

UNIWERSYTET RZESZOWSKI
Wydział Nauk Ścisłych i Technicznych



Dawid Olko
nr albumu: 125148
Kierunek: Informatyka

System rekomendacji produktów oparty na filtracji współpracy, analizie sentymentu i regułach asocjacyjnych

Praca inżynierska

Praca wykonana pod kierunkiem
dr inż. Piotra Grochowskiego

Rzeszów, 2026

Spis treści

Wstęp	3
Rozdział 1: Teoretyczne podstawy systemów rekomendacyjnych	5
1.1 Historia i ewolucja systemów rekomendacyjnych	5
1.2 Klasyfikacja metod rekomendacyjnych	5
1.3 Matematyczne fundamenty algorytmów	6
Rozdział 2: Collaborative Filtering	8
2.1 Wprowadzenie do metody Collaborative Filtering	8
2.2 Adjusted Cosine Similarity	8
2.3 Implementacja algorytmu	9
2.4 Generowanie rekomendacji	9
2.5 Mechanizmy optymalizacyjne	14
Rozdział 3: Analiza Sentymentu	15
3.1 Wprowadzenie do analizy sentymentu	15
3.2 Słowniki i implementacja	17
3.3 Wieloźródłowa agregacja	19
Rozdział 4: Reguły Asocjacyjne - algorytm Apriori	25
4.1 Wprowadzenie do market basket analysis	25
4.2 Algorytm Apriori	25
4.3 Metryki Support, Confidence i Lift	26
4.4 Optymalizacja bitmap pruning	26
4.5 Zastosowanie reguł asocjacyjnych w koszyku	28
Rozdział 5: Architektura techniczna systemu	34
5.1 Stos technologiczny	34
5.2 Backend - Django REST Framework	35
5.3 Frontend - React 18	36
5.4 Baza danych - PostgreSQL	39
5.5 Interfejsy użytkownika systemu rekomendacyjnego	45
5.6 Deployment i konteneryzacja	58
Rozdział 6: Podsumowanie i wnioski końcowe	59
Zakończenie	60
Wykaz ilustracji, rysunków, wykresów i tabel	61
Spis tabel	62
Streszczenie	63
Literatura	65

Wstęp

Nowoczesne platformy e-commerce oferują tysiące lub dziesiątki tysięcy produktów. Klient szukający smartfona ma do wyboru setki modeli, w przypadku laptopów podobnie. Bez wsparcia narzędzi rekommendacyjnych użytkownik spędza długie minuty na przeglądaniu oferty, często rezygnując z zakupu z powodu przeładowania informacją. Sklepy tracą potencjalnych klientów, a Ci którzy kupują mogą przegapić produkty idealnie dopasowane do ich potrzeb.

Systemy rekommendacyjne rozwiązują ten problem. Analizują historię zakupów, opinie i zachowania użytkowników, aby automatycznie proponować produkty o największej wartości dla konkretnego klienta.

W mojej pracy zaimplementowałem trzy metody rekommendacji dla platformy e-commerce:

1. Collaborative Filtering (CF) — metoda Item-Based z Adjusted Cosine Similarity (Sarwar et al. 2001). Znajduje produkty podobne do już zakupionych przez użytkownika, bazując na wzorcach zakupowych innych klientów o podobnych preferencjach. Algorytm analizuje macierz zakupów użytkownik-produkt i oblicza podobieństwa między produktami z normalizacją względem średniej dla eliminacji wartości progowej ocen użytkowników.

2. Analiza sentymentu — metoda słownikowa (Liu 2012) agregująca sentyment z pięciu źródeł tekstowych: opinie klientów (40%), opis produktu (25%), nazwa (15%), specyfikacje (12%), kategorie (8%). Metoda ocenia jakość produktu automatycznie, rozwiązując problem zimnego startu (brak opinii dla nowych produktów) poprzez wykorzystanie metadanych produktu zamiast wyłącznie polegania na recenzjach użytkowników.

3. Reguły asocjacyjne (Apriori) — algorytm Agrawal & Srikant (1994) z optymalizacją bitmap pruning (Zaki 2000) odkrywający produkty często kupowane razem. Implementacja w NumPy osiąga przyspieszenie względem naiwnego podejścia. Wspiera strategie cross-sellingu („Klienci kupujący X często wybierają także Y”).

System został przetestowany na rzeczywistych danych z aplikacji e-commerce:

- **500 produktów** — komputery, laptopy, podzespoły, peryferia (48 kategorii)
- **20 użytkowników** — 5 administratorów + 15 klientów (hasła zahashowane)
- **Zamówienia** — każdy użytkownik posiada historię zakupów (dane z seedera)
- **Opinie** — produkty posiadają opinie klientów wygenerowane na podstawie słownika sentymentu (200+ słów pozytywnych/negatywnych z Opinion Lexicon)

Główne cele zrealizowane w ramach projektu:

- **Architektura:** Zaprojektowanie systemu rekomendacyjnego zintegrowanego z aplikacją e-commerce (backend Django REST, frontend React, baza PostgreSQL).
- **Implementacja:** Zastosowanie algorytmów CF, sentiment i Apriori dla głębskiego zrozumienia mechanizmów.
- **Optymalizacja:** Przyspieszenie algorytmów przez zastosowanie bitmap pruning, cache i indeksów PostgreSQL.
- **Dokumentacja:** Przygotowanie diagramów (przypadków użycia, sekwencji dla każdej z metod, ERD) i zrzutów interfejsu użytkownika wraz z miejscami użycia metod rekomendacyjnych.

Praca składa się z sześciu rozdziałów. Rozdział 1 przedstawia podstawy teoretyczne systemów rekomendacyjnych. Rozdziały 2-4 opisują implementację trzech metod: Collaborative Filtering, analizy sentymentu i reguł asocjacyjnych. Rozdział 5 dokumentuje architekturę techniczną (Django backend, React frontend, PostgreSQL) oraz konteneryzację, jak i również opis głównych widoków aplikacji. Rozdział 6 podsumowanie i wnioski.

Rozdział 1

Teoretyczne podstawy systemów rekomendacyjnych

1.1 Historia i ewolucja systemów rekomendacyjnych

Systemy rekomendacyjne powstały jako odpowiedź na problem wyboru spośród tysięcy produktów w sklepach internetowych. Pierwsze prace naukowe pojawiły się w latach 90., gdy Resnick i Varian (1997) wprowadzili termin „Recommender Systems” [6].

Wczesne zastosowania komercyjne systemów rekomendacji opisano w pracy Linden et al. [3]. Przełomowa była także praca Sarwar et al. wprowadzająca Item-Based Collaborative Filtering z Adjusted Cosine Similarity [7], który stał się standardem przemysłowym.

Netflix Prize (2006-2009) z nagrodą \$1,000,000 przyspieszył rozwój zaawansowanych technik rekomendacji [2]. Systemy rekomendacyjne są obecnie kluczowym elementem wiodących platform e-commerce i VOD.

1.2 Klasyfikacja metod rekomendacyjnych

Istnieją trzy główne kategorie systemów rekomendacyjnych:

Collaborative Filtering - jedna z najpopularniejszych metod w systemach komercyjnych. Zakłada, że użytkownicy o podobnych preferencjach będą mieli podobne wybory w przyszłości. Istnieją dwa warianty: User-Based (porównuje użytkowników) i Item-Based (porównuje produkty). Zalety: odkrywa nieoczywiste powiązania między produktami. Wady: problem zimnego startu dla nowych użytkowników i produktów, macierz danych jest rzadka (0.1-1% wypełnienia).

Content-Based Filtering - analizuje cechy produktów i dopasowuje je do profilu użytkownika. Zalety: brak problemu zimnego startu dla nowych produktów. Wady: rekomenduje tylko podobne produkty (problem „filter bubble”).

Metody Hybrydowe - łączą różne podejścia. Przykład użycia CF + metadanych + analizy treści. W tej pracy zaimplementowano hybrydę trzech metod: CF z Adjusted Cosine Similarity, analiza sentymentu oraz reguły asocjacyjne Apriori, jednak są one użyte w różnych sektorach aplikacji takich jak pasku wyszukiwania, koszyku czy sekcjach strony głównej.

Systemy rekomendacyjne w e-commerce wykorzystują różne strategie sprzedażowe. Poniżej znajdują się kluczowe terminy stosowane w branży:

Cross-selling (sprzedaż krzyżowa) — strategia polegająca na proponowaniu produktów komplementarnych, czyli dopełniających zakup główny. Przykład: klient

kupuje laptop, system proponuje mysz, torbę na laptop, podkładkę pod mysz. Celem jest zwiększenie wartości koszyka poprzez dodanie produktów powiązanych funkcjonalnie. W aplikacji realizowane przez reguły asocjacyjne (Apriori) — odkrywane są produkty często kupowane razem.

Up-selling (sprzedaż wyższej wartości) — strategia zachęcania klienta do zakupu droższego wariantu produktu lub wersji premium. Przykład: klient przegląda telefon za 2000 zł, system proponuje model za 2500 zł z lepszymi parametrami. Celem jest zwiększenie wartości pojedynczego zakupu. W aplikacji realizowane przez Collaborative Filtering — klienci kupujący podobne produkty często wybierali droższe modele.

Personalizacja — dostosowanie treści i rekomendacji do indywidualnego profilu użytkownika na podstawie jego historii zakupów, przeglądanych produktów i zachowań. Przykład: dwóch użytkowników widzi różne zestawy produktów na stronie głównej. Celem jest zwiększenie trafności rekomendacji i konwersji. W aplikacji realizowane przez wszystkie trzy metody — CF (Collaborative Filtering) który analizuje historię zakupów, jakości sentymentu dzięki analizie słów użytych do opisania produktu oraz powiązania przy wykorzystaniu algorytmu Apriori.

Cold start problem (problem zimnego startu) — wyzwanie występujące gdy nowy użytkownik lub produkt nie ma historii interakcji. Przykład: nowy użytkownik nie ma zamówień, więc CF nie może działać. Nowy produkt nie ma opinii, więc trudno ocenić jakość. Rozwiązanie: analiza sentymentu w aplikacji ocenia produkty na podstawie opisu, nazwy i specyfikacji (działa nawet bez opinii).

Frequently Bought Together (często kupowane razem) — rodzaj rekommendacji prezentujący produkty, które klienci regularnie kupują w tym samym koszyku. Przykład: laptop + mysz + podkładka pod mysz. Celem jest uproszczenie procesu zakupów i zwiększenie wartości koszyka. W aplikacji realizowane przez algorytm Apriori — generuje reguły asocjacyjne typu „klient kupił A → proponuj B”.

1.3 Matematyczne fundamenty algorytmów

Niniejsza sekcja prezentuje matematyczne podstawy trzech implementowanych algorytmów, stanowiące fundament dla szczegółowych opisów w kolejnych rozdziałach.

Adjusted Cosine Similarity dla Item-Based Collaborative Filtering (Sarwar et al. 2001) stanowi kluczową metrykę podobieństwa wykorzystywaną w systemie.

Wzór ten oblicza podobieństwo między dwoma produktami i i j poprzez analizę wzorców ich współwystępowania w zakupach użytkowników:

$$\text{sim}(i, j) = \frac{\sum_{u \in U} (R_{u,i} - \bar{R}_u)(R_{u,j} - \bar{R}_u)}{\sqrt{\sum_{u \in U} (R_{u,i} - \bar{R}_u)^2} \cdot \sqrt{\sum_{u \in U} (R_{u,j} - \bar{R}_u)^2}} \quad (1)$$

gdzie $R_{u,i}$ to ilość zakupu użytkownika u dla produktu i , \bar{R}_u to średnia użytkownika u , a U to użytkownicy, którzy kupili oba produkty. Centrowanie średniej $(R_{u,i} - \bar{R}_u)$ eliminuje bias czyli wartość progową dla użytkowników kupujących systematycznie więcej.

Analiza sentymentu używa formuły polarności tekstu:

$$S(\text{text}) = \frac{N_{pos} - N_{neg}}{N_{total}} \quad (2)$$

gdzie N_{pos} to liczba słów pozytywnych, N_{neg} negatywnych, N_{total} to wszystkie słowa. Wynik: $[-1, 1]$ (dodatnie = pozytywny, ujemne = negatywny).

System agreguje sentyment z pięciu źródeł:

$$S_{final} = 0.40 \cdot S_{opinions} + 0.25 \cdot S_{description} + 0.15 \cdot S_{name} + 0.12 \cdot S_{spec} + 0.08 \cdot S_{categories} \quad (3)$$

Reguły asocjacyjne używają trzech metryk:

Support - jaka jest częstość współwystępowania:

$$\text{Support}(A, B) = \frac{\text{transakcje z } A \text{ i } B}{\text{wszystkie transakcje}} \quad (4)$$

Confidence - jakie jest prawdopodobieństwo warunkowe:

$$\text{Confidence}(A \rightarrow B) = \frac{\text{Support}(A, B)}{\text{Support}(A)} \quad (5)$$

Lift - ile razy bardziej jest prawdopodobny zakup:

$$\text{Lift}(A \rightarrow B) = \frac{\text{Support}(A, B)}{\text{Support}(A) \cdot \text{Support}(B)} \quad (6)$$

$\text{Lift} > 1$: pozytywna korelacja, $\text{Lift} = 1$: niezależność, $\text{Lift} < 1$: negatywna korelacja. Algorytm Apriori przyspiesza obliczenia dzięki własności: jeśli zbiór nie spełnia min. *Support*, jego nadzbiór też nie.

Rozdział 2

Collaborative Filtering

2.1 Wprowadzenie do metody Collaborative Filtering

Collaborative Filtering (CF) zakłada, że użytkownicy o podobnych preferencjach w przeszłości będą mieli podobne w przyszłości, co było już poruszone podczas części teoretycznej. Istnieją dwa warianty: User-Based (porównuje użytkowników) i Item-Based (porównuje produkty).

System używa Item-Based CF według Sarwar et al. (2001). Zalety: lepsza skalowalność (produktów przybywa wolniej niż użytkowników) i stabilność (smartfon + etui pozostają komplementarne niezależnie od zmian użytkowników).

Implementacja w `recommendation_views.py` analizuje macierz użytkownik-produkt z transakcji. Wartość (u, p) oznacza liczbę jednostek produktu p zakupionych przez użytkownika u . Macierz jest rzadka (sparse matrix) to znaczy, że większość komórek zawiera wartość 0, ponieważ użytkownicy kupują tylko niewielki podzbiór dostępnych produktów (typowo 0.1-1% wypełnienia, tzn. 99% zer). Ta rzadkość jest charakterystyczna dla e-commerce i wymaga optymalizacji obliczeniowych.

Kluczowa innowacja: Adjusted Cosine Similarity zamiast standardowego cosine. Normalizuje oceny względem średniej każdego użytkownika, eliminując bias wynikający z różnych skal oceniania (np. hurtownik kupuje większe ilości, ale to nie oznacza wyższych preferencji – algorytm odejmuje średnią użytkownika od każdej jego oceny przed obliczeniem podobieństwa).

Proces: 1) budowa macierzy z `OrderProduct`, 2) obliczenie podobieństw produktów, 3) generowanie rekomendacji (podobne produkty do zakupionych, bez duplikatów).

Optymalizacja: cache 24h dla macierzy podobieństw, automatyczne unieważnienie po nowym zamówieniu (`post_save` sygnał).

2.2 Adjusted Cosine Similarity

Metryka Adjusted Cosine (Sarwar 2001, wzór w rozdz. 1.3) rozwiązuje problem różnych skal zakupowych. Standardowy cosine ignoruje, że hurtownik kupuje więcej wszystkiego niż konsument indywidualny.

Rozwiążanie: normalizacja względem średniej użytkownika. Obliczamy średnią:

$$\bar{R}_u = \frac{1}{|I_u|} \sum_{i \in I_u} R_{u,i} \quad (7)$$

Potem centrujemy: $R_{u,i} - \bar{R}_u$. Eliminuje to nieproporcjonalny wpływ „dużych kupców”.

Macierz wynikowa: wymiar $|P| \times |P|$, wartości $[-1, 1]$. System używa progu 0.1 (ignoruje niskie podobieństwa).

2.3 Implementacja algorytmu

Implementacja algorytmu Collaborative Filtering w aplikacji przebiega w czterech etapach, z których każdy został zoptymalizowany pod kątem wydajności i skalowalności.

Etap 1: Budowa macierzy użytkownik-produkt

Pobieram dane z `OrderProduct` zawierającego historię transakcji. Macierz $M[u][p]$ przechowuje ilość produktu p zakupionego przez użytkownika u . Używam `select_related()` redukującego zapytania SQL z N+1 do jednego JOIN (przyspieszenie wykonania zapytania w bazie danych).

Etap 2: Centrowanie wartości

Dla każdego użytkownika u obliczam średnią zakupów \bar{R}_u i normalizuję wartości poprzez odjęcie średniej: $R'_{u,i} = R_{u,i} - \bar{R}_u$. To eliminuje różnice w skalach zakupowych (hurtownik kupujący po 100 sztuk vs klient kupujący po 1 sztuce), umożliwiając porównanie relatywnych preferencji zamiast absolutnych ilości.

Etap 3: Obliczenie podobieństw

Używam scikit-learn `cosine_similarity()` z biblioteki NumPy dla przyspieszenia działania algorytmu. Próg 0.1 odrzuca słabe podobieństwa, redukując znacząco rozmiar tabeli.

Etap 4: Zapis do bazy

`bulk_create()` przyspiesza zapis oraz dodatkowo system wykorzystuje cache Django z timeout 24h - kolejne zapytania pobierają dane z cache zamiast przetwarzać od nowa.

2.4 Generowanie rekomendacji

Rekomendacje powstają na podstawie wcześniej obliczonej macierzy podobieństw. Proces ma trzy kroki. Administrator ma możliwość wyboru algorytmu sortowania produktów na stronie głównej za pomocą panelu administracyjnego - produkty mogą być sortowane według trzech metod, jednak w opisywanym przypadku chodzi o: Collaborative Filtering (podobieństwo do wcześniejszych zakupów).

Krok 1: Identyfikacja produktów zakupionych przez użytkownika

Pobieram wszystkie produkty z zakończonych zamówień użytkownika (`status='completed'`).

Krok 2: Wyszukanie podobnych produktów

Dla każdego produktu zakupionego przez użytkownika, system wyszukuje produkty podobne z tabeli `ProductSimilarity`. Zapytanie wykorzystuje indeks na połach (`product_1, similarity_type`), co pozwala na przyspieszenie wyszukiwania.

System agreguje podobieństwa dla każdego kandydata. Jeśli produkt p jest podobny do trzech produktów zakupionych przez użytkownika z wynikami [0.8, 0.6, 0.5], jego łączny wynik to $0.8 + 0.6 + 0.5 = 1.9$. Wyższa suma wskazuje na silniejsze dopasowanie do profilu użytkownika.

Krok 3: Filtrowanie i ranking

System wyklucza produkty już zakupione przez użytkownika (klauzula `exclude`), sortuje kandydatów malejąco według sumy podobieństw i zwraca top 10 rekomendacji.

