

Budowanie systemu rekomendacji (cz. 3)

Factorization Machines – jak piekarz może stworzyć skuteczny system rekomendacji na podstawie danych z logów klienta



Współczesne sklepy internetowe, nawet te małe, prowadzone przez lokalnych piekarzy, mogą skutecznie wykorzystywać dane o klientach do personalizacji ofert. Jednym z najskuteczniejszych podejść do tworzenia rekomendacji produktowych jest zastosowanie metody Factorization Machines (FM).

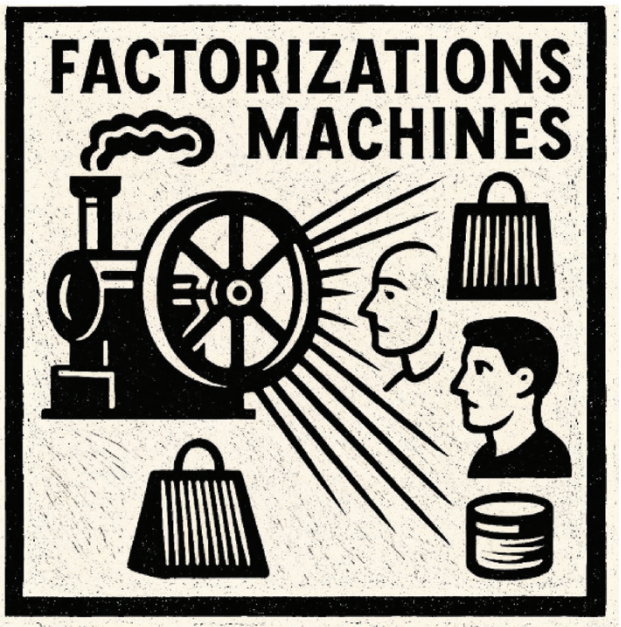
Poniżej wyjaśnię, jak stworzyć taki system krok po kroku, pokażę różnice między FM a klasycznym podejściem typu PCA+ k-means oraz zaprezentuję przykład, jak taka personalizacja może wpłynąć na wzrost sprzedaży.

Zbieranie danych o klientach

W poprzedniej części wyjaśniłem w jaki sposób zbierać informacje na temat swoich klientów. Przede wszystkim, aby zrobić jakąś analizę musimy mieć bazę danych dotyczącą naszych klientów. Analizujemy sklep z wyrobami cukierniczymi, który prowadzi działalność w internecie. Każdy klient, aby dokonać zakupu musi najpierw się zalogować. W poprzedniej publikacji wskazałem na to, że właściciel ma własną stronę internetową i bezpośredni dostęp do logów. Logi to proste informacje generowane przez system, które zapisują poczynania klientów. Pierwszy loguje, że klient zalogował się do sklepu używając hasła i własnego loginu. Kolejny log informuje co klient wpisał do wyszukiwarki, do jakich działów strony internetowej wchodził. Z logów możemy dowiedzieć się jak długo klient dokonywał zakupu czy się wahał, czy rezygnował z zakupu. Logi systemowe to skarbnica informacji. W poprzedniej publikacji wyjaśniłem w jaki sposób umieścić informacje z logów do tabeli. Tabela ta jest bazą danych opisującą cechy charakterystyczne i sposób zachowania każdego poszczególnego klienta. W tej części chciałbym pokazać w jaki sposób mając 30 charakterystycznych cech dla około 100 tys. klientów naszego sklepu stworzyć system rekomendacji, który indywidualnie dla każdego klienta będzie dobierał odpowiednie rekomendacje.

W naszym przykładzie na podstawie logów aktywności klienta wyodrębniono tzw. cechy opisowe (*features*), np.:

ID klienta	Średni czas do zakupu	Procent transakcji z rabatem	Maks. wartość koszyka	Czas aktywności	Wiek klienta
1001	krótki	5	120 zł	wieczory	26-35
1002	długi	25	80 zł	rano	18-25
1003	średni	10	200 zł	weekendy	35-45



W praktyce takich cech może być 30 i więcej, uwzględniając np. preferowane smaki, częstotliwość zakupów, ulubione dni tygodnia itd.

