Generator Optymalnego Magazynu

Dokumentacja
MISIE2

Aleksander Malinowski
Politechnika Warszawska

Szymon Matuszewski
Politechnika Warszawska

Michał Mazuryk
Politechnika Warszawska

16 kwietnia 2023

1 Idea

W ramach projektu, głównym celem jest utworzenie `modelu sieci Kohonena` w celu optymalizacji układu produktów w magazynie na podstawie `dwóch interpolowanych tabel`, które zostały przez nas utworzone na podstawie zbioru danych Order and Collect Factory Storage. W oparciu o te dane, opracowaliśmy algorytm, który pomoże w odpowiednim rozmieszczeniu produktów w magazynie, tak aby proces gromadzenia i zamawiania produktów był bardziej `efektywny`.

Sieć Kohonena, znana również jako mapa samoorganizująca, jest jednym z popularnych algorytmów uczenia nienadzorowanego, który jest używany do grupowania i wizualizacji danych wielowymiarowych. W naszym przypadku, chcemy użyliśmy sieci Kohonena do zgrupowania produktów w magazynie na podstawie utworzonych przez nas współczynników występowania produktów w jednym zamówieniu oraz współczynników dostarczanych jednego dnia w jednym kwartale. Opracowany model uczy się optymalnego układu produktów na podstawie dostępnych danych dla danego kwartału i przewiduje, jakie produkty powinny być umieszczone obok siebie w magazynie (na jednych regałach).

Jednym z głównych wyzwań w tym projekcie jest interpolacja danych z dwóch tabel, które stworzyliśmy. Pozwala to zleceniodawcy na kalibrację opisu biznesowego, o którym więcej we ww. sekcji.

Po utworzeniu i nauczeniu modelu, przeprowadziliśmy eksperymenty, które pozwoliły nam ocenić efektywność optymalnego układu produktów w magazynie. Porównaliśmy wyniki z optymalnym układem, opracowanym na podstawie modelu sieci Kohonena, z układem przypadkowo rozłożonych produktów. Ocena jest oparta na podstawie heurystycznego algorytmu optymalizacyjnego używany w problemach komiwojażera, który również zaimplementowaliśmy.

Wyniki tego projektu mogą znacznie przyczynić się do optymalizacji procesów gromadzenia i zamawiania produktów w magazynach fabryk, co może prowadzić do oszczędności czasu, pracy ludzkiej i zasobów, a także poprawy ogólnej wydajności i efektywności fabryki.

2 Kroki Rozwiązania

2.1 EDA

Etapy Wstępnej Analizy Danych:

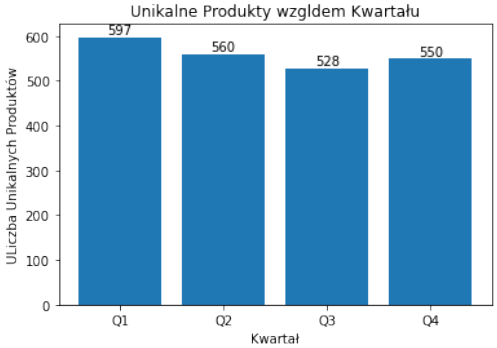
1. Wyświetlenie podstawowych informacji na temat zbioru danych np. podstawowe typy danych.
2. Analiza kwartałów w roku ze względu na zamawiane produkty
3. Dzienna i ogólna analiza statystyk poszczególnych produktów 1

Więcej tabel oraz informacji znajduje się w pliku `EDA.ipnyb`.
Wnioski ze Wstępnej Analizy Danych:

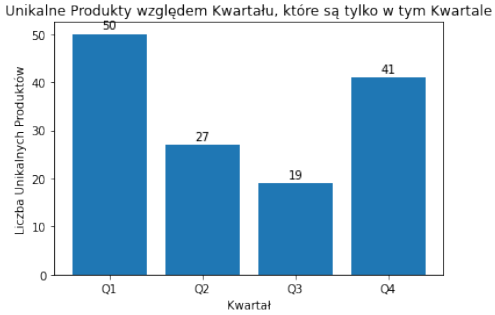
1. Należy zmienić typ zmiennej collect z object na datetime.

```
Data columns (total 7 columns):
 #   Column   Non-Null Count    Dtype
---  ------   --------------    -----
 0   order    1272973 non-null  int64
 1   costum   1272969 non-null  float64
 2   collect  1272973 non-null  object
 3   prod     1272973 non-null  object
 4   barcode  1245734 non-null  object
 5   weight   1272973 non-null  float64
 6   pacs     1272973 non-null  int64
dtypes: float64(2), int64(2), object(3)
```

Rysunek 1: Podstawowe informacje na temat typów zmiennych.



