Inżynieria uczenia maszynowego - projekt

Tomasz Owienko Anna Schäfer

29.11.2023

1 Temat projektu

Temat projektu przekazany przez Klienta:

> *Może bylibyśmy w stanie wygenerować playlistę, która spodoba się kilku wybranym osobom jednocześnie? Coraz więcej osób używa Pozytywki podczas różnego rodzaju imprez i taka funkcjonalność byłaby hitem!*

2 Problem biznesowy

Klientem jest właściciel portalu "Pozytywka", będącego serwisem muzycznym, pozwalającym użytkownikom na odtwarzanie utworów online.

Celem projektu jest realizacja funkcjonalności pozwalającej użytkownikom serwisu Pozytywka na generowanie playlist, z których utwory podobać się będą wybranej grupie użytkowników. Taka funkcjonalność mogłaby być wykorzystywana do automatycznego układania playlist na imprezy w taki sposób, aby ich zawartość trafiała w gust jak największej części odbiorców. Implementacja takiej funkcjonalności ma zwiększyć zadowolenie użytkowników z jakości playlist odtwarzanych na imprezach, tym samym zwiększając ich zadowolenie z użytkowania portalu.

Biznesowe kryterium sukcesu

- W co najmniej 1/4 uruchomień, playlista zostanie odtworzona przez minimum 35% jej czasu trwania w ciągu jednej z najbliższych 20 sesji jednego z użytkowników, który brał udział w tworzeniu playlisty.

2.1 Założenia

- Playlisty generowane będą na podstawie profili oraz historii sesji nie więcej niż 10 użytkowników jednocześnie,
- playlisty w większości przypadków użycia nie będą wykorzystywane wielokrotnie,
- dobór kolejności utworów na playliście nie jest przedmiotem zadania,
- dostęp do playlisty mają wszyscy użytkownicy, których profile i historia sesji były uwzględnione przy jej generowaniu,
- generowane playlisty składają się z 20 utworów każda.

2.2 Pożądane cechy rozwiązania

- Playlista może być wygenerowana w bardzo krótkim czasie,
- funkcjonalność zachowuje się poprawnie dla nowo dodanych użytkowników oraz utworów,
- w ocenianiu gustu muzycznego poszczególnych użytkowników większe znaczenie powinny mieć niedawno odtwarzane utwory.

3 Zadanie modelowania

Projekt zakłada zamodelowanie problemu jako zadanie generowania rekomendacji dla zadanej grupy użytkowników (wejście modelu - ich identyfikatory). Planowane jest zastosowanie podejścia *collaborative filtering*, które (w kontekście zadania) opiera się na wyszukiwaniu użytkowników podobnych do rozpatrywanych i generowania rekomendacji w oparciu o ich historie sesji. Do realizacji podejścia *collaborative filtering* zastosowana zostanie technika rozkładu macierzy interakcji między użytkownikami, a utworami. Przewidziane jest porównanie jakości modelu korzystającego z macierzy *feedbacku niejawnego* (użytkownik X odtworzył utwór Y), oraz *feedbacku jawnego* (użytkownik X wystawił utworowi Y ocenę Z).

Podejście *collaborative filtering* pozwala na generowanie rekomendacji dla pojedynczego użytkownika. Aby dostosować je do problemu, generowanie rekomendacji dla wielu użytkowników jednocześnie zamodelowane będzie jako:

1. wygenerowanie bardzo dużej liczby rekomendacji dla każdego z użytkowników wraz z oceną podobieństwa poszczególnych utworów do gustu muzycznego użytkownika (liczba rekomendacji proporcjonalna do liczby użytkowników),
2. znormalizowanie ocen podobieństwa do zakresu $< 0, 1 >$,
3. wyznacznie zbioru utworów, które pojawiły się w rekomendacjach wszystkich użytkowników,
4. wybór tych utworów, dla których iloczyn ich ocen dla wszystkich użytkowników jest największy.

Istotnym problemem w rozważanym zadaniu jest tzw. *cold-start*, czyli zachowanie modelu dla użytkowników bądź utworów, na których model nie był trenowany. Tradycyjne podejścia rozkładu macierzy interakcji takie jak *Funk MF* czy *SVD++* nie przewidują występowania takich sytuacji i spisują się słabo w scenariuszach *cold-start*. W rozwiązaniu zostanie wykorzystany model LightFM, który rozwiązuje problem *cold-start* przez zastosowanie metadanych (atrybutów) do opisywania zarówno użytkowników, jak i utworów. Przykładowo, nowy utwór dodany do systemu nie był brany pod uwagę przy trenowaniu modelu, ale posiada on atrybuty takie jak `instrumentalness`, `tempo`, `danceability` (zgodne z taksonomią Spotify), oraz wiele innych, co można wykorzystać do wyszukiwania podobieństw.

Model bazowy

Model bazowy zostanie zaimplementowany jako prosty algorytm losujący utwory z historii odtwarzania poszczególnych użytkowników biorących udział w tworzeniu playlisty. Jest to podejście referencyjne, które posłuży do porównania z modelem docelowym.