Budowa systemu rekomendacji z Factorization Machines

Factorization Machines to model uczenia nadzorowanego (regresja/klasyfikacja), który rozszerza zwykłą regresję liniową o możliwość *automatycznego uczenia się interakcji pomiędzy parami cech*, nawet przy bardzo dużej wymiarowości danych (30 i więcej cech). FM efektywnie modeluje zależności pomiędzy cechami dzięki tzw. ukrytym czynnikom (*embeddingom*) zamiast *explicit* (cechą widzialnym) tworzyć wszystkie kombinacje

cech widzialnych i niewidzialnych. To pozwala wychwycić wzorce, które tradycyjny model liniowy by pominął, a jednocześnie uniknąć eksplozji liczby parametrów (przekleństwo wymiarowości). FM stał się popularny w systemach rekomendacji i prognozowania kliknięć właśnie dzięki tej zdolności do modelowania interakcji na danych wysokiej wymiarowości. Model FM przewiduje więc prawdopodobieństwo kliknięć, przebiegu postępowania procesu zakupowego klienta.

Na bazie tych danych można trenować model Factorization Machines, który:

- rozpoznaje ukryte wzorce w preferencjach klientów,
- nie wymaga ręcznego tworzenia kombinacji cech,
- efektywnie radzi sobie z rzadkimi danymi (np. klient kupił tort tylko raz).

Model uczy się prognozować, jakie produkty są najbardziej prawdopodobne do zakupu przez konkretnego klienta. Działa lepiej niż zwykłe reguły typu „kto kupił A, kupi B”, ponieważ uwzględnia wiele subtelnych interakcji między cechami. Widzi to co widzialne i to co nie jest widzialne.

Na przykład Klient ID:1002 ma wysoką skłonność do rabatów i kupuje zwykle rano. FM nauczy się, że torty śmietankowe w poniedziałki rano z kodem rabatowym mają większą szansę zakupu przez tego klienta. Na podstawie tego, że model FM przewidział wyższe prawdopodobieństwo takiego zachowania (klienta ID:1002 – rano + promocja) system może automatycznie zasugerować ranne promocje na produkty, a nawet przygotować spersonalizowaną promocję tylko dla niego.

Typowy system rekomendacji oparty o FM może uwzględniać m.in.: cechy użytkownika (demografia, segment z zachowań, nawet kilkaset cech takich jak: płeć, skłonności do ryzyka czy różne odmiany fobii), cechy produktu (kategoria, cena, popularność itp.) oraz kontekst (pora dnia, urządzenie, kanał itp.). FM efektywnie wyliczy interakcje między tymi cechami. Znajduje ukryte interakcje czego model liniowy by nigdy nie uchwycił bez ręcznie zdefiniowanej interakcji. Właśnie dzięki takiemu automatycznemu łączeniu cech FM dobrze sprawdza się w zadaniach z bardzo rzadkimi danymi, takich jak rekomendacje produktowe czy przewidywanie kliknięć.

Matrix Factorization



towowi, co jeszcze może mu smakować – na podstawie preferencji innych podobnych klientów.

Aby lepiej zrozumieć Factorization Machines zaczniemy od jej pierwszej wersji, czyli Matrix Factorization. Nauczmy się tego przez działanie. Chcemy stworzyć system rekomendacji wypieków (np. bułek, ciast, chleba), który podpowiada każdemu klientowi,

Mamy 5 klientów i 5 produktów:

Klient	Chleb żytni	Drożdżówka	Rogal	Ciasto sernik	Chleb pszenny
Klient A	OK	OK	bd.	No-OK	bd.
Klient B	No-OK	OK	OK	OK	bd.
Klient C	OK	bd.	OK	bd.	OK
Klient D	bd.	No-OK	bd.	OK	OK
Klient E	OK	OK	OK	No-OK	OK

OK oznacza, że klientowi smakuje, No-OK nie smakuje, natomiast bd. to po prostu brak danych.

Zamiast zgadywać „co kto lubi”, Matrix Factorization:

- uczy się ukrytych cech klientów (np. preferują słodkie, miękkie, zdrowe, z makiem...),
- uczy się ukrytych cech produktów (np. słodkość, twardość, zawartość błonnika...).