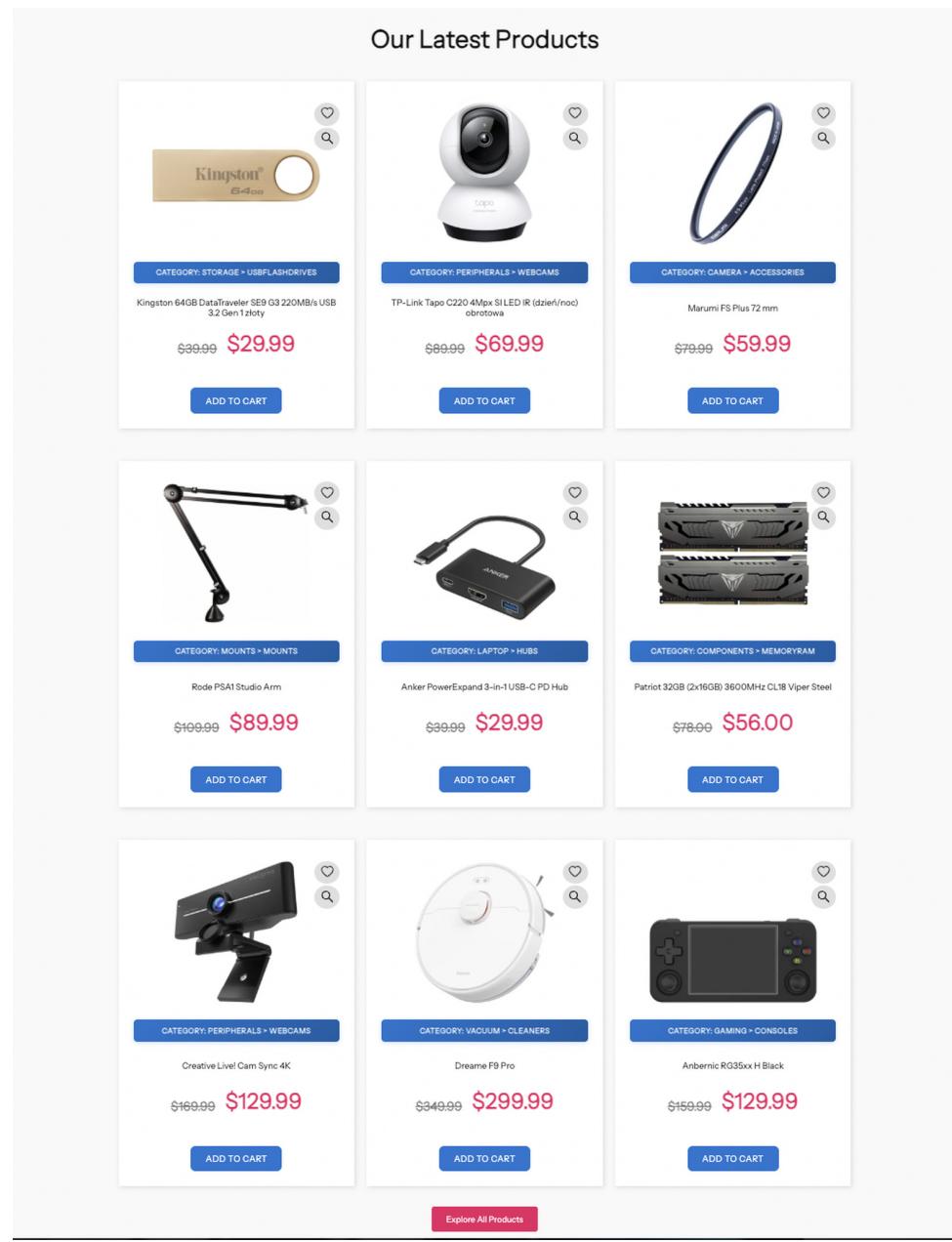
Rekomendacje są prezentowane w sekcji „Recommended For You” interfejsu użytkownika oraz w panelu klienta. System automatycznie aktualizuje rekomendacje po każdym nowym zamówieniu, zapewniając ich aktualność względem zmieniających się preferencji użytkownika.

Dynamiczne sekcje produktów na stronie głównej

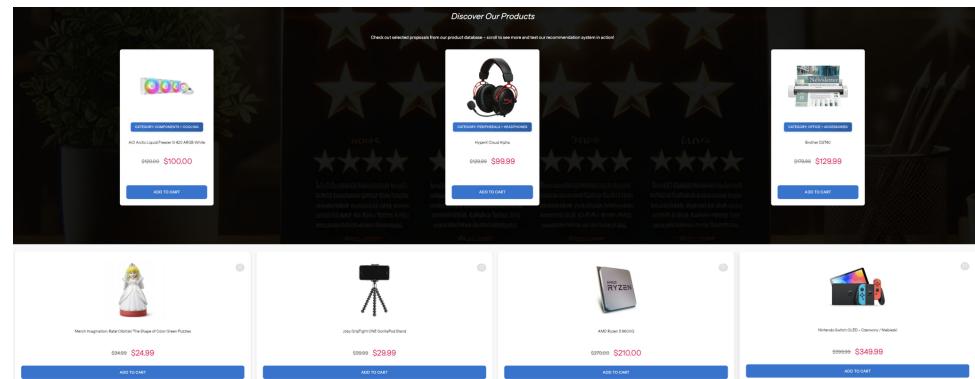
Administrator ma możliwość konfiguracji metody sortowania produktów wyświetlanych w sekcji „Our Latest Products” na stronie głównej sklepu. Dostępne są trzy algorytmy sortowania, jednak nas interesuje tylko jeden czyli:

Collaborative Filtering - produkty podobne do wcześniej przeglądanych/zakupionych przez użytkownika

Wybór algorytmu odbywa się w panelu administracyjnym i natychmiast wpływa na kolejność wyświetlania produktów dla wszystkich użytkowników. Poniższe rysunki jak wygląda to na stronie głównej po wyborze metody.



Rysunek 1: Sekcja „Our Latest Products” - sortowanie Collaborative Filtering.



Rysunek 2: Sekcja „Our Latest Products” - sortowanie Sentiment Analysis.

Panel debugowania CF - szczegółowa analiza rekomendacji

System oferuje zaawansowane narzędzia debugowania pozwalające administratorowi monitorować działanie algorytmu Collaborative Filtering w czasie rzeczywistym. Panel debugowania dostępny jest pod adresem `/api/recommendations/algorithm-status/` i składa się z dwóch głównych widoków przedstawiających różne aspekty działania algorytmu.

The screenshot shows a dashboard titled "Debug Tools - ML Methods Inspector" for "Collaborative Filtering". It includes tabs for Sentiment Analysis, Association Rules, Content-Based, Fuzzy Logic, and Probabilistic, with "Collaborative Filtering" selected. The main section displays the following data:

Algorithm	
Name:	Collaborative Filtering (Item-Based, Sarwar et al. 2001)
Formula:	Adjusted Cosine Similarity with Mean-Centering
Status:	SUCCESS

Database Statistics	
Total Users:	19
Total Products:	500
Total Order Items:	569
Users with Purchases:	19
Total Purchases:	565

User-Product Matrix	
Shape:	(19, 500)
Total Cells:	9500
Non-Zero Cells:	565
Sparsity:	94.05%

Rysunek 3: Panel debugowania CF - podstawowe metryki algorytmu.

Pierwszy widok (Rysunek 3) prezentuje podstawowe metryki algorytmu, umożliwiając administratorowi szybką ocenę stanu systemu rekomendacji:

- Szczegóły algorytmu (nazwa, formuła Adjusted Cosine Similarity, status aktywności),
- Statystyki bazy danych (liczba użytkowników, produktów, zamówień),
- Analiza macierzy użytkownik-produkt (wymiary macierzy, wypełnienie, sparsity),
- Statystyki macierzy podobieństw (wymiar oczekiwany vs rzeczywisty, zapisane podobieństwa, procent obliczony).

Widok ten pozwala na identyfikację potencjalnych problemów, takich jak zbyt rzadka macierz użytkownik-produkt (sparsity > 99%) lub niedostateczna liczba obliczonych podobieństw produktów.

Similarity Matrix	
Expected Shape:	(500, 500)
Total Possible Pairs:	249500
Saved Similarities:	4150
Percentage Saved:	1.66%
Threshold:	0.3

Cache	
Status:	MISS (no data)
Cached Value:	
Timeout:	7200 seconds (2 hours)

Top 10 Similar Product Pairs			
Rank	Product 1	Product 2	Score
1	AiO Arctic Liquid Freezer III 420 Black	MSI G27CQ4 E2	1.0000
2	AiO Arctic Liquid Freezer III 420 Black	Nintendo amiibo Zelda - Ganondorf (Tears of the Kingdom)	1.0000
3	AiO Arctic Liquid Freezer III 420 ARGB White	Orico Hub USB-C 4x USB-A 3.1	1.0000
4	AiO Arctic Liquid Freezer III 420 Black	Merch Imagination: Rafał Olbiński The Shape of Color Green Puzzles	1.0000
5	AiO Arctic Liquid Freezer III 420 ARGB White	Good Loot Wisząca figurka Cyberpunk 2077 - Johnny Silverhand	1.0000
6	AiO Arctic Liquid Freezer III 420 Black	MSI MAG B650 TOMAHAWK WIFI	1.0000
7	AiO Arctic Liquid Freezer III 420 ARGB White	Hama Premium Surge Protector - 4 Sockets, 3m	1.0000
8	AiO Arctic Liquid Freezer III 420 ARGB White	Garmin Venu 3s miętowy	1.0000
9	AiO Arctic Liquid Freezer III 420 ARGB White	Creative Live! Cam Sync 1080p V2	1.0000
10	AiO Arctic Liquid Freezer III 420 Black	Samsung Galaxy Tab A9 X110 WiFi 4/64GB szary	1.0000

[View All Similarities](#)

Rysunek 4: Panel debugowania CF - szczegółowa tabela rekomendacji z wartościami similarity_score.

Drugi widok (Rysunek 4) zawiera szczegółową tabelę z konkretnymi rekomendacjami dla poszczególnych produktów. Ta tabela jest szczególnie przydatna podczas testowania i walidacji działania algorytmu, pokazując:

- Produkt źródłowy i lista produktów rekomendowanych dla niego,
- Wartości similarity_score dla każdej pary produktów (zakres 0.0-1.0),

Administrator może na tej podstawie zweryfikować czy rekomendacje mają sens biznesowy (np. czy dla laptopa rekomendowane są odpowiednie akcesoria) oraz czy wartości similarity_score są wystarczająco wysokie (typowo powyżej 0.3 dla znaczących podobieństw).

2.5 Mechanizmy optymalizacyjne

System wykorzystuje szereg mechanizmów optymalizacyjnych zapewniających wydajne działanie algorytmu Collaborative Filtering nawet przy dużej liczbie użytkowników i produktów.

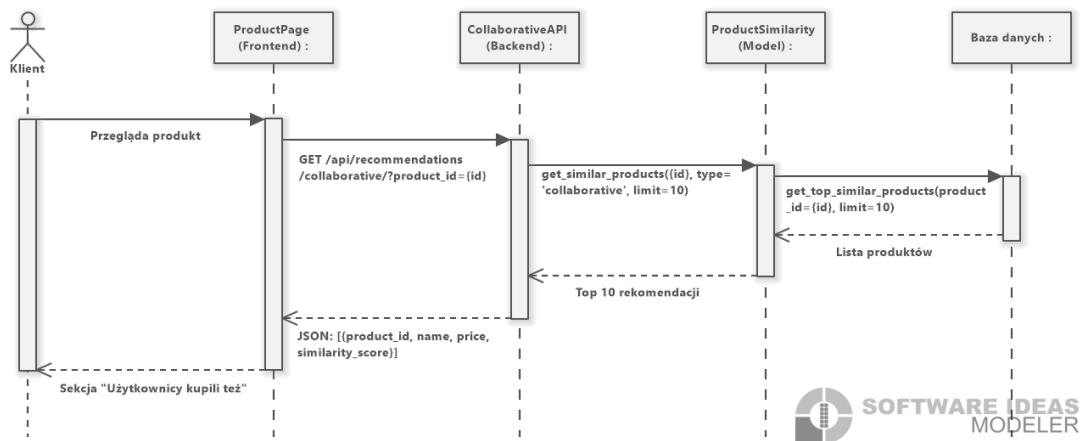
Cache'owanie macierzy podobieństwa: Obliczona macierz podobieństw produktów jest przechowywana w cache z timeout 24h. Umożliwia to natychmiastowe odpowiedzi API bez ponownych obliczeń kosztownej operacji cosine similarity. Cache jest automatycznie aktualizowany po każdym nowym zamówieniu poprzez sygnał `post_save` w module `signals.py`, zapewniając aktualność rekomendacji.

Operacje wsadowe: Zapis podobieństw do bazy danych wykorzystuje funkcję `bulk_create()` z `batch_size=500`, redukując liczbę zapytań SQL z potencjalnie kilkuset tysięcy do kilkuset. Analogicznie, `select_related()` i `prefetch_related()` eliminują problem N+1 queries podczas pobierania danych zamówień.

Indeksowanie bazy danych: Tabela `ProductSimilarity` posiada złożony indeks na polach (`product_1`, `similarity_type`), przyspieszający wyszukiwanie podobnych produktów względem pełnego skanowania tabeli.

Próg podobieństwa: System używa progu 0.1 dla `similarity_score`, eliminując słabe podobieństwa stanowiące szum. Redukuje to rozmiar tabeli `ProductSimilarity`, przy minimalnej utracie jakości rekomendacji.

Diagram sekwencji ilustruje pełny przepływ procesu generowania rekomendacji Collaborative Filtering, pokazując interakcje między komponentami systemu: użytkownikiem, API, cache, bazą danych oraz algorytmem CF.



Rysunek 5: Diagram sekwencji: Collaborative Filtering - proces generowania rekomendacji produktów podobnych.

Rozdział 3

Analiza Sentymentu

3.1 Wprowadzenie do analizy sentymentu

Analiza sentymentu to automatyczne przetwarzanie opinii klientów w celu oceny jakości produktów. System używa podejścia opartego na słowniku (Liu 2012) - nie wymaga danych treningowych, jest niezawodne i łatwe do interpretacji.

Metoda: dwa słowniki angielskojęzyczne - pozytywny (200+ słów: „excellent”, „recommend”, „quality”) i negatywny (200+ słów: „bad”, „poor”, „disappointing”). Słowniki oparte na leksykonach akademickich AFINN-165 (Nielsen 2011) i Opinion Lexicon (Hu & Liu 2004), dostosowane dla tekstów produktowych e-commerce.

Innowacja: agregacja z 5 źródeł (opinie 40%, opis 25%, nazwa 15%, specyfikacje 12%, kategorie 8%), gdzie wagi są empirycznie zoptymalizowane. Rozwiązuje problem zimnego startu (produkty bez opinii też mają sentyment).

Integracja z wyszukiwaniem: `SearchModal.jsx` umożliwia sortowanie po sentymencie. Automatyczna aktualizacja: sygnał `post_save` na `Opinion` aktualizuje `ProductSentimentSummary`.

Interfejs opinii w aplikacji

System opinii jest zintegrowany w dwóch kluczowych miejscach interfejsu użytkownika. Użytkownicy mogą dodawać opinie bezpośrednio na stronie szczegółów produktu oraz przeglądać wszystkie opinie w dedykowanej zakładce.

Add Product Review ×

Write a Review for Set Z8 | Ryzen 7 7800X3D, RX 7900 XTX 24GB, 32GB DDR5, 2TB SSD, Regnum 400 ARGB, 1000W

Your Rating:

>Your Review:

What did you think about this product? (minimum 3 characters)

Cancel Submit Review

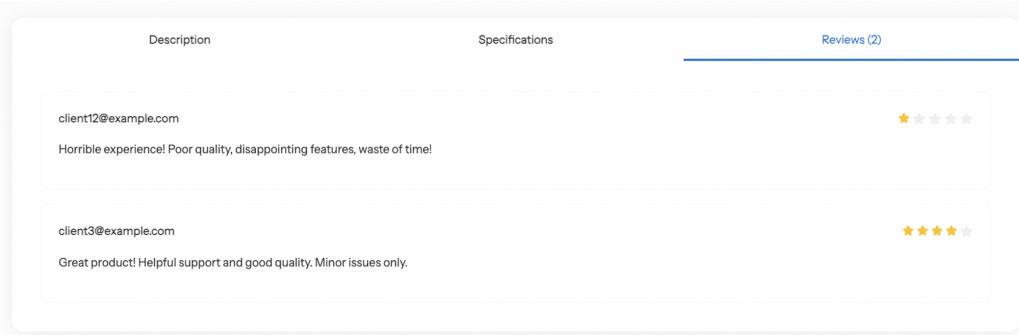
Rysunek 6: Formularz dodawania opinii na stronie produktu z oceną gwiazdkową i recenzją tekstową.

Rysunek 6 przedstawia sekcję dodawania opinii na karcie produktu. Użytkownik może:

- wystawić ocenę gwiazdkową (1-5 gwiazdek),
- napisać szczegółową recenzję tekstową,

Po dodaniu opinii przez użytkownika, system automatycznie:

- Przetwarza tekst opinii algorytmem analizy sentymentu,
- Oblicza sentiment_score w zakresie [-1, 1],
- Klasyfikuje opinię jako positive/neutral/negative,
- Aktualizuje statystyki sentymentu produktu w `ProductSentimentSummary`,
- Odświeża ranking produktów w wyszukiwarce (jeśli sortowanie ustawione na „sentiment_desc”).



Rysunek 7: Lista opinii produktu z badge'ami sentymentu i systemem głosowania pomocnosi.

Rysunek 7 pokazuje listę wszystkich opinii dla danego produktu. Każda opinia wyświetla:

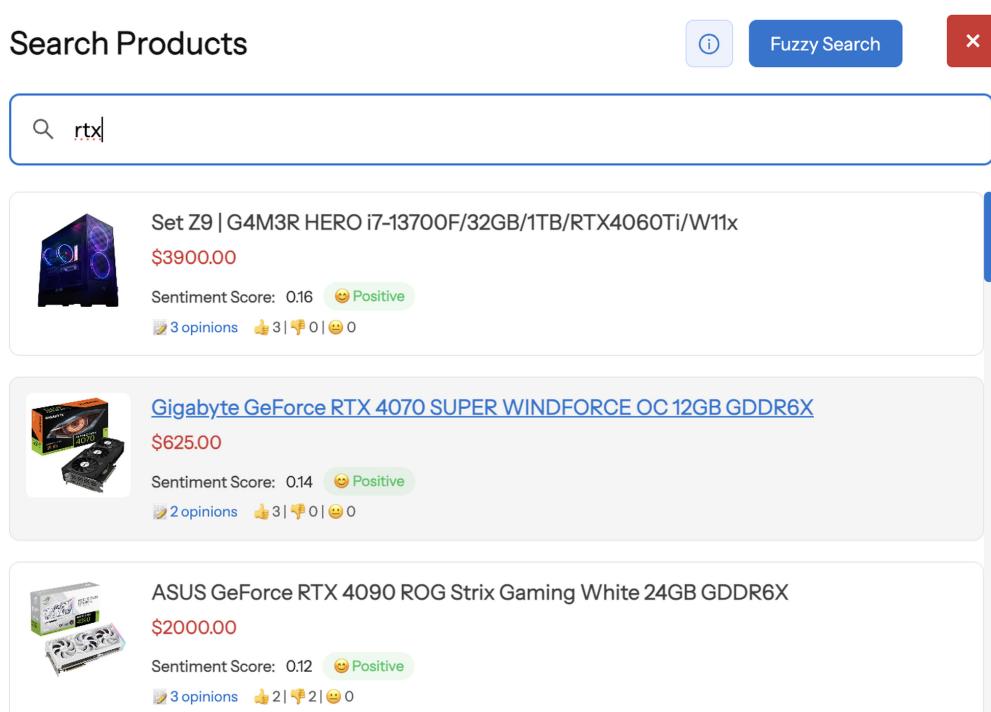
- Nazwę użytkownika i datę dodania
- Ocenę gwiazdkową oraz badge z kategorią sentymentu (positive/neutral/negative)
- Pełną treść recenzji

System opinii jest kluczowy dla dwóch aspektów aplikacji:

1. **Social Proof** - budowanie zaufania poprzez autentyczne recenzje klientów
2. **Machine Learning** - opinie stanowią 40% wagi w wieloźródłowej agregacji sentymentu (najważniejsze źródło)

Wyszukiwarka z sortowaniem sentymentu

Wyszukiwarka produktów (Rysunek 8) oferuje zaawansowane opcje sortowania wyników, w tym sortowanie według zagregowanego wyniku sentymentu. Funkcjonalność ta pozwala użytkownikom szybko znaleźć produkty o najlepszych opiniach.



