Problem biznesowy:

Aktualnie, z powodu potrzeby manualnego tworzenia playlisty dla grupy, pojawia się spore ryzyko stworzenia playlisty która przypadnie do gustu wyłącznie jej autorowi. Pojawia się znaczące ryzyko że żeby uniknąć takich problemów, użytkownicy przestaną wykorzystywać Pozytywkę na rzecz aplikacji lepiej dostosowanej do spotkań wieloosobowych.

Zadanie modelowania:

Wykorzystanie funkcji generowania rekomendacji, która dla kontekstu (zsumowanej listy piosenek wszystkich obecnych) zwraca wektor proponowanych utworów. Następnie za pomocą modelowania sekwencji jesteśmy w stanie na bieżąco dostosowywać playlistę.

Specyfikacja:

- kontekst uwzględniany w czasie rzeczywistym: pominięte utwory,

Dane wejściowe:

- dane o utworach (taneczność, energia, głośność, ilość mowy w utworze, akustyczność, instrumentalność, emocjonalny nastrój utworu, tempo)

- dane o gatunkach lubianych przez użytkowników i pisanych przez artystów

- historie słuchania użytkowników (które utwory pomija, częstotliwość odsłuchań, data danego odsłuchania/pominięcia)

Dane wyjściowe:

- generowana playlista grupowa (30 utworów z dynamiczną aktualizacją co 5 odtwarzanych utworów)

Kryteria sukcesu:

Analityczne:

Dokładność rekomendacji:

- Odsetek utworów z wygenerowanej playlisty, które użytkownicy odtwarzają (nie pomijają).
- Cel: ≥ 80% utworów z playlisty odtwarzanych bez pomijania

Czas reakcji modelu na zmiany kontekstu:

- Średni czas aktualizacji playlisty po wystąpieniu nowych danych
- Cel: Aktualizacja playlisty w czasie ≤ 15 sekund.

Różnorodność playlisty:

- Wskaźnik liczby różnych gatunków, wykonawców i nastrojów w playliście, mierzący uniknięcie monotematyczności.
- Cel: Minimum 5 różnych gatunków i 10 różnych artystów podczas generowania wstępnej playlisty

Biznesowe:

Retencja użytkowników:

- Odsetek użytkowników, którzy powracają do korzystania z funkcji tworzenia grupowych playlist w ciągu 30 dni.
- Cel: ≥ 85% użytkowników wraca w ciągu miesiąca.

Czas użytkowania aplikacji:

- Średni czas spędzany w aplikacji podczas sesji grupowego odtwarzania muzyki.
- Cel: Średni czas sesji ≥ 60 minut.

Wzrost liczby sesji grupowych:

- Liczba sesji grupowych utworzonych przez użytkowników.
- Cel: ≥ 20% użytkowników skorzysta z sesji grupowych w ciągu pierwszych 6 miesięcy po wdrożeniu.

Zwiększenie liczby użytkowników:

- Liczba nowych użytkowników przyciągniętych dzięki funkcji dynamicznych grupowych playlist.
- Cel: ≥ 10% wzrostu liczby nowych użytkowników kwartalnie.

Aktualny model bazowy

1) Modyfikujemy wyniki liczby odsłuchań, ostatnich, na przykład, trzydziestu przesłuchanych utworów każdego użytkownika.

   Robimy to poprzez zebranie liczby odsłuchań piosenek na podstawie sessions.jsonl gdzie event_type = “play” z danego zakresu czasowego.

   Skalujemy je, by otrzymać wyniki od 1 do 0 dla każdej z nich, na przykład na podstawie dzielenia wszystkich odsłuchań przez odsłuchania jednej piosenki, nazwijmy tę zmienną "wagą piosenki".

2) Łączymy wyniki w jedną listę, na podstawie łączonych (pewnie uśredniając wagowo wyniki) " wag piosenek" każdego użytkownika.

   W przypadku powtórzeń jednej piosenki na wielu listach, powtórki są eliminowane po połączeniu.

   Każdy użytkownik jest brany pod uwagę przy łączeniu wyników, jeśli w jego liście nie ma danego utworu, jest zakładane zero.

   Z powodu zmiennej ilości użytkowników, by mieć wektor o określonej długości, bierze się tylko m najlepszych wyników.

   Rekomenduje wektor utworów generowaniem rekomendacji otrzymującym listę piosenek i ich uśrednione “wagi” o długości m, zwracając wektor o stałej długości 2n.

3) Następnie rankinguje rekomendowane utwory względem preferencji każdego użytkownika, łączy wyniki dla każdego proponowanego utworu, usuwając najgorzej rankingowane, tworząc wektor o stałej długości n.