W efekcie MF możemy powiedzieć w naszym przykładzie: „Klient A lubi słodkie wypieki i nie lubi ciężkich ciast → prawdopodobnie spodoba mu się rogal, bo inni o podobnym profilu go kupili i chwalili”.

Dla każdego klienta i produktu liczy dopasowanie – ile punktów dany klient dałby danemu wypiekowi nawet jeśli go jeszcze nie jadł. Szuka więc podobieństw w preferencjach klientów w stosunku do produktów. Pojęcie punktów nie bierze się znikąd.

Matrix Factorization został pierwszy raz zastosowany masowo w konkursie Netflix. Nie został wymyślony przez Netflix, ale spopularyzowany dzięki konkursowi Netflix Prize ogłoszonemu w 2006 roku z nagrodą 1 mln dolarów za poprawę dokładności ich systemu rekomendacji o 10%. Uczestnicy mieli przewidywać, jakie filmy użytkownicy polubią, na podstawie oceny filmów przez innych. Zespół BellKor's Pragmatic Chaos wygrał konkurs w 2009 roku na rozwiązanie tego problemu, a kluczową częścią ich rozwiązania było Matrix Factorization, czyli rozkład macierzy ocen użytkowników-filmy na tzw. ukryte czynniki (*latent factors* lub inna nazwa *embedding*).

Dzięki temu podejściu system był w stanie uczyć się subtelnych preferencji klientów i podobieństw między filmami. Od tego momentu MF stało się jednym z fundamentów nowoczesnych systemów rekomendacji.

„Bogate dane”, czyli czym różni się Matrix Factorization od Factorization Machines?

Kolejne warianty Matrix Factorization, takie jak Factorization Machines, stosują tzw. bogate dane. Polega to na wzbogacaniu klasycznej filtracji kolaboracyjnej o dodatkowe informacje:

- dane o **kliencie**: wiek, pora zakupu, miasto, preferencje dietetyczne,
- dane o **produkcie**: typ mąki, kaloryczność, składniki, alergeny,
- dane o **kontekście**: zakup online vs. offline, pora dnia, promocje.



Takie cechy dokleja-
my do macierzy i zamiast
czystej Matrix Factoriza-
tion (z „Netflixa”) uży-
wamy np. Factorization
Machines albo DeepFM,
które uczą się interak-
cji między tymi cechami.
Można tego dokonać za
pomocą bibliotek pytho-
na: lightFM, xlearn, scikit-

-surprise, fastFM. Jeżeli chodzi o rozwiązania chmurowe
możemy zastosować: AWS: Amazon Personalize, GCP:
Recommendations AI, Azure: Azure ML + LightGBM
+ AutoML.

Warto pamiętać, że system Factorization Machines
sam się uczy na podstawie zakupów klientów – im więcej
danych, tym lepsze rekomendacje.

Jak wygląda system rekomendacji dla każdego klienta?

Gotowy wytrenowany model FM możemy wyko-
rzystać, aby rekomendować produkty. W trybie offline
(wsadowym) zazwyczaj robi się to okresowo.

Dla każdego użytkownika (lub dla aktywnych użyt-
kowników) generujemy prognozy dla poszczególnych
produktów, które mogą go zainteresować. Ponieważ
pełne sprawdzenie wszystkich kombinacji użytkow-
nik-produkt może być kosztowne przy bardzo dużym
asortymencie, w praktyce zawęża się je do produktów
popularnych, z danej kategorii, nowych itp. Model FM
liczy prawdopodobieństwo dla każdego indywidual-
nego klienta dla tego mniejszego zbioru produktów.
Oczywiście dla małego sklepu z tortami asortyment jest
na tyle mały, że można przeliczyć oceny dla wszystkich
produktów. Sortujemy produkty po przewidywanym
prawdopodobieństwie zakupu (lub ocenie preferencji)
i wybieramy top-10 dla każdego użytkownika. Tak wyge-
nerowane listy rekomendacji można zapisać np. w bazie
danych. Np. co tydzień aktualizujemy listę rekomendo-
wanych tortów dla każdego klienta na podstawie naj-
nowszych danych.