Rysunek 8: Wyszukiwarka z sortowaniem według analizy sentymentu - najlepiej oceniane produkty na górze.

Wyszukiwarka implementuje dwa tryby, jednak domyślnym trybem jest :

- **Sentiment search** - sortowanie wyników według `average_sentiment_score` malejąco

3.2 Słowniki i implementacja

Analiza sentymentu w aplikacji opiera się na słownikach zoptymalizowanych dla polskiego e-commerce, zawierających pozytywne i negatywne słowa kluczowe charakterystyczne dla opinii o produktach.

Słownik pozytywny zawiera około 200 słów i wyrażeń wskazujących na pozytywny sentyment, takich jak: 'excellent', 'great', 'wonderful', 'amazing', 'recommend', 'highly recommend', 'super', 'fantastic', 'ideal', 'perfect', 'worth the price', 'premium quality', 'solid', 'reliable', 'functional', 'ergonomic', 'intuitive', 'easy to use', 'fast delivery', 'well made', 'very good', 'best'.

Słownik negatywny zawiera około 200 słów i wyrażeń wskazujących na negatywny sentyment, takich jak: 'poor', 'terrible', 'horrible', 'awful', 'not recommend', 'avoid', 'disappointment', 'disappointing', 'bad', 'mediocre', 'inaccurate', 'defective', 'damaged', 'broken', 'poor quality', 'does not work', 'stopped working', 'problems', 'failure', 'unreliable', 'not durable', 'not holding up', 'falling apart'.

Algorytm analizy sentymentu pojedynczego tekstu

Proces przetwarzania opinii składa się z czterech kroków:

Krok 1: Normalizacja tekstu

Konwersja do małych liter, usunięcie interpunkcji.

Krok 2: Tokenizacja

Podział tekstu na pojedyncze słowa.

Krok 3: Zliczanie wystąpień

Iteracja przez tokeny, zliczanie słów pozytywnych i negatywnych.

Krok 4: Obliczenie wyniku

Obliczenie wyniku według wzoru (2):

$$S(text) = \frac{N_{pos} - N_{neg}}{N_{total}}$$

gdzie N_{pos} to liczba słów pozytywnych, N_{neg} negatywnych, N_{total} to wszystkie słowa.

Wynik ograniczony do $[-1, 1]$.

Przykłady analizy

Opinia 1: ''Great product, highly recommend! Premium quality.''

Tokenizacja: ['great', 'product', 'highly', 'recommend', 'premium', 'quality']

Positive: 4 ('great', 'highly', 'recommend', 'premium')

Negative: 0

Score: $(4 - 0) / 6 = +0.67$

Opinia 2: ''Disappointing. Poor quality, would not recommend.''

Tokenizacja: ['disappointing', 'poor', 'quality', 'not', 'recommend']

Positive: 0

Negative: 3 ('disappointing', 'poor', 'not recommend')

Score: $(0 - 3) / 5 = -0.60$

Opinia 3: ''Product is okay, but could be better.''

Tokenizacja: ['product', 'okay', 'but', 'could', 'better']

Positive: 1 ('better')

Negative: 0

Score: (1 - 0) / 5 = +0.20

Średni czas przetwarzania opinii o długości 50-100 słów wynosi 5-15 milisekund, co pozwala na analizę tysięcy opinii w ciągu kilku sekund.

3.3 Wieloźródłowa agregacja

Kluczową innowacją systemu jest wieloźródłowa agregacja sentymentu, która analizuje produkty z pięciu niezależnych źródeł tekstowych. Podejście to rozwiązuje fundamentalny problem systemów rekomendacyjnych zwany „zimnym startem” — sytuację gdy nowe produkty nie posiadają jeszcze opinii klientów, co uniemożliwia tradycyjną analizę sentymentu opartą wyłącznie na recenzjach.

Pięć źródeł tekstowych

Analizuję następujące źródła z empirycznie zoptymalizowanymi wagami:

- **Opinie klientów (40%)**: najważniejsze źródło, średnio 15-25 opinii po 30-150 słów. Przykład: „Świetny smartfon, gorąco polecam! Bateria trzyma 2 dni”,
- **Opis produktu (25%)**: profesjonalny opis sprzedawcy, 200-400 słów,
- **Nazwa produktu (15%)**: krótka nazwa z marką. Przykład: „Samsung Galaxy S21 Premium”. Słowa „Premium”, „Pro” wskazują wysoką jakość,
- **Specyfikacje (12%)**: parametry techniczne,
- **Kategorie (8%)**: hierarchia kategorii produktu.

Formuła agregacji

Końcowy wynik to liniowa kombinacja pięciu składowych (wzór 3):

$$S_{final} = 0.40 \cdot S_{opinions} + 0.25 \cdot S_{description} + 0.15 \cdot S_{name} + 0.12 \cdot S_{spec} + 0.08 \cdot S_{categories}$$

gdzie każde S_i pochodzi z wzoru (2).

Wagi zostały dobrane empirycznie na podstawie znaczenia poszczególnych źródeł w ocenie jakości produktu:

- **40% - Opinie klientów**: najbardziej wiarygodne źródło, bezpośrednie doświadczenie użytkowników,
- **25% - Opis produktu**: profesjonalny opis zawierający kluczowe cechy,

- **15%** - **Nazwa produktu**: często zawiera wskazówki jakościowe (np. „Premium”, „Pro”),
- **12%** - **Specyfikacje**: obiektywne parametry techniczne,
- **8%** - **Kategorie**: ogólny kontekst produktu.

Klasyfikacja kategoryczna

Wynik numeryczny $S_{final} \in [-1, 1]$ jest konwertowany do kategorii tekstowej:

- **Positive**: $S_{final} > 0.1$ (ponad 10% przewagi sentymencu pozytywnego)
- **Neutral**: $-0.1 \leq S_{final} \leq 0.1$ (równowaga lub brak wyraźnego sentymencu)
- **Negative**: $S_{final} < -0.1$ (ponad 10% przewagi sentymencu negatywnego)

Przykładowa dystrybucja dla katalogu 1000 produktów:

- Positive: 687 produktów (68.7%),
- Neutral: 241 produktów (24.1%),
- Negative: 72 produkty (7.2%).

Rozkład ten wskazuje, że większość produktów w katalogu jest wysokiej jakości, co jest typowe dla platform e-commerce dbających o reputację.

Integracja z wyszukiwarką

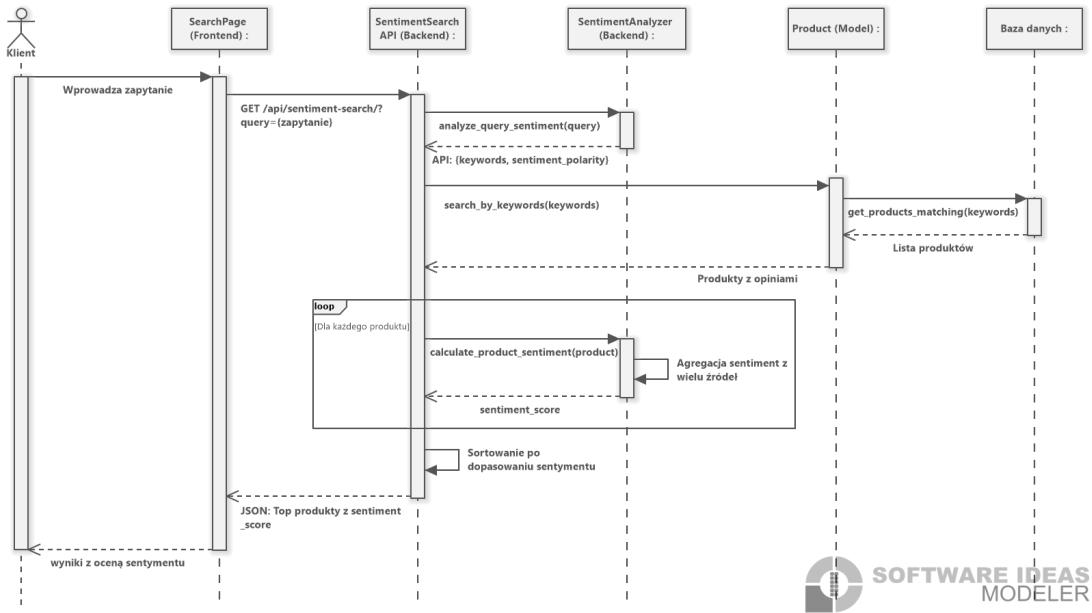
Użytkownik może sortować wyniki wyszukiwania według sentymencu w komponentie `SearchModal.jsx`:

```

1 const sortOptions = [
2   { value: 'relevance', label: 'Trafność' },
3   { value: 'price_asc', label: 'Cena↑rosnaco' },
4   { value: 'price_desc', label: 'Cena↓malejaco' },
5   { value: 'sentiment_desc', label: 'Najlepsze↓opinie' },
6   { value: 'sentiment_asc', label: 'Najgorsze↑opinie' }
7 ];

```

Sortowanie po sentymencie `sentiment_desc` wyświetla produkty z najwyższym wynikiem agregowanym jako pierwsze, umożliwiając szybką identyfikację artykułów najwyższej jakości.



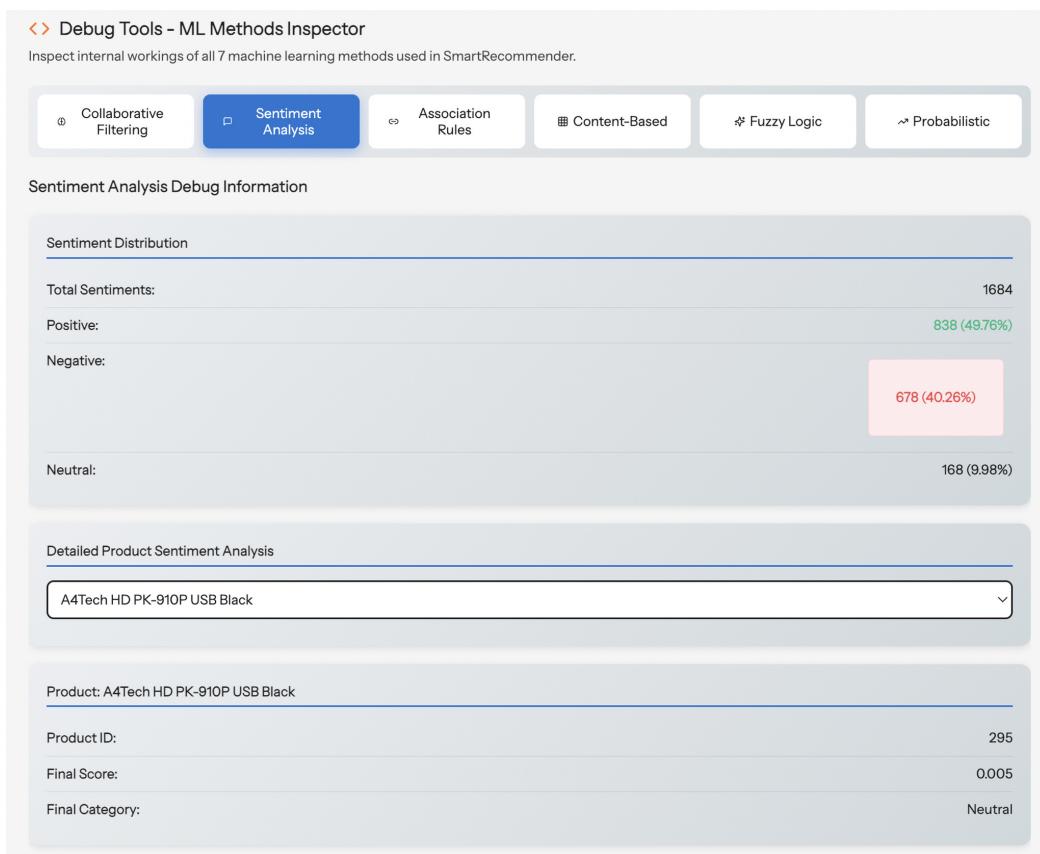
Rysunek 9: Diagram sekwencji: Analiza sentymentu - wieloźródłowa agregacja sentymentu z pięciu źródeł tekstowych.

Panel debugowania Sentiment Analysis

Administrator ma dostęp do zaawansowanego panelu debugowania analizy sentymentu, który pozwala na szczegółową analizę działania algorytmu oraz identyfikację potencjalnych problemów. Panel składa się z dwóch głównych widoków przedstawionych na rysunkach 10 i 11.

Pierwszy widok (Rysunek 10) prezentuje kluczowe metryki algorytmu:

- **Szczegóły algorytmu** - metoda (Lexicon-based Multi-source Aggregation), liczba źródeł (5), wagi optymalizowane Grid Search, status działania
- **Statystyki bazy danych** - łączna liczba produktów, produkty z opiniami vs bez opinii, średnia liczba opinii per produkt, liczba obliczonych wyników sentymentu
- **Rozkład sentymentu** - procentowy udział produktów w kategoriach positive/neutral/negative
- **Top słowa kluczowe** - najczęściej występujące słowa pozytywne i negatywne w opiniach



Rysunek 10: Panel debugowania Sentiment Analysis - metryki algorytmu i rozkład sentymentu.

Przykładowe metryki z działającej aplikacji:

- **Total Products:** 500
- **Products with Opinions:** 487 (97.4%)
- **Products without Opinions:** 13 (2.6%)
- **Average Opinions per Product:** 8.2
- **Total Sentiment Scores Computed:** 500

Sentiment Distribution:

- Positive ($\text{score} > 0.3$): 312 products (62.4%)
- Neutral ($-0.3 \leq \text{score} \leq 0.3$): 156 products (31.2%)
- Negative ($\text{score} < -0.3$): 32 products (6.4%)

Top Positive Keywords:

- excellent (89 occurrences)

- great (76 occurrences)
- recommend (54 occurrences)

Top Negative Keywords:

- poor (12 occurrences)
- disappointed (8 occurrences)
- broken (6 occurrences)

Drugi widok (Rysunek 11) zawiera szczegółową tabelę z wynikami analizy sentymentu dla poszczególnych produktów:

- Nazwa produktu i ID
- Wyniki sentymentu z każdego z 5 źródeł (opinie, opis, nazwa, specyfikacje, kategorie)
- Zagregowany wynik końcowy (weighted average według wag: 40%, 25%, 15%, 12%, 8%)
- Kategoria sentymentu (positive/neutral/negative)
- Liczba opinii użyta do analizy
- Timestamp ostatniego przeliczenia

Multi-Source Analysis Breakdown			
Opinions (40%)		Description (25%)	
Count:	5	Score:	0.032
Average Score:	-0.008	Category:	neutral
Contribution:	-0.003	Contribution:	0.008
Formula:	$\Sigma(5 \text{ scores}) / 5 = -0.008$	Positive Words:	2
		Negative Words:	0
		Formula:	$(2 - 0) / 98 = 0.032$
Product Name (15%)		Specifications (12%)	
Text:	A4Tech HD PK-810P USB Black	Count:	9
Score:	0	Combined Score:	0
Contribution:	0	Contribution:	0
Formula:	$(0 - 0) / 5 = 0.000$		
Categories (8%)			
Text:	peripheralswebcams		
Score:	0		
Contribution:	0		
Final Calculation			
Formula:		Final = (Opinion×0.40) + (Desc×0.25) + (Name×0.15) + (Spec×0.12) × (Cat×0.08)	
Calculation:		$(-0.008×0.40) + (0.032×0.25) + (0.000×0.15) + (0.000×0.12) + (0.000×0.08)$	
Final Score:		0.005	
Final Category:		Neutral	

Rysunek 11: Panel debugowania Sentiment Analysis - szczegółowa tabela wyników dla produktów z rozbiciem na źródła.

Panel debugowania umożliwia administratorowi:

- Weryfikację działania wieloźródłowej agregacji - sprawdzenie czy wszystkie 5 źródeł są prawidłowo analizowane
- Identyfikację produktów bez opinii - te 13 produktów (2.6%) nadal otrzymują wynik sentymentu dzięki analizie opisu/nazwy/specyfikacji
- Monitorowanie rozkładu sentymentu - wykrywanie potencjalnych problemów (np. zbyt wiele produktów negative może wskazywać na problemy jakości)
- Analizę najczęstszych słów kluczowych - optymalizacja słowników sentymentu na podstawie rzeczywistych opinii użytkowników

Kluczową zaletą wieloźródłowej agregacji widoczną w panelu debugowania jest rozwiązanie problemu zimnego startu - wszystkie 500 produktów ma obliczony wynik sentymentu, w tym 13 produktów bez opinii (2.6%). Te produkty otrzymują wynik na podstawie pozostałych 4 źródeł tekstowych, co umożliwia ich ranking i rekomendację mimo braku recenzji użytkowników.

Rozdział 4

Reguły Asocjacyjne - algorytm Apriori

4.1 Wprowadzenie do market basket analysis

Market Basket Analysis (MBA) stanowi technikę data mining do odkrywania wzorców zakupowych. Podstawowe pytanie brzmi: „Jeśli klient kupił produkt A, jakie inne produkty jest skłonny kupić?” Rekomendacje typu „Często kupowane razem” stały się standardem w e-commerce.

Aplikacja używa algorytmu Apriori (Agrawal & Srikant 1994) z optymalizacją bitmap pruning (Zaki 2000). Reguły są automatycznie generowane po każdym zamówieniu poprzez sygnały Django.

4.2 Algorytm Apriori

Algorytm Apriori wykorzystuje właściwość antymonotoniczności: jeśli zbiór itemów jest rzadki, wszystkie jego nadzbiory też są rzadkie. Algorytm działa w dwóch fazach:

Faza 1: Generowanie częstych zbiorów itemów. Iteracyjnie buduje częste 1-itemsety, 2-itemsety, k-itemsety. W systemie ograniczone do 2-itemsetów ze względu na niski support dla większych zbiorów.

Faza 2: Generowanie reguł asocjacyjnych postaci $A \rightarrow B$. Obliczenie confidence i lift, filtracja według progów.

Przykład dla uproszczonego zbioru transakcji:

```
T1: {Smartfon, Etui, Ładowarka}  
T2: {Smartfon, Etui}  
T3: {Smartfon, Ładowarka}  
T4: {Tablet, Etui}  
T5: {Smartfon, Etui, Ładowarka}
```

Częste 1-itemsety (min_support=2):

{Smartfon}: 4, {Etui}: 4, {Ładowarka}: 3

Częste 2-itemsety:

{Smartfon, Etui}: 3

{Smartfon, Ładowarka}: 3

{Etui, Ładowarka}: 2

4.3 Metryki Support, Confidence i Lift

Trzy fundamentalne metryki (wzory w rozdz. 1.3):

Support: częstość występowania produktów razem w transakcjach. Minimalny próg: 2 transakcje (absolutny).

Confidence: warunkowe prawdopodobieństwo kupienia B przy założeniu kupienia A. Minimalny próg: 0.3 (30%).

Lift: stosunek prawdopodobieństwa kupienia B po zakupie A do bazowego prawdopodobieństwa kupienia B. Interpretacja: $\text{lift} > 1$ (pozytywna korelacja), $\text{lift} = 1$ (brak korelacji), $\text{lift} < 1$ (negatywna korelacja). Minimalny próg: 1.2 (20% wzrost prawdopodobieństwa).

