stąd też należy mieć na uwadze dostępność wykorzystanych danych dla obu miast) i porównanie ich z faktyczną długością ścieżek w Krakowie.

## Kolejne etapy projektu i minimalne wymagania

To jest sugestia dot. kolejności wykonywania działań, nie jest obowiązkowa.

**UWAGA**: Jeśli problem ten będzie nierozwiązalny regresyjnie (modele będą mieć słabe wyniki) to:

1. Proszę się nie przejmować, to też jest wynik,
2. Można się zastanowić nad przejściem z problemu regresyjnego na klasyfikacyjny.

### Stworzenie siatek H3 dla Krakowa i Amsterdamu

Siatki H3 to system geoprzestrzennego indeksowania opracowany przez Uber, który dzieli powierzchnię Ziemi na heksagonalne komórki o różnych rozmiarach. System ten jest hierarchiczny, co oznacza, że każda komórka może być podzielona na mniejsze komórki, tworząc wielopoziomową strukturę. Heksagony są preferowanym kształtem, ponieważ zapewniają większą jednorodność odległości i pokrycia w porównaniu do tradycyjnych siatek kwadratowych.

W Pythonie wygenerować takie siatki można wykorzystując bibliotekę h3.

### Obliczenie długości ścieżek rowerowych wewnątrz każdego heksagonu

Ten etap pozwoli na stworzenie podstawy ostatecznego zestawu danych. Jeśli pominiemy geometrie, to mamy tak na prawdę zwykłe dane tabelaryczne.

Na tym etapie przyda się biblioteka geopandas.

### Dodanie kolejnych argumentów

Mogą to być zliczenia, pomiary czy inne dane dla kolejnych heksów. Źródłem mogą być inne dane z OSM, ale także cokolwiek innego. **Minimalna ilość dodatkowych argumentów to 5.**

Przykładowe propozycje argumentów:

* Odległość od centrum miasta,
* Powierzchnia terenów zielonych wewnątrz heksa,
* Ilość budynków mieszkalnych czy punktów usługowych,
* Gęstość zaludnienia,
* Długość dróg o różnej klasyfikacji.

### Trenowanie modeli

Można wybrać jeden typ, ale zachęcam do przetestowania większej ilości. Proszę o korzystanie z MLFlow do tworzenia eksperymentów i zapisanie wyników.

#### Podział danych

Proszę pamiętać o podzieleniu danych na zestawy: treningowy, walidacyjny i testowy. Zestaw treningowy służy do budowy modelu, walidacyjny do dostosowania hiperparametrów i unikania przeuczenia, a testowy do ostatecznej weryfikacji modelu.

#### Trenowanie modeli

Często w tym etapie stosuje się techniki takie jak walidacja krzyżowa, aby lepiej ocenić stabilność i ogólną skuteczność modelu.

Jeśli chodzi o typy modeli, które można spróbować zastosować:

* Regresja liniowa (sklearn – LinearRegression),
* Regresor wektorów nośnych (SVR) (sklearn – SVR),
* Las losowy (sklearn – RandomForestRegressor),
* Gradient Boosting Machines (GBM) (sklearn – GradientBoostingRegressor)
* Sieci neuronowe (tensorflow, keras)

#### Wybranie metryki do oceny modeli

Bez różnicy na ilość modeli, na których trenowanie się zdecydujemy, musimy wybrać metrykę, na podstawie której będziemy je oceniać (w celu optymalizacji i wyboru najlepszego z nich). W zależności od poruszanego problemu dostępne są różne metryki, dla regresji najcześćiej korzysta się z:

* Mean Absolute Error (MAE) - Średnia wartość bezwzględna błędów między przewidzianymi a prawdziwymi wartościami.
* Mean Squared Error (MSE) - Średnia kwadratów błędów. Kładzie większy nacisk na większe błędy niż MAE.
* Root Mean Squared Error (RMSE) - Pierwiastek kwadratowy z MSE. Jest w tej samej skali co zmienne zależne i często używany do oceny dokładności modeli regresyjnych.
* R-squared (R²) - Określa, jaka część zmienności zmiennej zależnej jest wyjaśniana przez model. Wartość 1 oznacza doskonałe dopasowanie modelu.

#### Optymalizacja hiperparametrów

W środowisku MLFlow możemy zdefiniować przestrzeń hiperparametrów ręcznie lub za pomocą metod zautomatyzowanych, tutaj możemy zastosować m.in.:

* Grid Search - Metoda przeszukiwania siatki (Grid Search) testuje wszystkie możliwe kombinacje hiperparametrów zdefiniowanych przez użytkownika w przestrzeni parametrów. Jest to metoda siłowa, która gwarantuje znalezienie optymalnego zestawu hiperparametrów, ale może być bardzo czasochłonna, szczególnie przy dużych przestrzeniach parametrów.

W scikit-learn można użyć klasy GridSearchCV, która wykonuje walidację krzyżową dla każdej kombinacji hiperparametrów.

* Random Search - W przeciwieństwie do Grid Search, Random Search losowo wybiera kombinacje hiperparametrów z przestrzeni parametrów na określoną liczbę iteracji. Ta metoda jest mniej kosztowna obliczeniowo i często znajduje wystarczająco dobre rozwiązania znacznie szybciej niż Grid Search.

RandomizedSearchCV w scikit-learn oferuje prostą implementację tej metody z walidacją krzyżową dla każdej losowo wybranej kombinacji.

* Bayesian Optimization - Optymalizacja bayesowska używa statystyki bayesowskiej do modelowania przestrzeni hiperparametrów i przewidywania obszarów, które mogą prowadzić do lepszych wyników. Zamiast testować hiperparametry losowo, metoda ta stara się uczyć z poprzednich prób, aby skuteczniej eksplorować przestrzeń parametrów.

Biblioteki takie jak Hyperopt i BayesianOptimization oferują implementacje tej techniki.

* Gradient-based Optimization - Niektóre nowoczesne techniki, takie jak te używane w TensorFlow (np. TFMOT) lub PyTorch, pozwalają na stosowanie optymalizacji opartej na gradientach dla hiperparametrów modelu, zwłaszcza gdy parametry te są różniczkowalne względem funkcji celu.
* Evolutionary Algorithms - Algorytmy ewolucyjne, takie jak algorytmy genetyczne, stosują metody inspirowane naturą do eksploracji przestrzeni hiperparametrów. Tworzą populacje rozwiązań, które ewoluują z czasem, przekazując "geny" najlepszych rozwiązań do kolejnych pokoleń.

Biblioteki takie jak DEAP implementują te algorytmy.

* Optuna - nowoczesne narzędzie do automatycznej optymalizacji hiperparametrów, które integruje różne strategie, takie jak przeszukiwanie oparte na drzewach. Jest bardzo elastyczne i łatwe w użyciu, a także pozwala na definiowanie bardziej złożonych przestrzeni hiperparametrów.

Użytkownik definiuje funkcję celu, która trenuje model i zwraca wartość metryki, którą Optuna próbuje zoptymalizować.

### Wykorzystanie najlepszego modelu na siatce dla Krakowa

Otrzymany model należy zastosować na siatce h3 dla Krakowa, a następnie otrzymane wyniki porównać z faktyczną długością ścieżek rowerowych w Krakowie.