Wykorzystanie rekomendacji: w aplikacji (np. sklepie
internetowym) po zalogowaniu użytkownika możemy
wyświetlić mu jego spersonalizowaną listę polecanych
produktów z tej gotowej listy. W systemie wsado-
wym lista ta pozostaje stała do kolejnej aktualizacji
(np. do następnego dnia).

Porównanie Factorization Machines z PCA + k-means

Swego czasu długo rozpisywałem się na temat do-
skonałości metody przypisywania klientów do klastrów
zakupowych, pisałem o metodzie PCA + k-means.

Metoda ta polegała na tym, że zbierałem informacje
na temat zachowań klientów w sklepie internetowym:
jak często bywali w sklepie, jak dużo wydawali na zaku-
py, jak długo podejmowali decyzje. Potem dodawałem
dane statystyczne, takie jak: wiek, miejsce zamieszka-
nia, rodzaj prowadzonej działalności. Nie można zro-
bić klasteryzacji k-mean na podstawie np. 40 cech.
Konieczna jest redukcja wymiarowości. Można zrobić
klasteryzację na podstawie dwóch, góra trzech cech.
W związku z tym odpalałem PCA, czyli metodę, która
grupowała cechy klientów w zazwyczaj 2-3 pakiety, tzw.
trzy główne komponenty, w których znajdowały się ce-
chy przypisane do poszczególnych klientów. Ta metoda
rekomendacji niestety jest już przestarzała. Wspomi-
nam o niej, ponieważ wiele moich publikacji było jej
poświęconych.

Klasyczne podejście segmentacyjne PCA + k-means
polega na:

- redukcji wymiarów cech (PCA),
- klasteryzacji klientów (np. na 5 grup),
- przypisaniu oferty czyli rekomendacji dla każdej
z 5 grup.

Była to ograniczona metoda rekomendacji, ponieważ
wszyscy w segmencie dostawali to samo, a przecież lu-
dzie nawet bardzo podobni się od siebie różnią. Metoda
ta nie uwzględniała zmieniających się preferencji. Nie
było też uczenia na podstawie wyników, nie było infor-
macji zwrotnej czy klient faktycznie kupił.

Dlaczego Factorization Machines jest lepsze od PCA + k-means?

Są trzy powody:

- FM uczy się bezpośrednio na danych z transakcji
zakupowych więc jest szybszy w reakcji,
- daje rekomendacje spersonalizowane, a nie gru-
powe, przecież nawet podobni ludzie się różnią,
- FM lepiej radzi sobie z nowymi produktami i klien-
tami.

Factorization Machines to nie jest system klasteryzacji

Czy metoda Factorization Machines (oraz jej rozwi-
nięcie DeepFM) to specjalny sposób klastrowania klien-
tów w oparciu o liczne cechy i jeżeli nie jest to system
klasteryzacyjny to czym różni się od np. segmentacji
po PCA + k-mean? Należy podkreślić, że FM to model
predykcyjny, a nie metoda klasteryzacji. PCA (analiza
głównych składowych) służy do redukcji wymiarów/wy-
krywania struktur w danych w sposób nienadzorowany,
a klasteryzacja k-means grupuje podobne obiekty bez
informacji o żadnej zmiennej docelowej. Natomiast FM
to algorytm nadzorowany – trenujemy go do przewi-
dywania określonego targetu (np. czy użytkownik kupi
produkt, czy nie). Jest więc systemem predykcyjnym.

W odróżnieniu od PCA, które tworzy liniowe kombinacje cech, FM uczy się nieliniowych wzorców dzięki ukrytym czynnikom. Oznacza to, że FM nie tyle przypisuje klientów do statycznych klastrów, co jest uczącym się modelem pozwalającym prognozować zachowanie (np. skłonność do zakupu danego produktu) na podstawie kombinacji cech (cech „bogatych” o czym wspominałem wcześniej). Oczywiście, w trakcie uczenia FM powstają wektory ukryte dla klientów – można je traktować jako osadzenia w przestrzeni czynników ukrytych (wspominane wcześniej *latent factors* lub inna nazwa *embedding*). Klienci o podobnych wektorach (grupujących różne cechy) będą zachowywać się podobnie (co przypomina klastry klientów), lecz te reprezentacje są wyuczone pod konkretne kryterium (np. maksymalizacja trafności rekomendacji), a nie czysto statystyczne jak w PCA. Dla segmentacji marketingowej można by użyć metody PCA + klasteryzacja k-means na 30 cechach, ale taki podział niekoniecznie maksymalizuje konwersję. FM natomiast nauczy się ukrytych zależności (*latent factors*) grupujących klientów w sposób optymalny z punktu widzenia rekomendacji (np. ujawni, którzy klienci reagują na kody rabatowe, a którzy na szybkość dostawy itp.), co ma większą wartość przy personalizacji oferty tworzonej przez system rekomendacji.