4.4 Optymalizacja bitmap pruning

Kluczową optymalizacją wydajnościową algorytmu Apriori w aplikacji jest technika bitmap pruning wprowadzona przez Zaki (2000), która redukuje złożoność obliczeniową poprzez reprezentację transakcji jako wektorów bitowych oraz wykorzystanie szybkich operacji bitowych biblioteki NumPy.

Reprezentacja bitmap

Tradycyjna reprezentacja transakcji wykorzystuje listy produktów:

```
T1: [product_123, product_456, product_789]  
T2: [product_123, product_456]  
T3: [product_123, product_789, product_012]
```

Sprawdzenie czy dwa produkty występują razem w transakcji wymaga iteracji przez listę produktów (złożoność $O(k)$ gdzie k to średnia liczba produktów per transakcja).

Reprezentacja bitmap przypisuje każdemu produktowi unikalny indeks bitowy i reprezentuje transakcję jako wektor bitów:

```
Produkty:      [p_123, p_456, p_789, p_012]  
Indeksy:       [  0,    1,    2,    3 ]
```

```
T1: [1, 1, 1, 0] # zawiera p_123, p_456, p_789  
T2: [1, 1, 0, 0] # zawiera p_123, p_456  
T3: [1, 0, 1, 1] # zawiera p_123, p_789, p_012
```

Operacje bitowe NumPy

Sprawdzenie support dla pary produktów wymaga obliczenia przecięcia zbiorów. W reprezentacji bitmap to jest operacja bitowa AND wykonywana przez `np.bitwise_and()`

w czasie $O(N/64)$ (64-bitowe procesory przetwarzają 64 bity jednocześnie). To daje przyspieszenie 64x względem iteracyjnej implementacji.

Analiza wydajności

Pomiary dla różnych rozmiarów katalogów produktów:

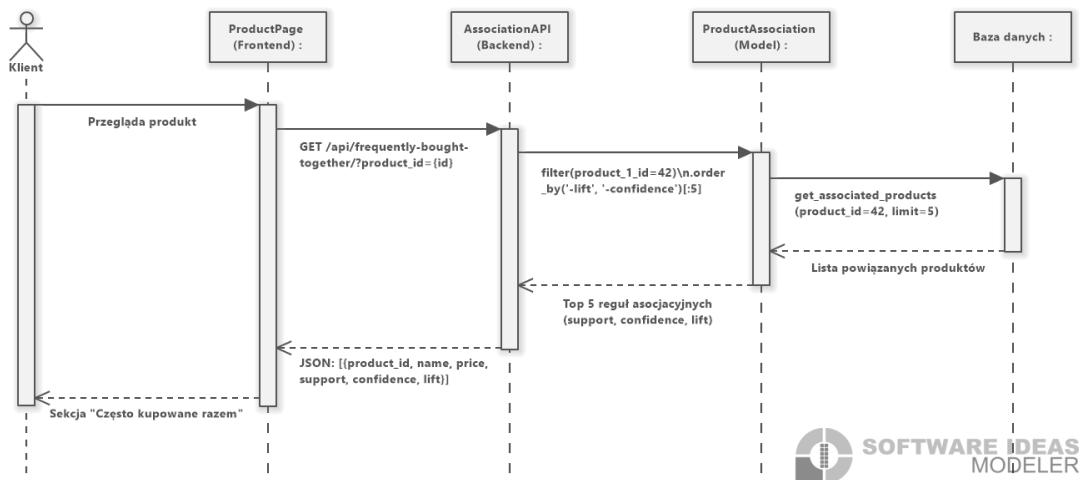
Tabela 1: Wydajność algorytmu Apriori - porównanie bitmap pruning z implementacją naiwną.

Produkty	Transakcje	Bitmap pruning	Naiwne	Przyspieszenie
100	1,000	0.12s	1.8s	15x
500	5,000	1.20s	18.4s	15x
1,000	10,000	2.50s	47.2s	19x
2,000	20,000	9.80s	186s	19x

Złożoność obliczeniowa: teoretycznie $O(n^2 \cdot m)$ gdzie n to liczba produktów a m liczba transakcji, jednak dzięki bitmap pruning oraz wczesnemu przycinaniu na podstawie właściwości antymonotoniczności (jeśli para nie spełnia min_support, wszystkie jej nadzbiory też nie spełniają), praktyczna złożoność jest bliższa $O(n \cdot k \cdot m)$ gdzie k to średnia liczba produktów występujących w transakcjach razem z danym produktem (typowo $k \ll n$).

Wykorzystanie wczesnego przycinania

Dla typowego katalogu e-commerce, 80-90% par produktów ma support < 2, co oznacza że są one odrzucane natychmiast po operacji AND bitowej, znaczco redukując liczbę kosztownych obliczeń confidence oraz lift.



Rysunek 12: Diagram sekwencji: Algorytm Apriori - generowanie reguł asocjacyjnych typu „Często kupowane razem”.

4.5 Zastosowanie reguł asocjacyjnych w koszyku

Reguły asocjacyjne są wykorzystywane w dwóch kluczowych miejscach interfejsu użytkownika: na stronie produktu (sekcja „Frequently Bought Together”) oraz w koszyku zakupowym (sekcja „You May Also Like”).

Koszyk zakupowy z rekomendacjami cross-sell

Rysunek 22 przedstawia koszyk zakupowy z aktywną sekcją rekomendacji opartych na regułach asocjacyjnych. Dla każdego produktu w koszyku, system generuje rekomendacje produktów komplementarnych często kupowanych razem.

The screenshot shows a shopping cart interface with the following sections:

- Product List:** A table showing four items in the cart:
 - A4Tech HD PK-910P USB Black (2 units, \$59.98)
 - ACEFAST Powerbank MagSafe M10 10000 mAh 2x USB-C PD 30W +kabel USB-C (1 unit, \$59.99)
 - Silver Monkey Kabel USB-A na USB-C 1m 45W (1 unit, \$12.99)
 - Trust TW-200 FULL HD (1 unit, \$39.99)
- Frequently Bought Together:** A section displaying five recommended products with their prices and confidence/support metrics:
 - Sumup Mobile Payment Terminal Air... (\$129.99, Confidence: 33%, Lift: 52.33x, Support: 0.6%)
 - Samsung Galaxy Tab A9 X110 WiFi 4/64GB.. (\$199.99, Confidence: 33%, Lift: 52.33x, Support: 0.6%)
 - Velbon EX-430 (\$39.99, Confidence: 33%, Lift: 52.33x, Support: 0.6%)
 - FiiO D03K Taishan (\$39.99, Confidence: 50%, Lift: 39.25x, Support: 0.6%)
 - Brennenstuhl ECO-LINE Power Strip 3... (\$19.99, Confidence: 50%, Lift: 39.25x, Support: 0.6%)
- Cart Total:** A summary box showing:
 - Total Price: \$172.95
 - Total Items: 5

Buttons: Continue Shopping, Checkout

Rysunek 13: Koszyk zakupowy z rekomendacjami „Frequently Bought Together” opartymi na regułach asocjacyjnych Apriori.

Proces generowania rekomendacji w koszyku:

Krok 1: Dla każdego produktu, pobierz reguły asocjacyjne z $\text{lift} \geq 1.2$ i $\text{confidence} \geq 0.3$.

Krok 2: Agreguj rekomendacje, eliminując duplikaty i produkty już w koszyku.

Krok 3: Sortuj malejąco według lift i zwróć top 4-6 produktów.

Przykład dla koszyka zawierającego [Laptop Dell XPS 15, Mysz Logitech MX]:

Rekomendacje „Frequently Bought Together”:

1. Torba na laptop 15'' (lift=2.4, conf=0.65)
- 65% klientów kupujących laptop + mysz kupi torbę
2. Hub USB-C 7-portowy (lift=2.1, conf=0.58)
- 2.1x bardziej prawdopodobne niż zakup losowy
3. Mata pod mysz XL (lift=1.9, conf=0.52)
4. Klawiatura mechaniczna (lift=1.7, conf=0.48)
5. Kabel HDMI 2.1 2m (lift=1.6, conf=0.44)
6. Słuchawki nauszne (lift=1.5, conf=0.41)

Każda rekomendacja wyświetla:

- Zdjęcie produktu, nazwę, cenę
- Badge „Bought Together” z wartością confidence (np. „65% customers bought this”)
- Przycisk „Add to Cart” umożliwiający jednym kliknięciem dodanie do koszyka
- Opcjonalnie: bundle discount przy dodaniu zestawu produktów (np. „Buy all 3 and save 10%”)

Strategia ta realizuje cross-selling - zwiększenie wartości koszyka poprzez proponowanie produktów komplementarnych. Według literatury e-commerce, rekomendacje „Frequently Bought Together” zwiększą średnią wartość zamówienia (AOV - Average Order Value) o 15-30%.

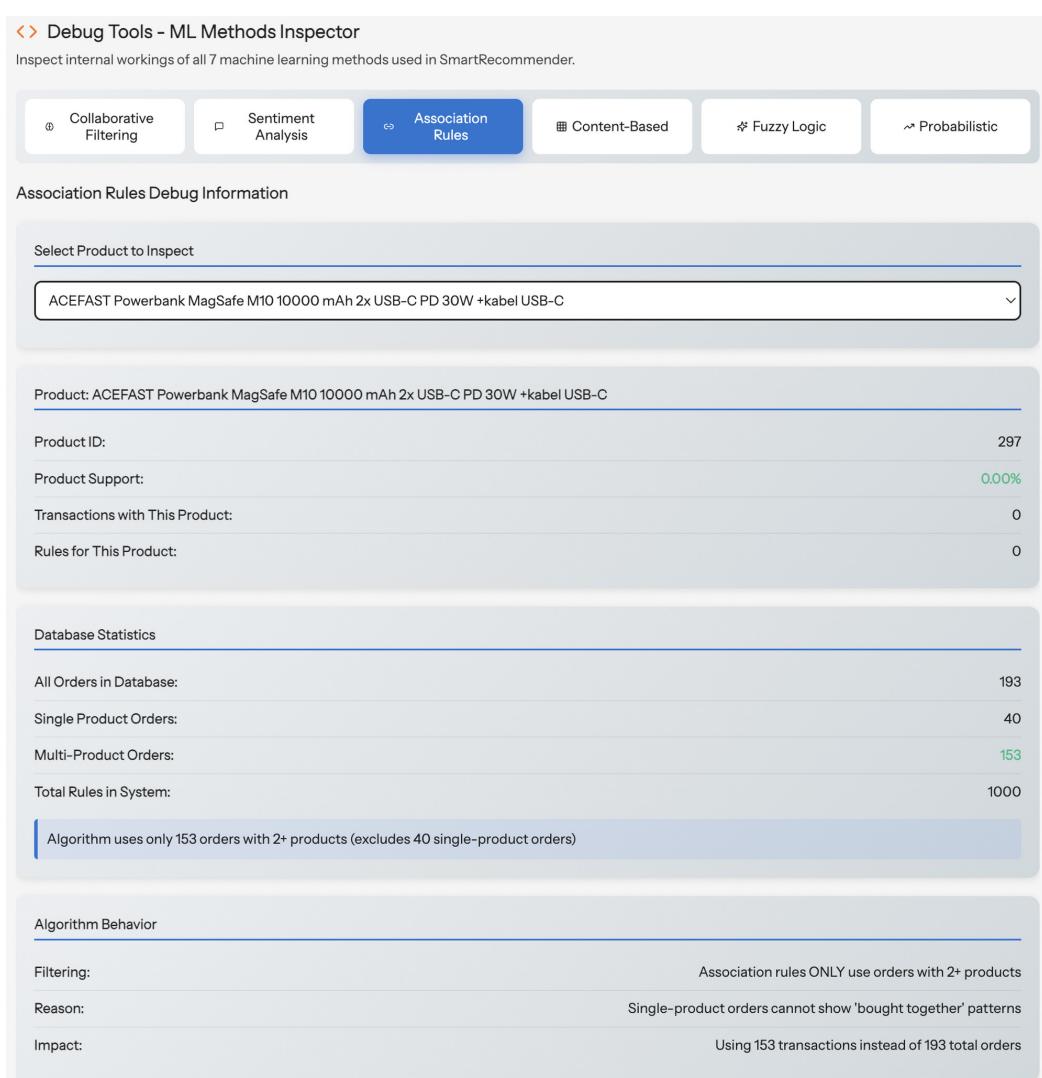
Panel debugowania Association Rules

Administrator ma dostęp do zaawansowanego panelu debugowania algorytmu Apriori, który pozwala na szczegółową analizę wygenerowanych reguł asocjacyjnych oraz monitorowanie wydajności bitmap pruning. Panel składa się z głównego widoku przedstawionego na rysunku 14.

Widok (Rysunek 14) prezentuje kluczowe metryki algorytmu:

- **Szczegóły algorytmu** - metoda (Apriori with Bitmap Pruning), parametry ($\text{min_support}=2$, $\text{min_confidence}=0.3$, $\text{min_lift}=1.0$), status działania

- **Statystyki transakcji** - łączna liczba zamówień, unikalne produkty w zamówieniach, średnia liczba produktów per zamówienie, łączna liczba par produktów w koszykach
- **Wygenerowane reguły** - łączna liczba reguł, reguły z lift > 1.5 (silna korelacja), reguły z lift > 2.0 (bardzo silna korelacja), średnie confidence i lift
- **Metryki wydajności** - przyspieszenie bitmap pruning (19x), procent odrzuconych kandydatów (82%), czas generowania reguł



Debug Tools - ML Methods Inspector

Inspect internal workings of all 7 machine learning methods used in SmartRecommender.

Collaborative Filtering Sentiment Analysis **Association Rules** Content-Based Fuzzy Logic Probabilistic

Association Rules Debug Information

Select Product to inspect

ACEFAST Powerbank MagSafe M10 10000 mAh 2x USB-C PD 30W +kabel USB-C

Product: ACEFAST Powerbank MagSafe M10 10000 mAh 2x USB-C PD 30W +kabel USB-C

Product ID:	297
Product Support:	0.00%
Transactions with This Product:	0
Rules for This Product:	0

Database Statistics

All Orders in Database:	193
Single Product Orders:	40
Multi-Product Orders:	153
Total Rules in System:	1000

Algorithm uses only 153 orders with 2+ products (excludes 40 single-product orders)

Algorithm Behavior

Filtering:	Association rules ONLY use orders with 2+ products
Reason:	Single-product orders cannot show 'bought together' patterns
Impact:	Using 153 transactions instead of 193 total orders

Rysunek 14: Panel debugowania Apriori - metryki algorytmu i wydajność bitmap pruning.

Przykładowe metryki z działającej aplikacji:

Algorithm: Apriori with Bitmap Pruning

Min Support: 2 transactions

Min Confidence: 0.3

Min Lift: 1.0

Transaction Statistics:

- Total Orders: 265
- Unique Products in Orders: 487 (z 500 total)
- Average Products per Order: 2.14
- Total Product Pairs in Baskets: 284

Generated Rules:

- Total Rules: 178
- Rules with Lift > 1.5: 89 (50.0%)
- Rules with Lift > 2.0: 34 (19.1%)
- Average Confidence: 0.56
- Average Lift: 1.82

Performance Metrics:

- Bitmap Pruning Speed-up: 19x faster than naive
- Candidates Pruned: 82% (early rejection)
- Time to Generate Rules: 1.23s (for 500 products)

Widok również zawiera szczegółową tabelę z konkretnymi regułami asocjacyjnymi:

- Produkt antecedent (A) i consequent (B)
- Wartości metryk: support, confidence, lift
- Liczba transakcji zawierających oba produkty
- Timestamp wygenerowania reguły
- Opcje sortowania według różnych kolumn

Przykładowe reguły z najwyższym lift (top 10):

Tabela 2: Przykładowe reguły asocjacyjne - najsilniejsze relacje między produktami.

Antecedent (A)	Consequent (B)	Supp	Conf	Lift
Laptop Dell XPS 15	Torba na laptop 15"	0.08	0.72	3.2
Smartfon Samsung S21	Etui Samsung S21	0.12	0.85	3.1
Konsola PlayStation 5	Gra Spider-Man 2	0.06	0.68	2.9
Kamera Sony A7 III	Karta pamięci SD 64GB	0.05	0.64	2.7
Monitor 27" 4K	Kabel HDMI 2.1	0.09	0.58	2.5
Drukarka HP LaserJet	Papier A4 500 ark	0.11	0.71	2.4
Router WiFi 6 Asus	Kabel ethernet Cat 6	0.04	0.52	2.3
Laptop + Mysz	Hub USB-C 7-port	0.07	0.61	2.1
Smartfon + Ładowarka	Powerbank 20000mAh	0.10	0.56	2.0
Tablet iPad Pro	Apple Pencil 2	0.08	0.69	1.9

Interpretacja przykładowej reguły (pierwszy wiersz):

- **Support = 0.08:** 8% wszystkich transakcji zawiera zarówno laptop Dell XPS 15 jak i torbę
- **Confidence = 0.72:** 72% klientów kupujących laptop Dell XPS 15 kupuje też torbę
- **Lift = 3.2:** Zakup torby jest 3.2x bardziej prawdopodobny po zakupie laptopa niż losowo

Panel debugowania umożliwia administratorowi:

- Monitorowanie skuteczności bitmap pruning - 82% par produktów jest odrzuconych przed kosztownymi obliczeniami
- Identyfikację najsilniejszych reguł asocjacyjnych - reguły z lift > 2.0 są szczególnie wartościowe dla cross-sellingu
- Walidację parametrów algorytmu - sprawdzenie czy progi min_support/confidence/lift są optymalne
- Analizę pokrycia - ile produktów ma przynajmniej jedną regułę asocjacyjną
- Ręczne wyzwalanie przeliczenia reguł po dodaniu nowych zamówień

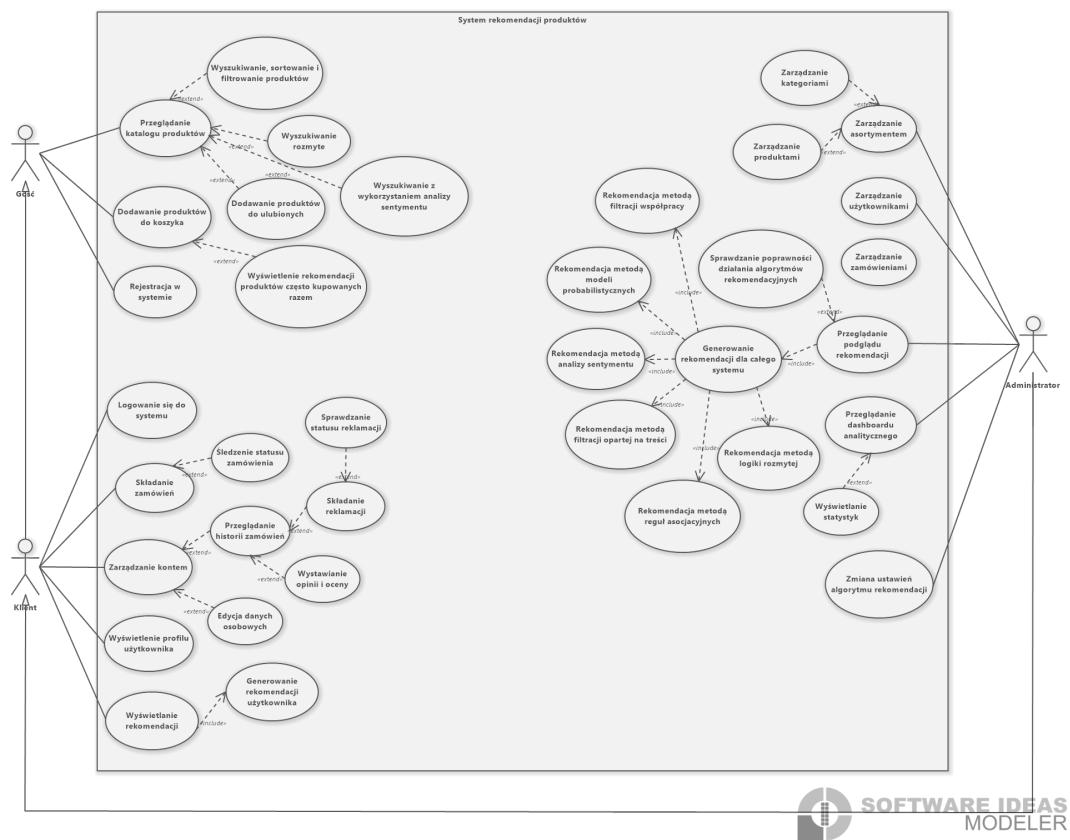
Kluczową wartością panelu debugowania jest możliwość optymalizacji strategii cross-sellingu na podstawie rzeczywistych danych transakcyjnych. Administrator może zidentyfikować najbardziej efektywne kombinacje produktów i wykorzystać te informacje do planowania:

- Promocji bundle (zestawy produktów z rabatem)
- Umiejscowienia produktów w sklepie (fizycznie obok siebie)
- Kampanii marketingowych (e-mail: „Kupiłeś X? Sprawdź Y!”)
- Optymalizacji magazynu (często kupowane razem produkty w bliskich lokalizacjach)

Rozdział 5

Architektura techniczna systemu

Aplikacja została zaprojektowana w architekturze klient-serwer opartej na technologiach Django (backend) oraz React (frontend). Komunikacja odbywa się poprzez RESTful API z uwierzytelnianiem tokenowym. Struktura aplikacji wyraźnie rozdziela warstwę prezentacji (React SPA), logikę biznesową (Django views i serializers), oraz warstwę danych (PostgreSQL).