Rysunek 2: Unikalne Produkty wzgldem Kwartału.



Rysunek 3: Unikalne Produkty względem Kwartału, które są tylko w tym Kwartale.

2. Rozdzielenie zadania na 4 ramki danych odpowiadające 4 kwartałom w roku ma duże uzasadnienie: występuje sporo produktów jedynie w poszczególnym kwartale, reorganizacja magazynu co 4 miesiące pozwoli na przechowywanie jedynie potrzebnych produktów, a pozostałe mogą w tym czasie być określone jako 'niedostępne'. Ponadto, podział modelu na 4 podmodele tym bardziej jest uzasadniony zmiennością pór roku. Podobne podejścia stosuje sie m.in. w bankowości przy podsumowaniach kwartalnych (którymi się inspirowaliśmy).
3. Zmienne, które wybieramy do modelowania, czyli `order`, `collect`, `prod` nie zawierają braków danych.
4. Zmienne, które wybieramy do modelowania są łatwo oraz tanio dostępne dla firm, które chciałyby skorzystać z podobnego rozwiązania w przyszłości.

2.2 Ekstrakcja Cech

Utworzyliśmy dwie tabele ze współczynnikami dla każdego kwartału:

- scaled_matrix - tabela, gdzie wiersze i kolumny to produkty a wartość to współczynnik wspólnego występowania dwóch produktów wyliczany według następującej formuły:

$$wsp_{A,B} = N_{A,B}/N_B \tag{1}$$

, gdzie:

  - A - produkt A
  - B - produkt B
  - $N_{A,B}$ - liczba wystąpień wspólnych zamówień, gdzie razem były produkty A i B
  - N_B - liczba zamówień, gdzie zamówiony był produkt B

- scaled_matrix_date - tabela, gdzie wiersze i kolumny to produkty a wartość to współczynnik występowania dwóch produktów w zamówieniach tego samego dnia wyliczany według następującej formuły:

$$wspd_{A,B} = Nd_{A,B}/Nd_B \tag{2}$$

, gdzie:

	112082CGVPS3	331I	6001	4005-7
112082CGVPS3	1.000000	0.0125	0.002882	0.037975
331I	0.004255	1.0000	0.008646	0.082278
6001	0.004255	0.0375	1.000000	0.044304
4005-7	0.025532	0.1625	0.020173	1.000000
11162CGVM3	0.000000	0.0125	0.000000	0.012658

Rysunek 4: Pierwsze 4 wiersze i pierwsze 4 kolumny tabeli scaled_matrix.

	112082CGVPS3	331I	6001	4005-7
112082CGVPS3	1.000000	0.2000	0.051873	0.113924
331I	0.068085	1.0000	0.046110	0.145570
6001	0.076596	0.2000	1.000000	0.113924
4005-7	0.076596	0.2875	0.051873	1.000000
11162CGVM3	0.004255	0.0250	0.005764	0.012658

Rysunek 5: Pierwsze 4 wiersze i pierwsze 4 kolumny tabeli scaled_matrix_date.

- A - produkt A
- B - produkt B
- $Nd_{A,B}$ - liczba dni, w których zamawiane były produkty A i B
- Nd_B - liczba dni, gdzie zamówiony był produkt B

Współczynnik $wsp_{A,B}$ interpretujemy, jako szansę, że produkty A i B będą zamówione razem. Współczynnik $wspd_{A,B}$ interpretujemy, jako szansę, że produkty A i B będą zamówione tego samego dnia.

2.3 Interpolacja

W naszym modelu dla kwartału wprowadziliśmy możliwość interpolacji dla zleceniodawcy. Tworzymy ramkę danych `df_merged` za pomocą wzoru:

$$df_merged = \lambda * scaled\_matrix + (1 - \lambda) * scaled\_matrix\_date \tag{3}$$

, gdzie λ to parametr interpolacyjny.

Dla przykladu: jeśli zleceniodawca dysponuje lźejszym asortymentem, wtedy moźliwe jest kompletowanie większej ilości zamówień jednocześnie -> wtedy *lambda* powinna być niższa, ponieważ związek między produktami powinien opierać się na realizacji dużej ilości zamówień w ciągu dnia. Z drugiej strony, jeśli mamy doczynienia z cięższym asortymentem λ powinna być wyższa, bo chcemy jak najszybciej skompletować konkretne zamówienie.