Czynniki ukryte

W metodzie Factorization Machines czynniki ukryte (*latent factors*) to ukryte cechy (czyli niewidoczne bezpośrednio w danych), które reprezentują właściwości



obiektów i cech, pozwalając modelowi uchwycić interakcje między nimi, nawet gdy dane są rzadkie.

FM rozszerzają klasyczny model regresji liniowej, dodając drugorzędowe interakcje między zmiennymi, które są modelowane za pomocą wektorów ukrytych (*latent vectors*). Każdej zmiennej przypisany jest taki wektor, czyli właśnie *latent factors*.

Jeśli mamy użytkownika A i produkt „Tort bezowy” to zamiast po prostu przyjąć typową dla modelu liniowego

wagę interakcji: „użytkownik A – produkt „Tort bezowy””, FM przypisuje użytkownikowi i produktowi wektory latentne (np. o wymiarze 10), które kodują np. „gust użytkownika” i „charakterystyki produktu”, czyli wektor zawierający wiele, w tym wypadku 10, cech produktu i użytkownika. Te 10 cech nazywane jest „danymi bogatymi”. Model uczy się, jak bardzo te 10 wektorów jest do siebie dopasowanych (np. przez ich iloczyn skalarny), co stanowi predykcję interakcji (np. oceny produktu, kliknięcia, zakupu).

Latent factors pozwalają odtworzyć ukryte powiązania między elementami, np. że: użytkownicy lubiący „karmel” często lubią też „bezy” (nawet jeśli nie ma na to powiązanie danych „na wprost”). Produkty o podobnych *latent factors* są postrzegane podobnie przez użytkowników.

Na koniec ważna uwaga: iloczyn skalarny może być używany do różnych celów i pojęcie wektory latentne może się od siebie różnić.

FM: Użytkownik A ma wektor latentny, Produkt B ma inny wektor latentny. Model liczy iloczyn skalarny tych wektorów i przewiduje np. ocenę produktu, prawdopodobieństwo zakupu.

PCA: Zbiór cech (np. wiek, płeć, liczba zakupów) PCA tworzy nowe zmienne: PC1, PC2... które są kombinacjami oryginalnych cech i wyjaśniają jak najwięcej wariacji wykorzystywane np. do wizualizacji lub uproszczenia danych.

Wektory latentne w FM to nie to samo co komponenty główne w PCA, chociaż obie koncepcje są związane z ukrytą strukturą danych i mogą prowadzić do obniżenia wymiarowości.

Latent vectors są uczone i zoptymalizowane pod predykcję. PCA tworzy statystyczną projekcję bez znajomości celu.

Podsumowanie

Rzemieślnik prowadzący sklep internetowy, dysponując danymi o klientach, może zbudować skuteczny system rekomendacji z wykorzystaniem FM. Takie podejście jest bardziej precyzyjne niż klasyczne grupowanie klientów, a przy tym relatywnie proste we wdrożeniu, szczególnie w wersji wsadowej. Dzięki temu nawet lokalna cukiernia może korzystać z narzędzi, które do niedawna zarezerwowane były dla największych platform e-commerce.

Wojciech Moszczyński

Wojciech Moszczyński – absolwent Katedry Ekonometrii i Statystyki Uniwersytetu Mikołaja Kopernika w Toruniu, specjalista z zakresu ekonometrii, finansów, *data science* oraz rachunkowości zarządczej. Specjalizuje się w optymalizacji procesów produkcyjnych i logistycznych. Prowadzi badania w obszarze rozwoju i zastosowania sztucznej inteligencji. Od lat zaangażowany w popularyzację *machine learning* oraz *data science* w środowiskach biznesowych.