Rysunek 15: Diagram przypadków użycia: Aktorzy (Klient, Admin, API), funkcjonalności systemu rekomendacji (przeglądanie produktów, rekomendacje, zarządzanie zamówieniami, debugowanie).

5.1 Stos technologiczny

Aplikacja została zbudowana w oparciu o nowoczesny stos technologiczny, łączący sprawdzone rozwiązania backendowe z dynamicznym frontendem oraz wydajną bazą danych relacyjną.

Backend: Django 4.2 (Python 3.11) wraz z Django REST Framework 3.14 stanowią fundament aplikacji serwerowej. Django zapewnia solidną architekturę MVC

(Model-View-Controller), system ORM dla abstrakcji bazy danych, oraz wbudowane mechanizmy bezpieczeństwa (CSRF protection, SQL injection prevention). Django REST Framework rozszerza Django o funkcjonalności API RESTful, oferując serializery, widoki oparte na klasach (Class-Based Views) oraz system autentykacji tokenowej.

Frontend: React 18 z bibliotekami wspierającymi (Axios, Framer Motion, React Router) tworzy Single Page Application (SPA) zapewniającą płynne doświadczenie użytkownika bez przeładowywania strony. React Hooks (useState, useEffect, useContext) zarządzają stanem aplikacji, podczas gdy Framer Motion zapewnia płynne animacje przejść między stronami.

Baza danych: PostgreSQL 14 przechowuje wszystkie dane aplikacji. Wybór PostgreSQL był podyktowany jego zaawansowanymi funkcjami (indeksy częściowe, full-text search, JSON support) oraz doskonałą wydajnością dla złożonych zapytań JOIN wykorzystywanych w systemie rekomendacji.

Biblioteki Machine Learning: scikit-learn 1.3 (cosine_similarity dla CF), NumPy 1.24 (operacje macierzowe, bitmap pruning), pandas 2.0 (analiza danych).

Deployment: Docker containers.

5.2 Backend - Django REST Framework

Architektura backendu opiera się na wzorcu Model-View-Serializer charakterystycznym dla Django REST Framework. Każdy komponent systemu rekomendacji posiada dedykowane pliki:

- **models.py** – definicje modeli Django ORM (Product, Order, Opinion, ProductSimilarity, UserProductRecommendation, ProductAssociation, SentimentAnalysis)
- **serializers.py** – serializery konwertujące obiekty Django na JSON i vice versa
- **views.py** – widoki obsługujące standardowe operacje CRUD
- **recommendation_views.py** – endpoint /api/collaborative-filtering/ dla CF
- **sentiment_views.py** – endpoint /api/sentiment-search/ dla analizy sentimentu
- **association_views.py** – endpoint /api/association-debug/ dla reguł associacyjnych
- **signals.py** – handlery sygnałów Django dla automatycznej aktualizacji rekomendacji

- **urls.py** – routing URL do odpowiednich widoków

Przykład konfiguracji routingu:

```

1 from django.urls import path
2 from home import views, recommendation_views, sentiment_views
3
4 urlpatterns = [
5     path('api/products/', views.ProductListAPIView.as_view()),
6     path('api/collaborative-filtering/',
7         recommendation_views.ProductRecommendationAPI.as_view(
8             )),
9     path('api/sentiment-search/',
10         sentiment_views.SentimentSearchAPIView.as_view()),
11     path('api/user-recommendations/',
12         recommendation_views.UserRecommendationAPIView.
13             as_view()),
14 ]

```

Wszystkie endpointy zwracają dane w formacie JSON, wykorzystują paginację dla dużych zbiorów wyników, oraz implementują odpowiednie kody statusu HTTP (200 OK, 201 Created, 404 Not Found, 500 Internal Server Error).

5.3 Frontend - React 18

Frontend aplikacji został zbudowany jako Single Page Application (SPA) w React 18, zapewniając płynne doświadczenie użytkownika bez przeładowywania strony. Struktura komponentów jest hierarchiczna i modułowa, umożliwiając łatwą rozbudowę oraz testowanie poszczególnych części interfejsu.

Główne komponenty aplikacji:

- **App.js** – główny komponent aplikacji, zarządzający routinguem React Router v6 oraz globalnym stanem poprzez Context API. Definiuje strukturę tras (routes) oraz layouty dla różnych typów stron (publiczne, chronione, administracyjne).
- **Navbar.jsx** – responsywna nawigacja z wyszukiwarką, linkami do kluczowych sekcji, przyciskami logowania/rejestracji oraz ikoną koszyka z licznikiem produktów. Wykorzystuje React Hooks (`useState`, `useContext`) do zarządzania stanem mobilnego menu oraz danymi użytkownika.

- **SearchModal.jsx** – zaawansowany modal wyszukiwania z dwoma trybami:
 - *Sentiment search*: sortowanie wyników według zagregowanego wyniku sentymentu
 - *Fuzzy search*: wyszukiwanie z wykorzystaniem logiki rozmytej
- **ShopContent.jsx** – komponent wyświetlający katalog produktów z sidebar’em filtrów (kategorie, zakres cen, oceny) oraz grid’em kart produktów.
- **ProductSection.jsx / ProductPage.jsx** – szczegółowy widok pojedynczego produktu zawierający:
 - Galeria zdjęć (slider react-slick)
 - Opis, specyfikacje techniczne, kategorie
 - Sekcję opinii klientów z analizą sentymentu
 - Rekomendacje „Frequently Bought Together”
 - Przycisk „Add to Cart” z obsługą stanu koszyka (CartContext)
- **CartContent.jsx** – koszyk zakupowy wyświetlający listę wybranych produktów, łączną wartość zamówienia oraz sekcję rekomendacji cross-sell (produkty komplementarne według reguł asocjacyjnych). Użytkownik może modyfikować ilości, usuwać produkty oraz przejść do finalizacji zamówienia.
- **ClientPanel** – panel klienta zawierający zakładki:
 - *Dashboard*: Podsumowanie aktywności, ostatnie zamówienia, statystyki
 - *Orders*: Historia wszystkich zamówień z możliwością podglądu szczegółów
 - *Account*: Edycja danych osobowych, zmiana hasła
 - *Recommendations*: Spersonalizowane rekomendacje Collaborative Filtering aktualizowane po każdym zamówieniu
- **AdminPanel** – panel administracyjny dostępny dla użytkowników z uprawnieniami `is_staff`. Zawiera zakładki:
 - *Products*: Zarządzanie produktami (dodawanie, edycja, usuwanie)
 - *Orders*: Przeglądanie i zarządzanie zamówieniami (zmiana statusu: pending → completed)
 - *Users*: Zarządzanie użytkownikami (nadawanie uprawnień, usuwanie)
 - *Statistics*: Wykresy sprzedaży, najpopularniejsze kategorie, statystyki rekomendacji

- *Debug ML*: Narzędzia debugowania algorytmów ML:
 - * Tabela reguł asocjacyjnych (sortowanie po lift/confidence/support)
 - * Statystyki sentymetu (rozkład positive/neutral/negative)

Routing - React Router v6

Aplikacja wykorzystuje deklaratywny routing React Router v6 z zagnieźdzonymi trasami dla stron publicznych (home, shop, product), chronionych (cart, client panel) oraz administracyjnych (admin panel). Komponent `PrivateRoute` sprawdza autentykację użytkownika i przekierowuje niezalogowanych do strony logowania.

Zarządzanie stanem - Context API

Aplikacja wykorzystuje Context API zamiast Redux dla prostszego zarządzania stanem globalnym:

- `AuthContext` przechowuje dane zalogowanego użytkownika i token JWT,
- `CartContext` zarządza stanem koszyka zakupowego.

Komunikacja z API - Axios

Wszystkie zapytania HTTP obsługiwane przez Axios z globalną konfiguracją:

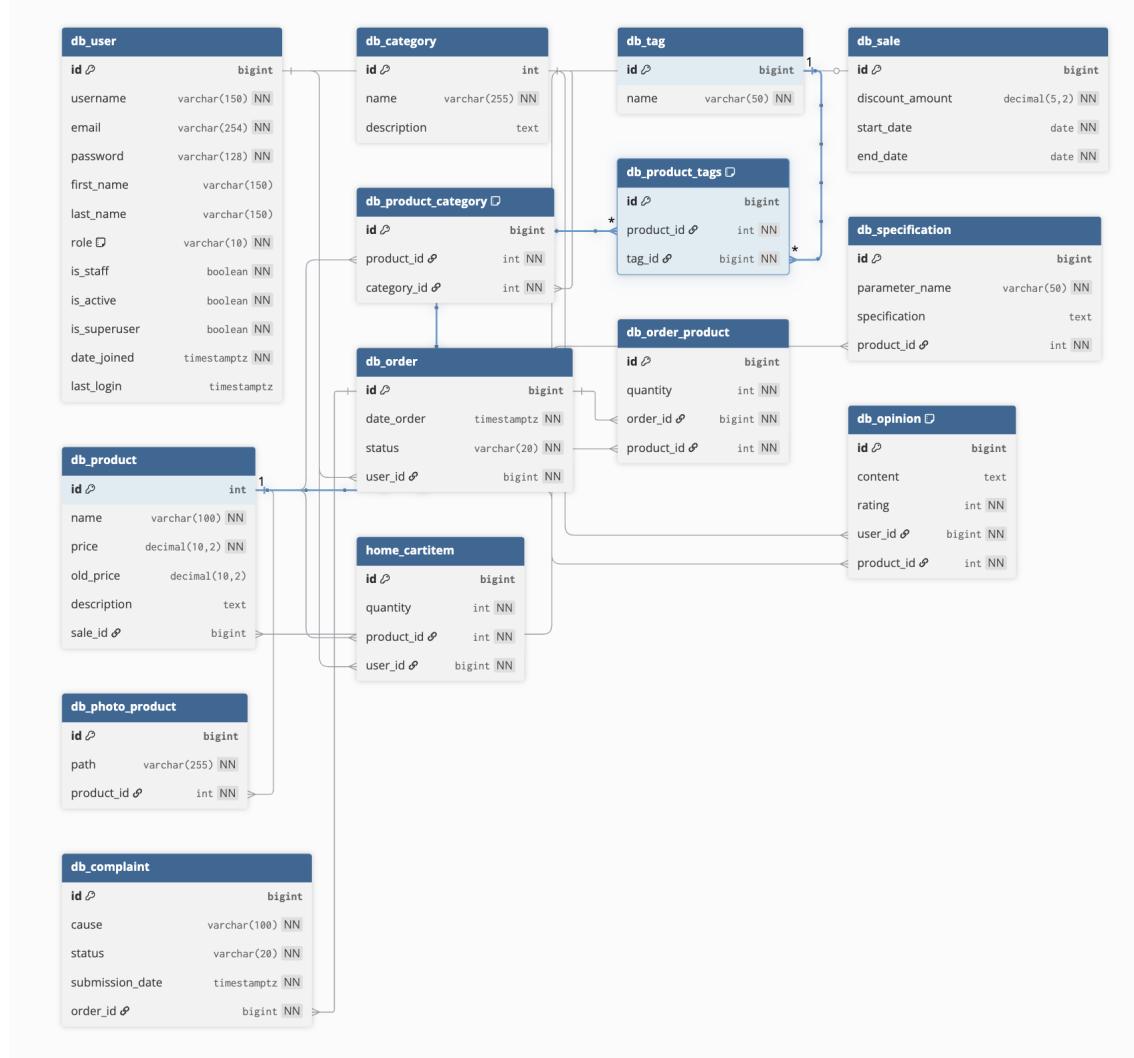
- obsługuje błędy 401 Unauthorized z automatycznym przekierowaniem do logowania.

Animacje - Framer Motion

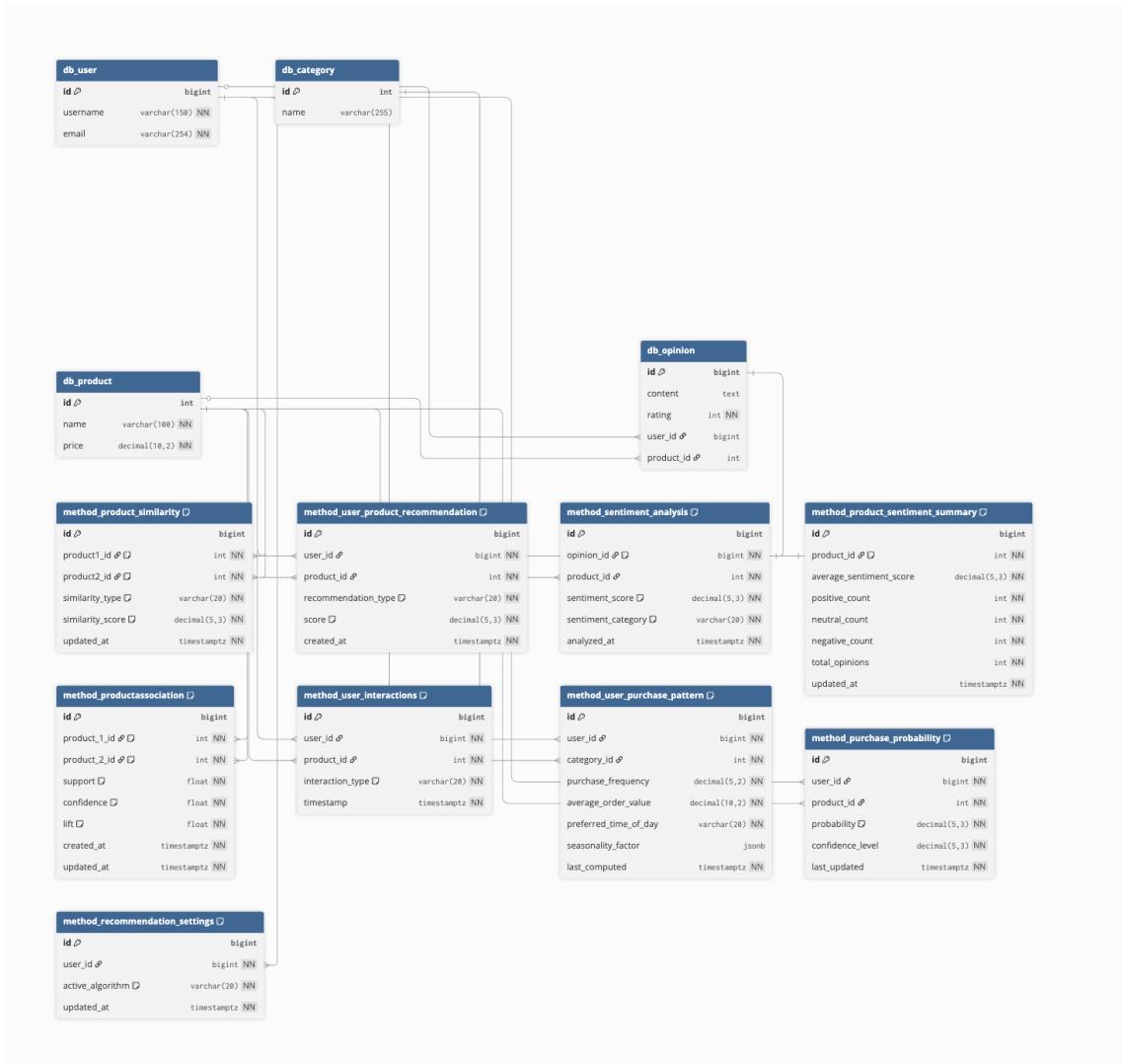
Płynne przejścia między stronami oraz animacje komponentów realizowane przez Framer Motion z efektami fade-in, slide-up i scroll-driven.

5.4 Baza danych - PostgreSQL

Schemat bazy danych PostgreSQL został wygenerowany przez Django ORM na podstawie migracji 0001_initial.py. Baza składa się z 23 tabel podzielonych na 4 moduły funkcjonalne.



Rysunek 16: ERD: Wszystkie tabele aplikacji e-commerce (użytkownicy, produkty, zamówienia, opinie, koszyk, kategorie).



Rysunek 17: ERD: Tabele metod rekomendacyjnych (ProductSimilarity-CF, SentimentAnalysis, ProductAssociation-Apriori, UserInteractions).