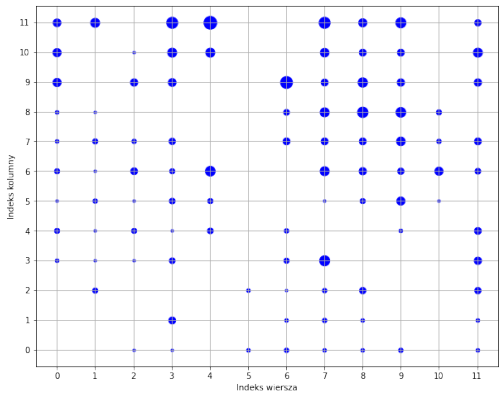
Dla naszego modelu przyjęliśmy $\lambda = 0.9$.

2.4 Modelowanie

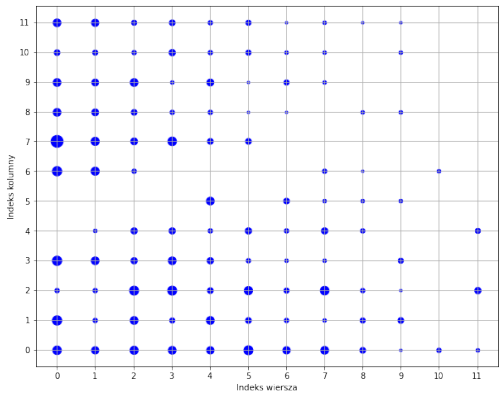
Dla każdej zintepolowanej ramki danych kwartału użyliśmy własnoręcznie napisanej klasy `sieci Kohonena`. Sieć Kohonena jest używana do mapowania danych wielowymiarowych na dwuwymiarową mapę (zwaną także mapą cech), gdzie dane są reprezentowane jako wektory cech na wejściu sieci. Używa ona niezadziorowanych procesów uczenia, co oznacza, że nie wymaga etykiet ani danych treningowych do nauczenia się wzorców. Z tego powodu nie potrzebowaliśmy rozdzielenia zbioru na treneingowy oraz testowy. Proces uczenia sieci Kohonena skupia się na samoorganizacji neuronów, które są rozmieszczone na mapie cech. Neurony konkurują ze sobą o aktywację na podstawie odległości euklidesowej między wektorem wejściowym a wagami neuronów. W ten sposób na koniec powstaje mapa dowolnych wymiarów NxM z poszczególnymi produktami przypisanymi do danego punktu na mapie (np. punkt (3,2)). Pozwala nam to na rzeczywiste zwizualizowanie wyglądu przyszłego magazynu. Poniżej prezentujemy wygląd poszczególnych magazynów dla kwartałów Q1, Q2, Q3 oraz Q4. Wielkość kropek oznacza ilość produktów znajdującej się w tej samej strefie. Słowniki stref (klastrów) są zapisanego w folderze `Maps`.

2.5 Metryka

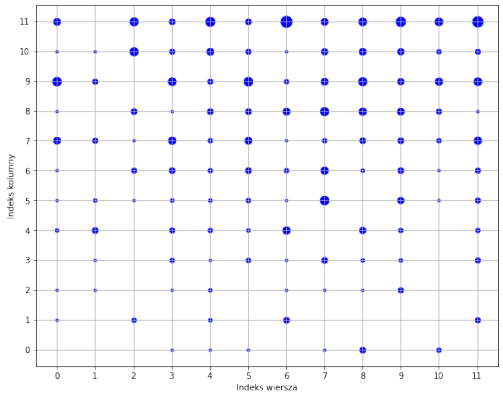
Do sprawdzenia słuszności naszego modelu postanowiliśmy zbudować metrykę liczącą minimalny dystans jaki pracownik musi przebyć po fabryce w celu skompletowania danego zamówienia. Przykładowo, gdy mamy produkt A w regionie (1,0), produkt B w regionie (3,5) oraz produkt C w regionie (2,1) uzyskalibyśmy odpowiedź, należy zacząć z punktu A, przejść do C a na koniec do B a odległość jaką wykonujemy (odległość euklidesowa) wynosi $\sqrt{1+1} + \sqrt{1+16} \approx 5.53$.



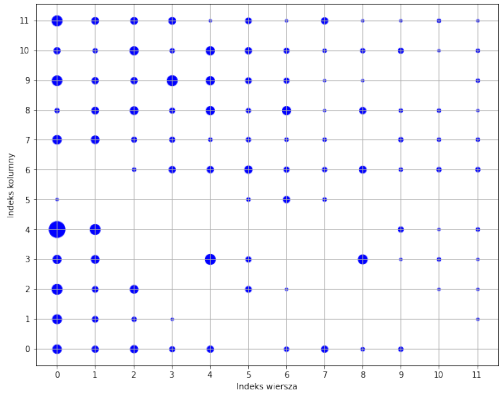
Rysunek 6: Wymodelowany plan magazynu w kwartale I.