Analityczne kryterium sukcesu

Wyznaczanie rekomendacji dla pojedynczego użytkownika - pole pod krzywą ROC powyżej $0,6$ przy szacowaniu oceny dla utworów ze zbioru testowego (kryterium bezpośrednio związane z modelem)

Wyznaczanie rekomendacji dla wielu (n) użytkowników - model lepszy niż model bazowy. Jakość wyników mierzona będzie jako stopień klasteryzacji reprezentacji utworów wchodzących w skład playlisty. Stopień klasteryzacji określony będzie jako średnia odległość reprezentacji utworów od środka klastra w przestrzeni embeddingów modelu LightFM - jest ona generowana podczas trenowania modelu. Rekomendacje lepszego modelu powinny cechować się lepszą klasteryzacją, zatem średnia odległość powinna być niższa. Stopień klasteryzacji można potraktować jako minimalizowaną zmienną celu.

4 Analiza dostarczonych danych

Z perspektywy rozwiązania problemu za pomocą podejścia *collaborative filtering* kwestią kluczową jest pozyskanie dużej ilości informacji na temat użytkowników i historii sesji - zbyt mała ilość nie pozwoli na precyzyjne wyszukiwanie podobieństw pomiędzy użytkownikami. W trzeciej wersji otrzymanych danych uzyskano dane na temat 1100 użytkowników oraz blisko 102.000 sesji, co wstępnie uznano za ilość wystarczającą.

Model LightFM pozwala na wykorzystanie metadanych do opisu utworów i użytkowników, aby rozwiązać problem *cold-start*. W przypadku użytkowników, wykorzystać można informację o zadeklarowanych przez nich preferencjach co do gatunków muzycznych. Należy mieć na uwadze, że sama deklaracja może nie być wystarczająca do znalezienia podobnych użytkowników - np w sytuacji, kiedy dana osoba posiada bardzo ubogą historię sesji, a ponadto zadeklaruje wyjątkowo nietypową kombinację preferowanych gatunków. W tym wypadku za preferencje muzyczne takiej osoby przyjęte zostaną utwory, które w ogólności charakteryzują się dużą popularnością, a ich waga przy wyznaczaniu końcowych rekomendacji zostanie ograniczona.

Z perspektywy wytestowania podejścia opartego na *jawnym feedbacku* należy rozważyć sposób zamodelowania takiej informacji - nie jest ona dana bezpośrednio w zbiorze danych. Można ją jednak oszacować wyznaczyć przez analizę historii sesji użytkownika - jednym z możliwych podejść może być wyznaczenie domniemanej oceny wystawionej przez użytkownika danemu utworowi na podstawie:

- częstotliwości odtwarzania danego utworu,
- częstotliwości występowania zdarzenia `like`,
- częstotliwości występowania zdarzenia `skip`.

Wstępnie uznano, że do realizacji zadania będą istotne przede wszystkim dane o użytkownikach portalu, ich historiach sesji, oraz utworach - pliki `users.jsonl`, `sessions.jsonl`, oraz `tracks.jsonl`:

- `users.jsonl`

  Zdecydowano się na wykorzystanie jedynie informacji o `id` użytkownika oraz (w przypadku problemów z obsługą scenariusza *cold start*) - deklarowanych preferencjach muzycznych.

- `sessions.jsonl`

  Nie odrzucono wstępnie żadnych atrybutów.

- `tracks.jsonl`

  Odrzucono (na potrzeby modelowania) jedynie atrybut `name` - ponieważ pozostałe atrybuty nie są wykorzystywane w sposób bezpośredni, a jedynie służą do wygenerowania reprezentacji utworu w przestrzeni embeddingów, zdecydowano się nie odrzucać wstępnie żadnego z nich.

4.1 Pierwsza iteracja zbierania danych

Zostały odkryte liczne braki lub błędy w danych:

- `id=-1` oraz `genres=null` w pliku `artists.jsonl`,
- `id=null` w pliku `tracks.jsonl`,
- `id=null` w pliku `sessions.jsonl`,
- `user_id=null`, `event_type=null`, oraz `track_id=null` dla `event_type!=advertisement` w pliku `sessions.jsonl`,
- sekwencje wierszy, gdzie `session_id` oraz `timestamp` mają tę samą wartość w pliku `sessions.jsonl`,
- `favourite_genres=null` w pliku `users.jsonl`.

Przy okazji prośby o nowe dane, uzgodniono znaczenie atrybutów i ich wartości w pliku `tracks.jsonl` oraz przyczynę obecności małej liczby wierszy z wartością `storage_class=fast` w pliku `track_storage.jsonl`.

4.2 Druga iteracja zbierania danych

- Otrzymano szczegółowe informacje dotyczące znaczenia atrybutów utworów z pliku `tracks.jsonl`,
- Klient wyeliminował braki oraz błędy wymienione w poprzedniej iteracji danych,
- Został dostrzeżony fakt, że w pliku `tracks.jsonl` niektóre utwory występują kilkukrotnie, a owe wystąpienia różnią się jedynie wartościami `id` i `popularity` - zwrócono na to uwagę Klientowi.

Na tym etapie podjęto decyzję o skorzystaniu z podejścia collaborative filtering do rozwiązania problemu. Niezbędne było uzyskanie od Klienta większej ilości danych na temat użytkowników oraz ich historii sesji.

4.3 Trzecia iteracja zbierania danych

- Wyjaśniono wielokrotne wystąpienia tego samego utworu w pliku `tracks.jsonl` - kilkukrotne wystąpienie tego samego utworu pod innym `id` i `popularity` nie jest błędem, a wynika z kilkukrotnego pojawienia się utworu na rynku w różnych wydaniach,
- Uzyskano większą ilość danych na temat użytkowników oraz ich historii sesji - wstępnie uznano ją za wystarczającą.