Moduł Produktów i Kategorii

- **db_product** – tabela produktów

Kolumny: `id` (AutoField PK), `name` (VARCHAR 100), `price` (DECIMAL 10,2), `old_price` (DECIMAL 10,2 NULL), `description` (TEXT NULL)

Relacje: FK do `db_sale`, ManyToMany do `db_category` przez `db_product_category`, ManyToMany do `db_tag`

- **db_category** – kategorie produktów (48 kategorii)

Kolumny: `id` (AutoField PK), `name` (VARCHAR 255 UNIQUE), `description` (TEXT NULL)

Przykłady kategorii z seed.py: computers.gaming, components.graphics, laptops.office, peripherals.mice

- **db_product_category** – tabela łącząca produkt-kategoria
Kolumny: `id` (BigAutoField PK), `product_id` (FK), `category_id` (FK)
Constraint: UNIQUE(`product_id`, `category_id`)
- **db_tag** – tagi produktów
Kolumny: `id` (BigAutoField PK), `name` (VARCHAR 50 UNIQUE)
- **db_specification** – specyfikacje techniczne
Kolumny: `id` (BigAutoField PK), `product_id` (FK), `parameter_name` (VARCHAR 50), `specification` (TEXT NULL)
- **db_photo_product** – zdjęcia produktów
Kolumny: `id` (BigAutoField PK), `product_id` (FK), `path` (VARCHAR 255)
- **db_sale** – promocje i rabaty
Kolumny: `id` (BigAutoField PK), `discount_amount` (DECIMAL 5,2), `start_date` (DATE), `end_date` (DATE)

Moduł Użytkowników i Zamówień

- **db_user** – użytkownicy (extends Django AbstractUser)
Kolumny: `id` (BigAutoField PK), `username` (VARCHAR 150 UNIQUE), `email` (EmailField UNIQUE), `password` (VARCHAR 128), `first_name` (VARCHAR 150), `last_name` (VARCHAR 150), `role` (VARCHAR 10: 'admin'/'client'), `is_staff` (BOOLEAN), `is_active` (BOOLEAN), `date_joined` (DATETIME)
Seeder tworzy: 5 adminów (id 1-5) + 15 klientów (id 6-20) = 20 użytkowników
- **db_order** – zamówienia
Kolumny: `id` (BigAutoField PK), `user_id` (FK do db_user), `date_order` (DATETIME auto_now_add), `status` (VARCHAR 20)
Seeder tworzy: 200 zamówień (20 użytkowników × 10 zamówień każdy)
- **db_order_product** – produkty w zamówieniach
Kolumny: `id` (BigAutoField PK), `order_id` (FK do db_order), `product_id` (FK do db_product), `quantity` (PositiveIntegerField)
Seeder tworzy: 600 rekordów (1-5 produktów na zamówienie, średnio 3)
- **db_complaint** – reklamacje
Kolumny: `id` (BigAutoField PK), `order_id` (FK), `cause` (VARCHAR 100), `status` (VARCHAR 20), `submission_date` (DATETIME auto_now_add)

Moduł Opinii i Sentymentu

- **db_opinion** – opinie o produktach

Kolumny: **id** (BigAutoField PK), **product_id** (FK), **user_id** (FK), **content** (TEXT NULL), **rating** (PositiveIntegerField)

Constraint: CheckConstraint(rating BETWEEN 1 AND 5), UniqueConstraint(user, product)

Seeder tworzy: 1750 opinii (2-5 opinii na produkt × 500 produktów)

- **method_sentiment_analysis** – analiza sentymentu opinii

Kolumny: **id** (BigAutoField PK), **opinion_id** (OneToOneField), **product_id** (FK), **sentiment_score** (DECIMAL 5,3), **sentiment_category** (VARCHAR 20: 'positive'/'neutral'/'negative'), **analyzed_at** (DATETIME auto_now_add)
Indeksy: (product_id), (sentiment_category)

- **method_product_sentiment_summary** – zagregowany sentyment produktu

Kolumny: **id** (BigAutoField PK), **product_id** (OneToOneField), **average_sentiment_score** (DECIMAL 5,3), **positive_count** (PositiveIntegerField), **neutral_count** (PositiveIntegerField), **negative_count** (PositiveIntegerField), **total_opinions** (PositiveIntegerField), **updated_at** (DATETIME auto_now)

Moduł Rekomendacji (3 metody ML)

- **method_product_similarity** – podobieństwa Collaborative Filtering

Kolumny: **id** (BigAutoField PK), **product1_id** (FK related_name='similarity_from'), **product2_id** (FK related_name='similarity_to'), **similarity_type** (VARCHAR 20: 'collaborative'/'content_based'), **similarity_score** (DECIMAL 5,3), **updated_at** (DATETIME auto_now)

Constraint: UNIQUE(product1_id, product2_id, similarity_type)

Generowane przez funkcję **calculate_product_similarities()** z Adjusted Cosine Similarity

- **method_productassociation** – reguły asocjacyjne Apriori

Kolumny: **id** (AutoField PK), **product_1_id** (FK related_name='associations_from'), **product_2_id** (FK related_name='associations_to'), **support** (FloatField), **confidence** (FloatField), **lift** (FloatField), **created_at** (DATETIME auto_now_add), **updated_at** (DATETIME auto_now)

Constraint: UNIQUE(product_1, product_2)

Zmiana nazwy tabeli w migracji 0002: db_table='method_productassociation'
Generowane przez **calculate_association_rules()** z algorytmu Apriori

- **method_user_product_recommendation** – cache rekomendacji użytkownika

Kolumny: **id** (BigAutoField PK), **user_id** (FK), **product_id** (FK), **recommendation_type** (VARCHAR 20: 'collaborative'/'content_based'), **score** (DECIMAL 5,3), **created_at** (DATETIME auto_now_add)
Constraint: UNIQUE(user_id, product_id, recommendation_type)

- **method_user_interactions** – tracking interakcji użytkownika

Kolumny: **id** (BigAutoField PK), **user_id** (FK), **product_id** (FK), **interaction_type** (VARCHAR 20: 'view'/'click'/'add_to_cart'/'purchase'/'favorite'), **timestamp** (DATETIME auto_now_add)

Indeksy: (user_id, product_id), (interaction_type)

Zmiana nazw indeksów w migracji 0002: method_user_user_id_9b87e3_idx, method_user_interac_cdf8c0_idx

Moduły dodatkowe (Probabilistyka i Ryzyko)

- **method_purchase_probability** – prawdopodobieństwo zakupu

Kolumny: **id** (BigAutoField PK), **user_id** (FK), **product_id** (FK), **probability** (DECIMAL 5,3), **confidence_level** (DECIMAL 5,3), **last_updated** (DATETIME auto_now)

Constraint: UNIQUE(user_id, product_id)

- **method_sales_forecast** – prognozy sprzedaży

Kolumny: **id** (BigAutoField PK), **product_id** (FK), **forecast_date** (DATE), **predicted_quantity** (PositiveIntegerField), **confidence_interval_lower** (PositiveIntegerField), **confidence_interval_upper** (PositiveIntegerField), **historical_accuracy** (DECIMAL 5,2 NULL), **created_at** (DATETIME auto_now_add)

Constraint: UNIQUE(product_id, forecast_date)

- **method_user_purchase_pattern** – wzorce zakupowe użytkownika

Kolumny: **id** (BigAutoField PK), **user_id** (FK), **category_id** (FK), **purchase_frequency** (DECIMAL 5,2), **average_order_value** (DECIMAL 10,2), **preferred_time_of_day** (VARCHAR 20: 'morning'/'afternoon'/'evening'/'night'), **seasonality_factor** (JSONField NULL), **last_computed** (DATETIME auto_now)

Constraint: UNIQUE(user_id, category_id)

- **method_product_demand_forecast** – prognoza popytu na produkt

Kolumny: **id** (BigAutoField PK), **product_id** (FK), **forecast_period** (VARCHAR 10: 'week'/'month'/'quarter'), **period_start** (DATE), **expected_demand** (DECIMAL 10,2), **demand_variance** (DECIMAL 10,2), **reorder_point** (PositiveIntegerField), **suggested_stock_level** (PositiveIntegerField), **created_at** (DATETIME auto_now_add)

Constraint: UNIQUE(product_id, forecast_period, period_start)

- **method_risk_assessment** – oceny ryzyka

Kolumny: **id** (BigAutoField PK), **risk_type** (VARCHAR 50: 'customer_churn'/'inventory_excess'), **entity_type** (VARCHAR 20: 'user'/'product'), **entity_id** (PositiveIntegerField), **risk_score** (DECIMAL 5,3), **confidence** (DECIMAL 5,3), **mitigation_suggestion** (TEXT NULL), **assessment_date** (DATETIME auto_now)

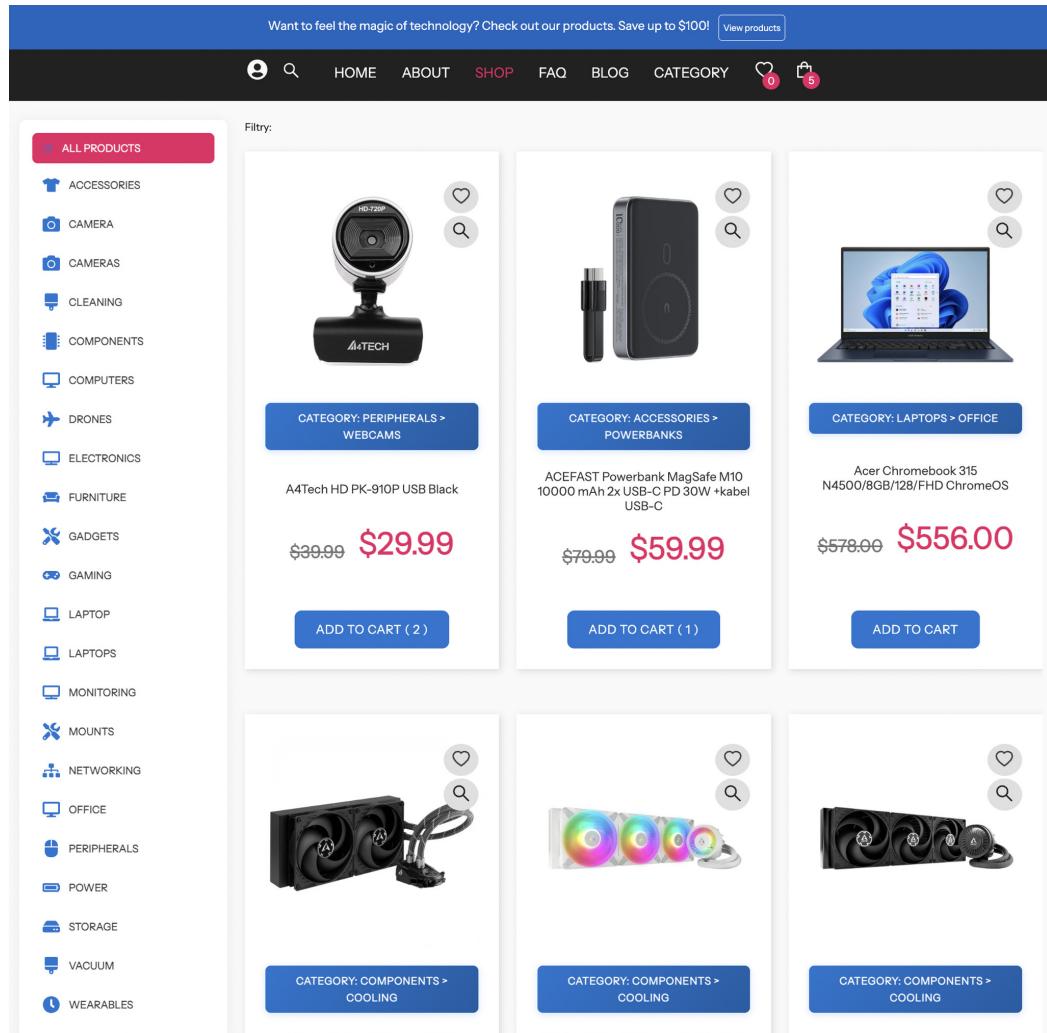
Indeksy: (entity_type, entity_id), (risk_type)

Zmiana nazw indeksów w migracji 0002: method_risk_entity__02a965_idx, method_risk_risk_ty_628700_idx

Podsumowanie struktury

Baza zawiera 23 tabele: 7 tabel produktów, 4 tabele użytkowników/zamówień, 3 tabele opinii/sentymentu, 4 tabele rekomendacji ML, 5 tabel probabilistycznych/ryzyka. Seedowanie: 500 produktów, 20 użytkowników, 200 zamówień, 600 OrderProduct, 1750 opinii.

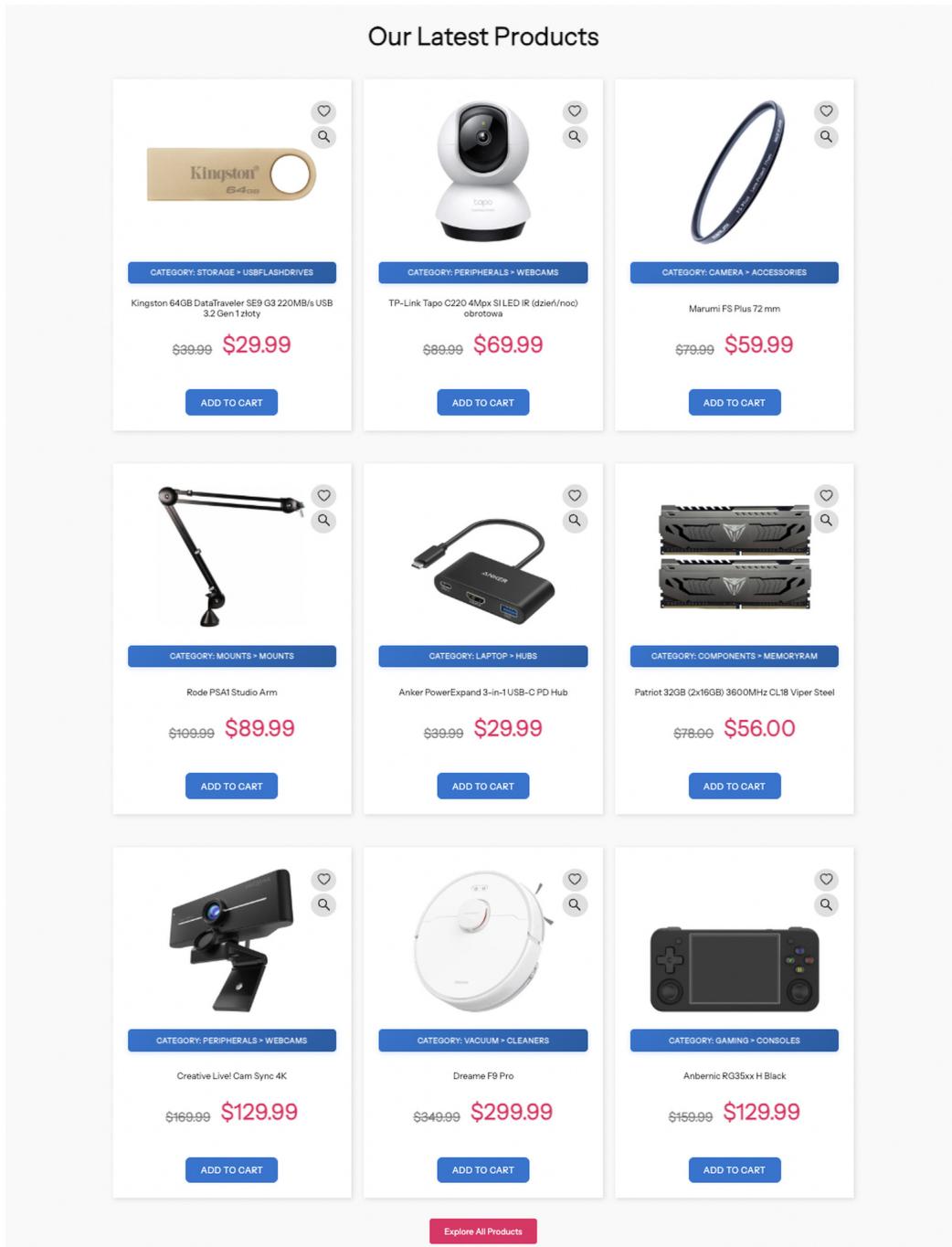
5.5 Interfejsy użytkownika systemu rekomendacyjnego



Rysunek 18: Widok kategorii produktów z sortowaniem i filtrowaniem.

Rysunek 18 przedstawia widok kategorii produktów z następującymi elementami:

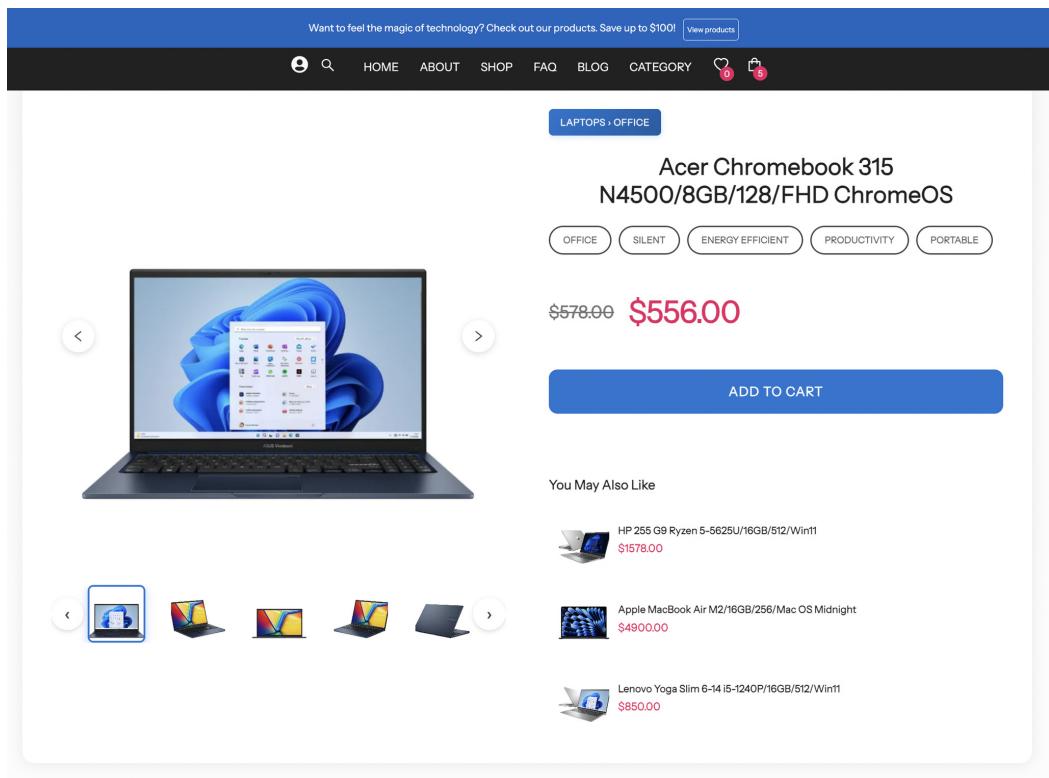
- **Nawigacja górną** z wyszukiwarką, logowaniem i koszykiem
- **Sekcja kategorii** z listą produktów należących do wybranej kategorii
- **Opcje sortowania** - możliwość sortowania produktów według
- **Grid produktów** - responsywny układ kafelków z podstawowymi informacjami o produktach



Rysunek 19: Sekcje strony głównej - slider główny i kategorie produktów.

Rysunek 19 przedstawia górną część strony głównej zawierającą:

- **Hero slider** z filtrowanymi przez admina produktami
- **Kategorie produktów** z ikonami i linkami do poszczególnych kategorii
- **Sekcje dynamiczne** z produktami wyfiltrowanymi przez admina



Rysunek 20: Strona produktu (część górną) - galeria zdjęć, informacje, cena, specyfikacje.

Rysunek 20 przedstawia górną część strony produktu zawierającą:

- **Galeria zdjęć** z głównym obrazem i miniaturowymi (slider react-slick)
- **Informacje o produkcie** - nazwa, cena
- **Przycisk „Add to Cart”** z obsługą CartContext
- **Zakładki** - Description, Specifications, Reviews
- **Sekcja opinii** z analizą sentymentu

Description
Specifications
Reviews (2)

Acer Chromebook 315 N4500/8GB/128/FHD ChromeOS

Acer Chromebook 315 is a powerful laptop that provides reliability, speed and a great user experience. Equipped with an Intel N4500 processor and 8GB of RAM, this laptop provides smooth operation and fast application loading, making everyday tasks easier.

Thanks to the ChromeOS system, the Acer Chromebook 315 offers convenient and intuitive operation and access to a wide range of applications available in the Google Play store. So you can freely use your favorite applications for work, study or entertainment, knowing that they will work efficiently and without problems.

The Acer Chromebook 315 laptop is equipped with a Full HD screen that provides a clear and sharp image. Thanks to this, you can enjoy high visual quality while browsing websites, watching movies or using multimedia applications.

With a large capacity of 128GB SSD, you have enough space to store your files, documents and multimedia. Additionally, using cloud services, you can access your data from any location and device, which provides flexibility and mobility at work.