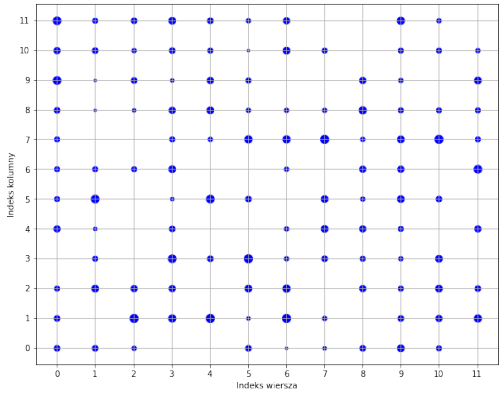
Rysunek 7: Wymodelowany plan magazynu w kwartale II.



Rysunek 8: Wymodelowany plan magazynu w kwartale III.



Rysunek 9: Wymodelowany plan magazynu w kwartale IV.



Rysunek 10: Przykład losowo wygenerowanego planu magazynu w kwartale I.

Żeby imitować najlepszą ścieżkę dla danego planu magazynu i danego zamówienia posłużyliśmy się algorytmem `Two Opt`. Jest to heurystyczny algorytm optymalizacyjny używany do rozwiązywania problemów trasy komiwojażera, który iteracyjnie poprawia rozwiązanie poprzez zamianę dwóch krawędzi w celu minimalizacji długości trasy.

Następnie 10 razy dla każdego kwartału losowo wygenerowaliśmy przykładowy plan magazynu. Dla każdego takiego planu porównaliśmy całkowitą trasę przejścia wszystkich zamówień w wygenerowanym planie z tym wymodelowanym.

3 Model Końcowy

Parametry uczenia modeli:

- Funkcja sąsiedztwa = Funkcja gaussowska
- Parametr uczenia $\alpha = 0.1$
- Parametr wygaszania uczenia $\lambda = 1000$
- Parametr sąsiedztwa $\sigma = 1.0$
- Liczba epoch = 100 (w przypadku modelu na kwartale I - 50)

Poniżej przedstawiamy wyniki przeprowadzonych testów na zaproponowanej przez nas metryce.

Model	**Iteracje**	**Średnia**	**Max**
Q1	10	9.37%	12.75%
Q2	10	19.94%	23.99%
Q3	10	15.05%	18.66%
Q4	10	22.52%	24.68%

Tabela 1: Tabela przedstawia procentowe wyniki poprawy jakości ułożenia w magazynie przy zastosowaniu utworzonych modeli w porownaniu z wylosowanymi modelami. Objaśnieie kolumn: Model: kwartał, którego dotyczy model; Iteracje: stworzone losowe plany magazynu wykorzystywane do porównania z utworzonym modelem; Średnia: Procent o jaki średnio nasz model poprawia optymalny dystans przejścia po wszystkich zamówieniach; Max: Procent o jaki maksymalnie nasz model poprawia optymalny dystans przejścia po wszystkich zamówieniach.

4 Możliwości Modelu

Zaproponowany przez nas model jest bardzo efektywnym i tanim w kontekście zgromadzenia danych rozwiązaniem problemu efektywnego magazynowania. Przy tym zawiera on bardzo dużo mozliwości kalibracyjnych dla użytkownika. Może on z łatwością dostosować go do swoich potrzeb za pomocą:

- kalibracji parametru interpolacyjnego λ - decyduje, czy ważniejsza jest realizacja zamówienia czy przetworzenie jak największej ilości produktów w ciągu dnia,
- manipulacji podziału czasowego - zamiast kwartałów klient może zmienić podział na lata,
- ograniczenia liczebności produktów znajdujących się w jednej sekcji (klastrze) w magazynie,
- określenia wymiarów magazynu do siatki Kohonena - pozwala to na bardzo dokładne przybliżenie przyszłych wymiarów magazynu.

5 Opis Githuba

- CODE - folder z kodem źródłowym
  - EDA.ipnyb - notatnik Python'a ze wstępną analizą danych wykorzystywana później do Feature Engineering,
  - MAIN_BHL.ipnyb - notatnik Python'a z głównym kodem źródłowym,
- Maps - pliki .txt ze słownikami przynależności produktów do odpowiednich sekcji,
- Photos - plik ze zdjęciami planów magazynów,
- Raw - wstepne zapiski i pliki,
- Task - folder z treścią zadania.