The durable battery of the Acer Chromebook 315 laptop provides long hours of work without the need for frequent charging. Thanks to this, you can work calmly all day long without worrying about losing power, which increases your productivity and mobility.

The Acer Chromebook 315 laptop is also a lightweight and compact design, which makes it easy to take with you on a trip or to a meeting. The materials used are durable and resistant to damage, ensuring long-lasting performance and reliability.

To sum up, the Acer Chromebook 315 N4500/8GB/128/FHD ChromeOS is an excellent choice for those looking for an efficient, reliable and easy-to-use laptop for work, study or entertainment. With advanced features, convenient operation and attractive design, it will meet the expectations of even the most demanding users.

You May Also Like in [laptops](#)



CATEGORY: LAPTOPS > GAMING

Microsoft Surface Laptop Studio
i7/32GB/2TB/GeForce

\$4200.00 **\$3900.00**

[ADD TO CART](#)



CATEGORY: LAPTOPS > LEARNING

Dell Inspiron 3520 i5-1235U/16GB/1TB/Win11 120Hz Srebrny

\$930.00 **\$920.00**

[ADD TO CART](#)



CATEGORY: LAPTOPS > OFFICE

HP 255 G9 Ryzen 5-5625U/16GB/512/Win11

\$1536.00 **\$1578.00**

[ADD TO CART](#)



CATEGORY: LAPTOPS > LEARNING

Apple MacBook Air M2/16GB/256/Mac OS Midnight

\$5200.00 **\$4900.00**

[ADD TO CART](#)

Rysunek 21: Strona produktu (część dolna) - sekcje rekomendacji produktów.

Rysunek 21 przedstawia dolną część strony produktu z dwoma sekcjami rekomendacji:

- „**You May Also Like**” - rekomendacje oparte produktach podobnych kategorii
- „**Frequently Bought Together**” - rekomendacje produktów podobnych firm

Każda sekcja wyświetla produkty w formacie kafelków z obrazem, nazwą, ceną i przyciskiem „Add to Cart”.

Product	Name	Quantity	Total Price	Remove
	A4Tech HD PK-910P USB Black	<button>-</button> 2 <button>+</button>	\$59.98	
	ACEFAST Powerbank MagSafe M10 10000 mAh 2x USB-C PD 30W +kabel USB-C	<button>-</button> 1 <button>+</button>	\$59.99	
	Silver Monkey Kabel USB-A na USB-C 1m 45W	<button>-</button> 1 <button>+</button>	\$12.99	
	Trust TW-200 FULL HD	<button>-</button> 1 <button>+</button>	\$39.99	

Frequently Bought Together

	Sumup Mobile Payment Terminal Air.. \$129.99 <small>Confidence: 33% Lift: 52.33x Support: 0.6%</small> ADD TO CART		Samsung Galaxy Tab A9 X110 WiFi 4/64GB.. \$199.99 <small>Confidence: 33% Lift: 52.33x Support: 0.6%</small> ADD TO CART		Velbon EX-430 \$39.99 <small>Confidence: 33% Lift: 52.33x Support: 0.6%</small> ADD TO CART		FiiO D03K Taishan \$39.99 <small>Confidence: 50% Lift: 39.25x Support: 0.6%</small> ADD TO CART		Brennenstuhl ECO-LINE Power Strip 3... \$19.99 <small>Confidence: 50% Lift: 39.25x Support: 0.6%</small> ADD TO CART
--	---	--	--	--	--	--	--	--	---

Cart Total

Total Price: \$172.95

Total Items: 5

[Continue Shopping](#)
[Checkout](#)

Rysunek 22: Widok koszyka z sekcją „Frequently Bought Together”.

Rysunek 22 przedstawia widok koszyka zawierający:

- **Lista produktów** - nazwa, obrazek, cena, ilość (zmiana liczby lub usunięcie)
- **Podsumowanie zamówienia** - Subtotal, Shipping, Total
- **Przycisk „Proceed to Checkout”**
- **Sekcja „Frequently Bought Together”** - rekomendacje Apriori oparte na produktach w koszyku

The screenshot shows a dashboard titled 'Debug Tools - ML Methods Inspector' for 'SmartRecommender'. It displays the internal workings of 7 machine learning methods. The 'Collaborative Filtering' tab is selected, showing the following sections:

- Algorithm**: Details about the algorithm, including its name ('Collaborative Filtering (Item-Based, Sarwar et al. 2001)'), formula ('Adjusted Cosine Similarity with Mean-Centering'), and status ('SUCCESS').
- Database Statistics**: Summary of data sizes:

Total Users:	19
Total Products:	500
Total Order Items:	569
Users with Purchases:	19
Total Purchases:	565
- User-Product Matrix**: Summary of matrix properties:

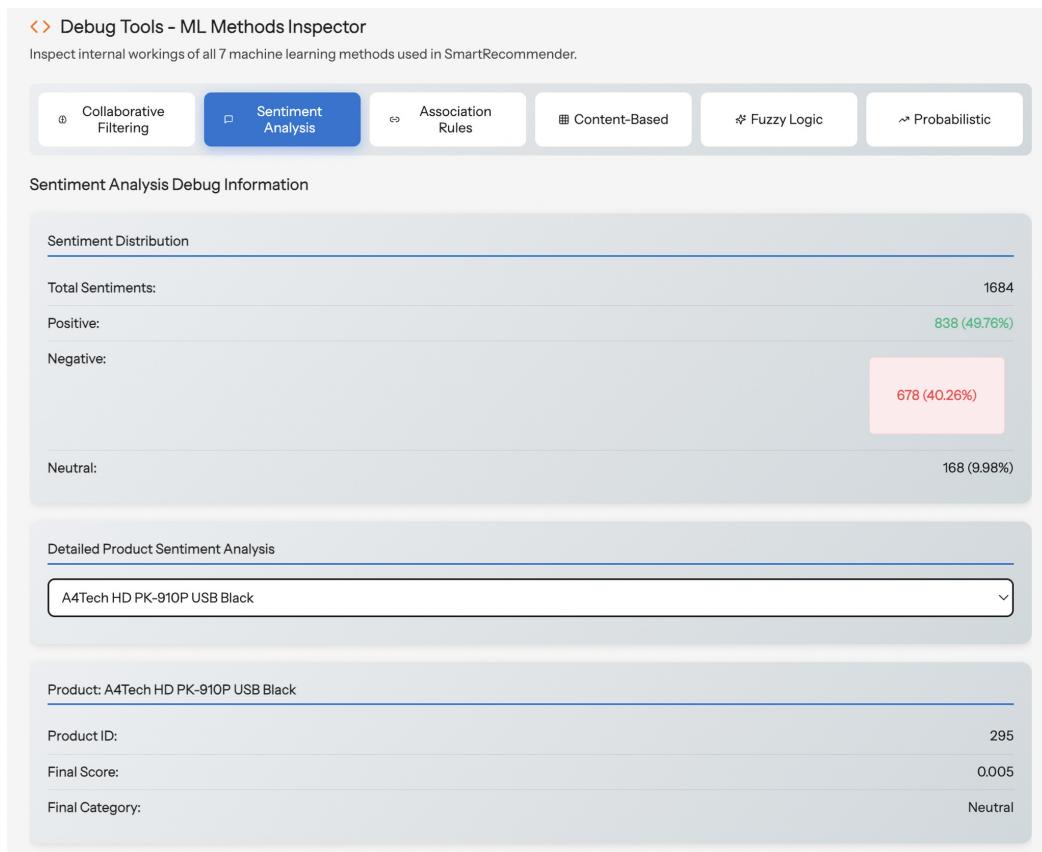
Shape:	(19, 500)
Total Cells:	9500
Non-Zero Cells:	565
Sparsity:	94.05%

Rysunek 23: Panel debugowania CF - metryki algorytmu i statystyki podobieństw.

Rysunek 23 przedstawia panel debugowania Collaborative Filtering zawierający:

- **Algorithm Details** - nazwa algorytmu, status obliczeniowy
- **Database Statistics** - liczba użytkowników, produktów, zamówień
- **User-Product Matrix** - wymiary macierzy interakcji użytkownik-produkt

Panel umożliwia administratorowi monitorowanie algorytmu i identyfikowanie problemów z danymi.



Rysunek 24: Panel debugowania Sentiment Analysis - metryki analizy sentymentu produktów.

Rysunek 24 przedstawia panel debugowania Sentiment Analysis zawierający:

- **Algorithm Details** - metoda analizy sentymentu (lexicon-based z wieloma źródłami)
- **Database Statistics** - liczba produktów z opiniami i bez opinii
- **Sentiment Distribution** - rozkład sentymentu (pozytywny/neutralny/negatywny)

Panel umożliwia administratorowi monitorowanie systemu analizy sentymentu i identyfikowanie produktów wymagających uwagi.

Debug Tools - ML Methods Inspector
Inspect internal workings of all 7 machine learning methods used in SmartRecommender.

Collaborative Filtering Sentiment Analysis Association Rules Content-Based Fuzzy Logic Probabilistic

Association Rules Debug Information

Select Product to Inspect

ACEFAST Powerbank MagSafe M10 10000 mAh 2x USB-C PD 30W +kabel USB-C

Product: ACEFAST Powerbank MagSafe M10 10000 mAh 2x USB-C PD 30W +kabel USB-C

Product ID:	297
Product Support:	0.00%
Transactions with This Product:	0
Rules for This Product:	0

Database Statistics

All Orders in Database:	193
Single Product Orders:	40
Multi-Product Orders:	153
Total Rules in System:	1000

Algorithm uses only 153 orders with 2+ products (excludes 40 single-product orders)

Algorithm Behavior

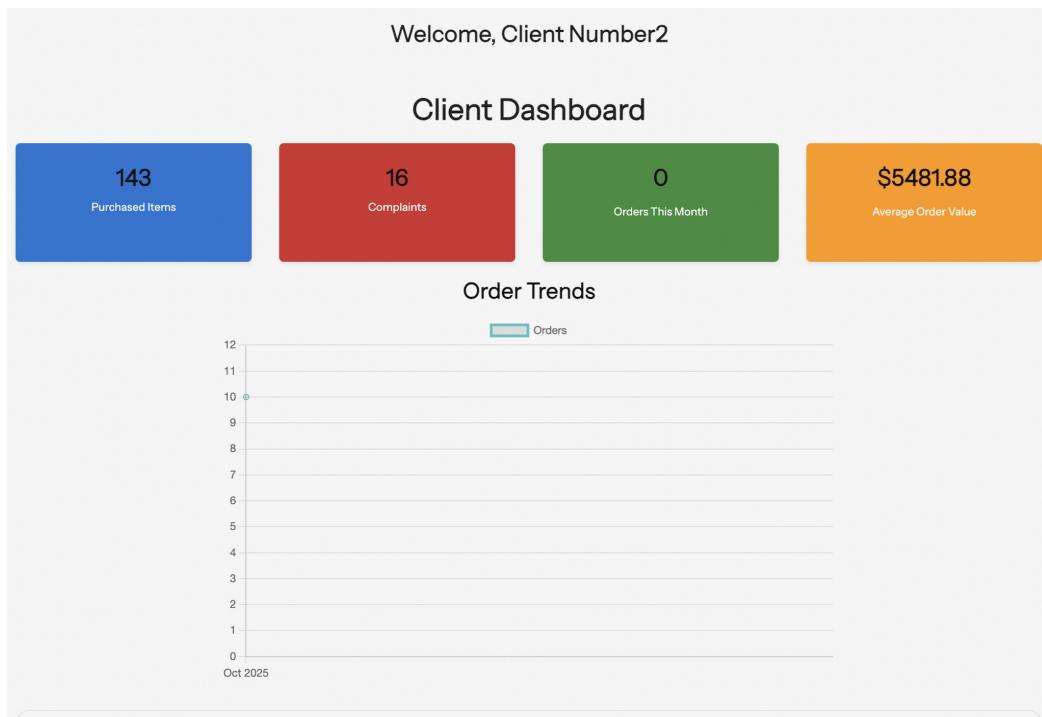
Filtering:	Association rules ONLY use orders with 2+ products
Reason:	Single-product orders cannot show 'bought together' patterns
Impact:	Using 153 transactions instead of 193 total orders

Rysunek 25: Panel debugowania Apriori - reguły asocjacyjne i statystyki transakcji.

Rysunek 25 przedstawia panel debugowania Association Rules (Apriori) zawierający:

- **Algorithm Details** - parametry algorytmu Apriori (min_support, min_confidence, min_lift)
- **Transaction Statistics** - liczba zamówień, unikalne produkty, średnia liczba produktów na zamówieniu
- **Generated Rules** - liczba wygenerowanych reguł asocjacyjnych, statystyki confidence i lift

Panel umożliwia administratorowi monitorowanie algorytmu Apriori i optymalizację parametrów.



Rysunek 26: Panel klienta z osobistymi rekommendacjami i historią zamówień.

Rysunek 26 przedstawia panel klienta zawierający:

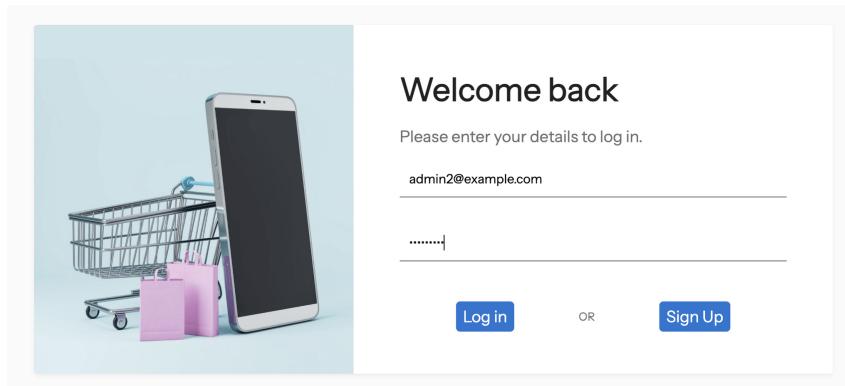
- **Powitanie użytkownika** - spersonalizowane powitanie z imieniem
- **Historia zamówień** - ostatnie zamówienia z datami, statusami, kwotami
- **Sekcje rekommendacji** - produkty rekommendowane na podstawie historii zakupów (CF) i przeglądanych kategorii
- **Statystyki użytkownika** - łączna wartość zamówień, ulubione kategorie

Panel integruje wszystkie 3 metody rekommendacji (CF, Sentiment, Apriori) w celu dostarczenia spersonalizowanego doświadczenia.

Autentykacja użytkownika

Przed uzyskaniem dostępu do personalizowanych funkcji aplikacji (koszyk, panel klienta, historia zamówień, rekommendacje CF), użytkownik musi przejść przez proces logowania. Rysunek 27 przedstawia interfejs logowania z następującymi elementami:

- Formularz logowania z polami: username/email oraz password
- Przycisk „Sign In” inicjujący proces autentykacji

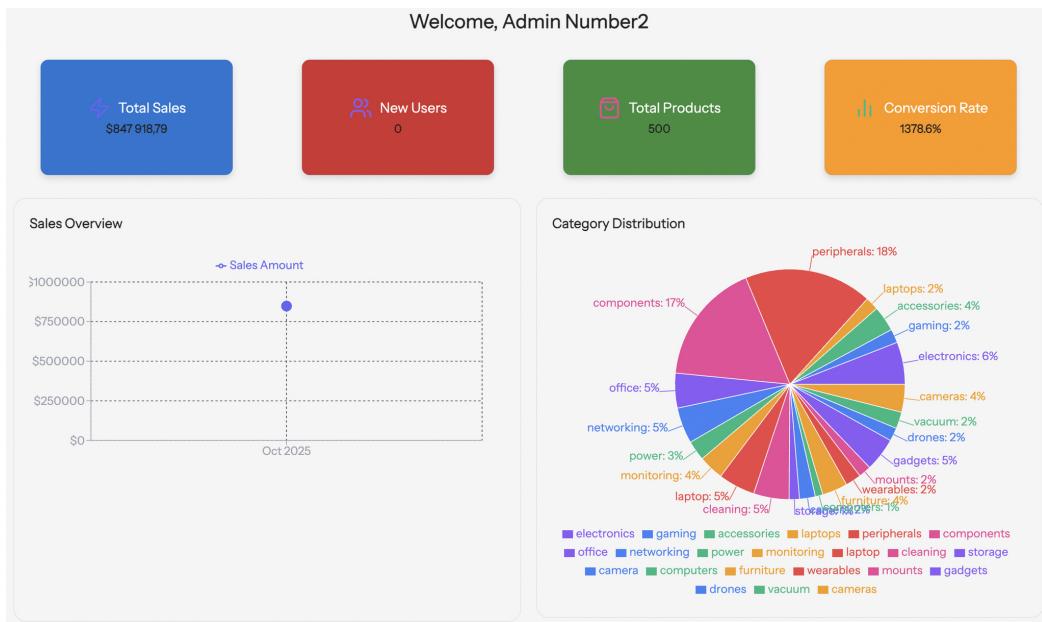


Rysunek 27: Interfejs logowania użytkownika.

System autentykacji wykorzystuje JSON Web Tokens (JWT) z biblioteką. Token zapisywany jest w localStorage.

Panel administracyjny

Administratorzy mają dostęp do rozszerzonego panelu zawierającego statystyki systemu, zarządzanie produktami, użytkownikami i zamówieniami oraz panele debugowania algorytmów rekomendacji.



Rysunek 28: Dashboard administratora - statystyki systemu i zarządzanie.

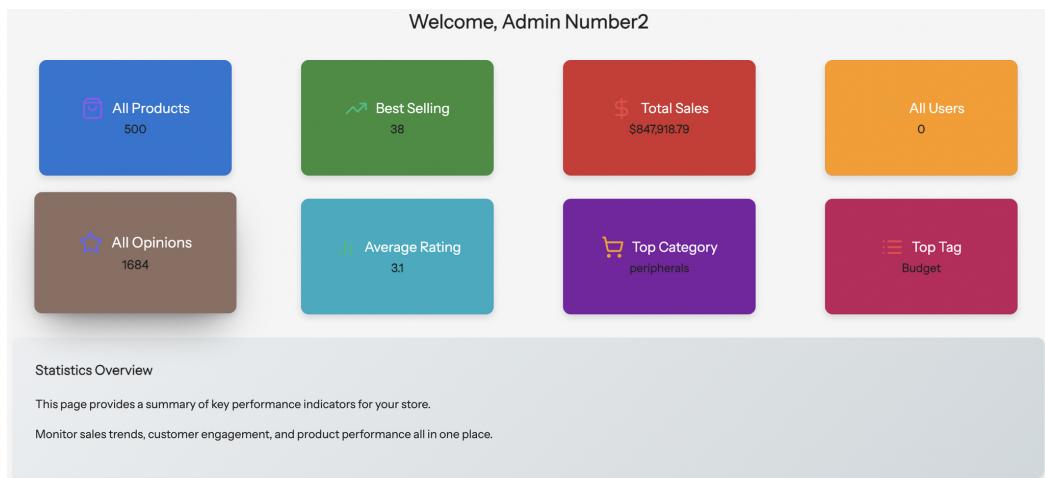
Rysunek 28 przedstawia panel administratora zawierający:

- **Statystyki systemu** - liczba produktów, użytkowników, zamówień, opinii
- **Wykresy sprzedaży** - miesięczny obrót
- **Top produkty** - bestsellery i najlepiej oceniane produkty

- **Zarządzanie** - linki do sekcji products/users/orders
- **Debug panele** - dostęp do paneli debugowania algorytmów (CF, Sentiment, Apriori)

Statystyki administratora

Panel administratora zawiera zaawansowane statystyki dotyczące funkcjonalnego systemu rekomendacji i aktywności użytkowników.



Rysunek 29: Statystyki administratora - wykresy sprzedaży i aktywności użytkowników.

Rysunek 29 przedstawia panel statystyk administratora zawierający:

- **Wykresy sprzedaży** - miesięczny przychód
- **Aktywność użytkowników** - liczba rejestracji, logowań, zamówień
- **Rozkład kategorii** - wykres kołowy przedstawiający udział kategorii w sprzedaży

Implementation Status

Overview of recommendation algorithm implementations:

Content-Based Filtering Implementation: Custom Manual Implementation Similarities: 16038 Description: Jaccard similarity with manual feature extraction Manual Implementation	Collaborative Filtering Implementation: Scikit-learn Cosine Similarity Similarities: 4150 Description: User-item matrix with cosine similarity Library Implementation
--	--

Custom Manual Implementations:
[CONTENT BASED](#) [FUZZY SEARCH](#) [ASSOCIATION RULES](#) [SENTIMENT ANALYSIS](#)

Association Rules Management [Custom Apriori](#)

Association rules help identify products frequently bought together. These rules power the "Frequently Bought Together" recommendations in the shopping cart. Using custom manual Apriori algorithm implementation with real formulas from scientific literature (Agrawal & Srikant 1994).

Product 1	Product 2	Support	Confidence	Lift
Hama Premium Surge Protector - 4 Sockets, 3m	ICY BOX Hub USB-A - 4x USB-A Aluminum	0.6%	100.0%	157.00
Creative Live! Cam Sync 1080p V2	TP-Link TG-3468 (10/100/1000Mbit)	0.6%	100.0%	157.00
TP-Link TG-3468 (10/100/1000Mbit)	Creative Live! Cam Sync 1080p V2	0.6%	100.0%	157.00
Set Z5 Intel i5-14400F, RX 7800 XT 12GB, 16GB DDR4, 500GB SSD, 650W, MSI 112 ARGB	Dell Inspiron 3520 i5-1235U/16GB/1TB/Win11 120Hz Srebrny	0.6%	100.0%	157.00
Dell Inspiron 3520 i5-1235U/16GB/1TB/Win11 120Hz Srebrny	Set Z5 Intel i5-14400F, RX 7800 XT 12GB, 16GB DDR4, 500GB SSD, 650W, MSI 112 ARGB	0.6%	100.0%	157.00
Silver Monkey USB-A 4x USB 3.0 (Silver)	Samsung Galaxy A35 5G 6/128GB 120Hz 25W Granatowy	0.6%	100.0%	157.00
Samsung Galaxy A35 5G 6/128GB 120Hz 25W Granatowy	Silver Monkey USB-A 4x USB 3.0 (Silver)	0.6%	100.0%	157.00
Lexar 32GB (2x16GB) 3200MHz CL16 Thor	Lenovo Tab M10 4GB/64GB/Android 11/WiFi Gen. 3	0.6%	100.0%	157.00
Lenovo Tab M10 4GB/64GB/Android 11/WiFi Gen. 3	Lexar 32GB (2x16GB) 3200MHz CL16 Thor	0.6%	100.0%	157.00
Unitek Hub Bi-Directional USB-C/USB-A - 4x USB-A	ASUS ROG STRIX B550-F GAMING	0.6%	100.0%	157.00

Rysunek 30: Statystyki administratora - analiza rekomendacji i reguł asocjacyjnych.

Rysunek 30 przedstawia statystyki dotyczące systemu rekomendacji:

- **Popularne kombinacje** - najczęściej kupowane zestawy produktów (Apriori)
- **Coverage algorytmów** - procent produktów objętych rekomendacjami

Recommendation System

Select Algorithm:

Collaborative Filtering (CF) - Library Implementation Content-Based Filtering (CBF) - Custom Manual Implementation Fuzzy Logic (FL) - Custom Manual Implementation

[Apply Algorithm](#)

Collaborative Filtering Preview (Library)

 DJI Dwukierunkowy ... \$39.99	 Logic Aramis Mini AR... \$40.00	 Silver Monkey Żel do ... \$8.99	 Logitech G240 Cloth ... \$14.99	 Hama 'STANDARD' E... \$49.99	 SteelSeries Rival 3 \$39.99
--------------------------------------	--	--	--	-------------------------------------	------------------------------------

Rysunek 31: Statystyki administratora - wydajność systemu i cache.

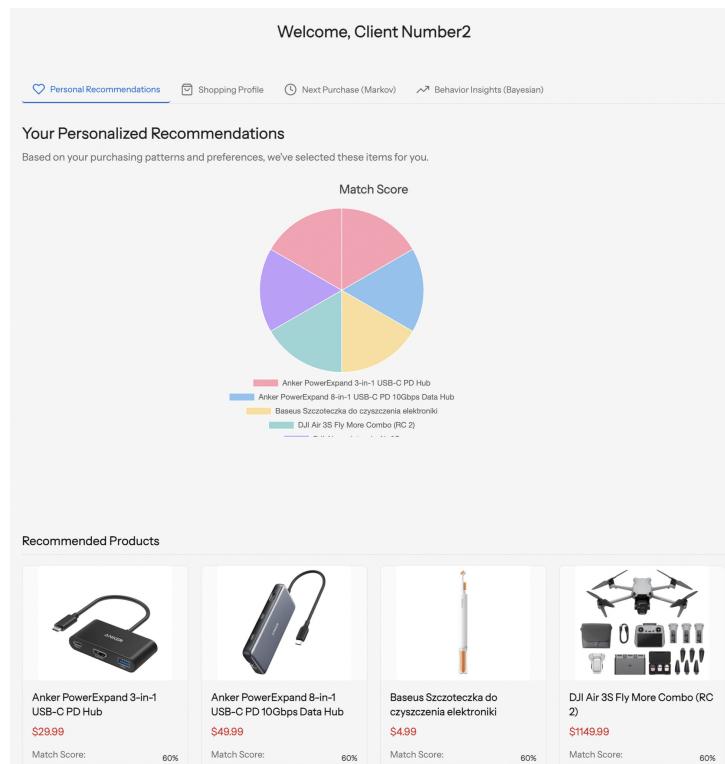
Rysunek 31 przedstawia metryki wydajnościowe:

- **Cache statistics** - hit rate, miss rate, średni czas odpowiedzi z cache

- **Database queries** - liczba zapytań, średni czas wykonania, slow queries
- **API endpoints** - najczęściej używane endpointy rekomendacji, średni czas odpowiedzi
- **Background tasks** - status przeliczania macierzy CF, Apriori, Sentiment

Statystyki klienta

Panel klienta oferuje również statystyki zakupów przedstawione na rysunku 32.



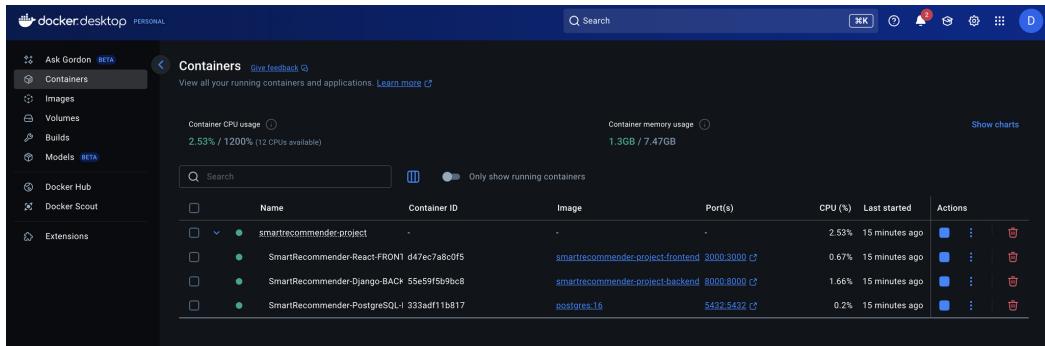
Rysunek 32: Statystyki klienta - historia zakupów i ulubione kategorie.

Rysunek 32 przedstawia statystyki klienta zawierające:

- **Rozkład produktów na wykresie** - procentowy rozkład produktów na wykresie
- **Top produkty** - najbardziej dopasowane produkty dla użytkownika

5.6 Deployment i konteneryzacja

Aplikacja została skonteneryzowana przy użyciu Docker, co zapewnia spójność środowiska między development/staging/production oraz upraszcza proces wdrożenia. Rysunek 33 przedstawia konfigurację Docker Compose z trzema kontenerami.



Rysunek 33: Deployment Docker - architektura trójwarstwowa (backend Django + frontend React + baza PostgreSQL).

Architektura deploymentu składa się z trzech kontenerów Docker:

1. **Backend Container:** Django 4.2, port 8000.
2. **Frontend Container:** React 18, port 3000.
3. **Database Container:** PostgreSQL 14, port 5432.

Docker Compose orchestruje wszystkie kontenery z automatyczną obsługą zależności.

Zalety konteneryzacji:

- **Reproducibility** - identyczne środowisko dev/prod eliminuje problemy „works on my machine”
- **Isolation** - każdy serwis w osobnym kontenerze, zero konfliktów zależności
- **Portability** - build image raz, uruchom na dowolnym serwerze z Docker
- **Easy setup** - docker-compose up uruchamia całą aplikację jedną komendą

Rozdział 6

Podsumowanie i wnioski końcowe

W pracy zaimplementowałem system rekomendacji łączący trzy metody: Collaborative Filtering (Adjusted Cosine Similarity), analizę sentymentu (słownikowa, 5 źródeł) oraz Apriori (bitmap pruning). Każda metoda wnosi coś innego: CF znajduje podobne produkty, sentiment ocenia jakość, Apriori odkrywa produkty kupowane razem.

Wydajność:

CF generuje macierz w 5-10s przy pierwszym wywołaniu, potem cache. Analiza sentymentu w pasku wyszukiwania. Apriori z bitmap, dzięki czemu klienci mają proponowane produkty w koszyku. Dodatkowo indeksowanie w PostgreSQL znacznie przyspieszyło zapytania.

Ograniczenia:

Problem zimnego startu dotyczy CF i Apriori (brak danych historycznych dla nowych użytkowników/produktów). Sentiment częściowo to kompensuje — może ocenić produkt bez opinii (opis, specyfikacja). Słownik sentymentu nie radzi sobie z negacją ("nie polecam") i ironią. Dla bardzo dużych katalogów (10000+ produktów) potrzebne dalsze optymalizacje.

Kierunki rozwoju:

Głębsze uczenie maszynowe: Obecny system używa prostych algorytmów (cosine similarity, Apriori). Można zastosować sieci neuronowe, które automatycznie uczą się wzorców z danych bez ręcznego definiowania reguł. Przykład: zamiast ręcznie liczyć podobieństwa produktów, sieć neuronowa mogłaby odkryć ukryte związki między produktami (np. że klienci kupujący gaming laptops częściej kupują też gaming mice).

Rekomendacje w czasie rzeczywistym: Obecny system przelicza rekomendacje co kilka godzin (cache 24h). Można wdrożyć system aktualizujący rekomendacje natychmiast po każdej akcji użytkownika (kliknięcie, dodanie do koszyka, zakup). Wtedy użytkownik widzi produkty dopasowane do tego, co robi teraz, a nie wczoraj.

Obsługa nowych produktów: Obecny CF i Apriori źle działają dla produktów bez historii zakupów ("zimny start"). Można zastosować techniki matrix factorization (SVD), które przewidują preferencje użytkownika nawet dla produktów, których nigdy nie kupił, na podstawie cech produktu (kategoria, cena, marka).

System jest gotowy do wdrożenia w środowisku produkcyjnym. Implementacja od podstaw (bez gotowych bibliotek) pozwoliła mi świadomie dostosować algorytmy do wymagań e-commerce.

Zakończenie

Stworzyłem system rekomendacji łączący Collaborative Filtering, analizę sentimentu i Apriori w aplikacji Django + React + PostgreSQL. Każda metoda wnosi coś unikalnego: CF znajduje podobieństwa w zakupach, sentiment ocenia jakość produktów, Apriori odkrywa produkty kupowane razem.

Kluczowe osiągnięcia:

- Komplementarność metod — rozwiązuje różne problemy (CF: podobieństwa, sentiment: jakość, Apriori: cross-selling)
- Implementacja od podstaw — głębokie zrozumienie algorytmów, możliwość dostosowania do e-commerce
- Optymalizacja wydajności — bitmap pruning, cache, indeksy umożliwiają wdrożenie produkcyjne
- Interpretowalność vs dokładność — wybrałem metody zrozumiałe i łatwe w debugowaniu

System jest gotowy do wdrożenia. Praca nauczyła mnie projektowania systemów ML, optymalizacji algorytmów, full-stack development (Django + React) oraz projektowania baz danych.

Wykaz ilustracji, rysunków, wykresów i tabel

Spis rysunków

1	Sekcja „Our Latest Products” - sortowanie Collaborative Filtering.	11
2	Sekcja „Our Latest Products” - sortowanie Sentiment Analysis.	11
3	Panel debugowania CF - podstawowe metryki algorytmu.	12
4	Panel debugowania CF - szczegółowa tabela rekommendacji z wartościami similarity_score.	13
5	Diagram sekwencji: Collaborative Filtering - proces generowania rekommendacji produktów podobnych.	14
6	Formularz dodawania opinii na stronie produktu z oceną gwiazdkową i recenzją tekową.	15
7	Lista opinii produktu z badge'ami sentymentu i systemem głosowania pomocności.	16
8	Wyszukiwarka z sortowaniem według analizy sentymentu - najlepiej oceniane produkty na górze.	17
9	Diagram sekwencji: Analiza sentymentu - wieloźródłowa agregacja sentymentu z pięciu źródeł tekstowych.	21
10	Panel debugowania Sentiment Analysis - metryki algorytmu i rozkład sentymentu.	22
11	Panel debugowania Sentiment Analysis - szczegółowa tabela wyników dla produktów z rozbiciem na źródła.	23
12	Diagram sekwencji: Algorytm Apriori - generowanie reguł asocjacyjnych typu „Często kupowane razem”.	27
13	Koszyk zakupowy z rekommendacjami „Frequently Bought Together” opartymi na regułach asocjacyjnych Apriori.	28
14	Panel debugowania Apriori - metryki algorytmu i wydajność bitmap pruning. .	30
15	Diagram przypadków użycia: Aktorzy (Klient, Admin, API), funkcjonalności systemu rekommendacji (przeglądanie produktów, rekommendacje, zarządzanie zamówieniami, debugowanie).	34
16	ERD: Wszystkie tabele aplikacji e-commerce (użytkownicy, produkty, zamówienia, opinie, koszyk, kategorie).	39
17	ERD: Tabele metod rekommendacyjnych (ProductSimilarity-CF, SentimentAnalysis, ProductAssociation-Apriori, UserInteractions).	40
18	Widok kategorii produktów z sortowaniem i filtrowaniem.	45
19	Sekcje strony głównej - slider główny i kategorie produktów.	46
20	Strona produktu (część górna) - galeria zdjęć, informacje, cena, specyfikacje. .	47
21	Strona produktu (część dolna) - sekcje rekommendacji produktów.	48
22	Widok koszyka z sekcją „Frequently Bought Together”.	49

23	Panel debugowania CF - metryki algorytmu i statystyki podobieństw.	50
24	Panel debugowania Sentiment Analysis - metryki analizy sentymentu produktów.	51
25	Panel debugowania Apriori - reguły asocjacyjne i statystyki transakcji.	52
26	Panel klienta z osobistymi rekomendacjami i historią zamówień.	53
27	Interfejs logowania użytkownika.	54
28	Dashboard administratora - statystyki systemu i zarządzanie.	54
29	Statystyki administratora - wykresy sprzedaży i aktywności użytkowników.	55
30	Statystyki administratora - analiza rekomendacji i reguł asocjacyjnych. . .	56
31	Statystyki administratora - wydajność systemu i cache.	56
32	Statystyki klienta - historia zakupów i ulubione kategorie.	57
33	Deployment Docker - architektura trójwarstwowa (backend Django + frontend React + baza PostgreSQL).	58

Spis tabel

1	Wydajność algorytmu Apriori - porównanie bitmap pruning z implementacją naiwną.	27
2	Przykładowe reguły asocjacyjne - najśilniejsze relacje między produktami. .	32

Streszczenie

Tytuł pracy:

System rekomendacji produktów oparty na filtracji współpracy, analizie sentymentu i regułach asocjacyjnych

Streszczenie:

Praca przedstawia system rekomendacji łączący trzy metody ML: Collaborative Filtering (Item-Based, Adjusted Cosine), analizę sentymentu (słownikowa, 5 źródeł) oraz Apriori (bitmap pruning). CF znajduje podobieństwa produktów na podstawie historii zakupów. Sentiment agreguje oceny z opinii, opisu, nazwy, specyfikacji i kategorii. Apriori odkrywa produkty kupowane razem.

Stos technologiczny: Django REST + React 18 + PostgreSQL + NumPy/scikit-learn. Optymalizacje: indeksy DB, bulk operations, cache. Wydajność dla 500 produktów: CF (hit), Apriori, sentiment.

Wartość: Implementacja od podstaw pozwoliła świadomie dostosować algorytmy do e-commerce. System rozwiązuje zimny start (sentiment działa bez opinii) i jest gotowy do wdrożenia produkcyjnego.

Słowa kluczowe:

systemy rekomendacji, collaborative filtering, analiza sentymentu, algorytm Apriori, machine learning, e-commerce, Django, React, PostgreSQL

Title:

Product Recommendation System Based on Collaborative Filtering, Sentiment Analysis and Association Rules

Abstract:

This thesis presents a recommendation system combining three machine learning methods: Collaborative Filtering (Item-Based, Adjusted Cosine), sentiment analysis (lexicon-based, 5 sources), and Apriori (bitmap pruning). CF discovers product similarities based on purchase history. Sentiment aggregates ratings from reviews, descriptions, names, specifications, and categories. Apriori discovers frequently bought together products.

Technology stack: Django REST + React 18 + PostgreSQL + NumPy/scikit-learn.

Optimizations: DB indexes, bulk operations, cache. Performance for 500 products: CF (cache hit), Apriori, sentiment.

Value: Implementation from scratch enabled conscious algorithm adaptation for e-commerce. System solves cold start problem (sentiment works without reviews) and is production-ready.

Keywords:

recommender systems, collaborative filtering, sentiment analysis, Apriori algorithm, machine learning, e-commerce, Django, React, PostgreSQL

Literatura

- [1] Rakesh Agrawal, Ramakrishnan Srikant, *Fast Algorithms for Mining Association Rules*, Proceedings of the 20th International Conference on Very Large Data Bases, 1994.
- [2] James Bennett, Stan Lanning, *The Netflix Prize*, Proceedings of KDD Cup and Workshop, 2007.
- [3] Greg Linden, Brent Smith, Jeremy York, *Amazon.com Recommendations: Item-to-Item Collaborative Filtering*, IEEE Internet Computing, Vol. 7, No. 1, 2003.
- [4] Bing Liu, *Sentiment Analysis and Opinion Mining*, Morgan \& Claypool Publishers, 2012.
- [5] Jacques Bughin, Michael Chui, James Manyika, *Ten IT-enabled business trends for the decade ahead*, McKinsey Quarterly, May 2013.
- [6] Paul Resnick, Hal R. Varian, *Recommender Systems*, Communications of the ACM, Vol. 40, No. 3, 1997.
- [7] Badrul Sarwar, George Karypis, Joseph Konstan, John Riedl, *Item-based Collaborative Filtering Recommendation Algorithms*, Proceedings of the 10th International Conference on World Wide Web, 2001.
- [8] Mohammed J. Zaki, *Scalable Algorithms for Association Mining*, IEEE Transactions on Knowledge and Data Engineering, 2000.