   Skala rankingów musi być odpowiednio przeskalowana, by utwory z dużym kontrastem, gdzie jedni je kochają a inni nienawidzą były mniej popularne od utworów które wszyscy po prostu lubią.

   Nie przejmujemy się tu wybraniem głónego tematu playlisty, na razie pokazywane są "propozycje" utworów które mogą się spodobać.

Krok dodatkowy dla modelu bardziej zaawansowanego

4) Potem playlista trafia do użytku i jest edytowana w czasie rzeczywistym. Można usuwać lub odsłuchiwać ponownie piosenki z listy , wpływając na generację nowych rekomendacji za pomocą modelowania sekwencji.

   Po odsłuchaniu bieżącej listy, patrząc na to które piosenki były pomijane a które odtwarzane ponownie zostają generowane nowe rekomendacje.

   Bierze się już pod uwagę jakiego typu piosenki są najbardziej popularne i dostosowywuje się do tego zamiast do preferencji indywidualnych użytkowników.

   Algorytm dostaje sekwencje utworów przesłuchanych i generuje spodziewane następne o elementów.

Wstępnie dobrane dane:

1) pobierane byłyby dane sesji z session.jsonl z ostatnich trzech miesięcy. Następnie zliczono by wystąpienia piosenek i wybrano 30 najpopularniejszych,

2) bierzemy 50 utworów z najwyższą “wagą piosenki”,

3) po rankingu wybieramy 25 najlepszych piosenek,

4) następnie generowane jest 5-10 piosenek na raz (na podstawie informacji o ostatnich 25 utworach),

5) zalecana ilość użytkowników byłaby w przedziale [2; 10], ale samo rozwiązanie nie ma ograniczeń ilości osób. Po stworzeniu wstępnej playlisty, czas generowania nowych piosenek modelowaniem sekwencji będzie stały.

Wejścia wyjścia:

1) **generowanie rekomendacji**

    a) otrzymuje listę odsłuchanych utworów z informacją o ilości przesłuchań, oraz informacje o piosenkach.

    b) zwraca listę utworów

2) **rankingowanie**

    a) trenujemy algorytm by dla każdego użytkownika na podstawie danych polubienia i pomijania z ostatnich sześciu miesięcy, podawał prawdopodobieństwo polubienia utworu

    b) zwracamy wynik dla wygenerowanych wcześniej utworów, oznaczający prawdopodobieństwo polubienia utworu przez użytkownika

3) **modelowania sekwencji**

    a) otrzymuje listę odsłuchanych utworów z przygotowanejw poprzednich krokach listy z informacją oreakcji (skip/play/like), oraz informacje o piosenkach

    b) zwraca listę kolejnych utworów

4) **wykorzystywane dane**

    a) informacje o użytkownikach (indentyfikator)

    b) informacje o sesjach odsłuchiwań (infromacja co dany użytkownik zrobił po odsłuchaniu danej piosenki w danym czasie)

    c) informacje o piosenkach (atrybuty piosenek)

Założenia:

1) Użytkownicy mają różne gusta muzyczne.

2) Zależy nam na playliście która nie powinna za często "ryzykować".

   Nie powinna wybierać mało znanych piosenek które mogą być "ryzykowne", lub piosenki które są kochane przez połowę grupy, a nienawidzone przez drugą część
   Ma to być jak "najbezpieczniejsza" playlista jaką można zrobić.
   Unikamy kontrowersyjnych utworów. na przykład jeśli 20 osób kocha piosenkę, ale 10 jej szczerze nienawidzi, piosenka ma małe szansę bycia wybraną.

   Unikamy konfliktów gustów.

3) Mamy możliwość otrzymania informacji o ilości przesłuchań piosenek użytkownika posortowanych chronologicznie.

4) Utwory mogą być pomijane, co będzie wpływać na modelowanie sekwencji i brane pod uwagę jako nieudany utwór.

5) Wstępna playlista nie musi zachowywać jednego nastroju.

Uwagi do danych:

Różnorodność i ilość wydaje się być wystarczająca, ale występują w nich błędy takie jak:

nieprawidłowo podane wartości

- ilość nieprawidłowych wartości według kolumn w pliku tracks.jsonl:
    - {'id': 1173, 'popularity': 1113, 'name': 1102, 'id_artist': 1143, 'tempo': 4}
- ilość nieprawidłowych wartości według kolumn w pliku users.jsonl:
    - {'premium_user': 5, 'favourite_genres': 3, 'id': 4}
- ilość nieprawidłowych wartości według kolumn w pliku sessions.jsonl:
    - {'track_id': 1352, 'event_type': 1316, 'user_id': 1321}

      (jako błąd nie są brane pod uwagę podanie "track_id": null, jeśli event_type to reklama)
- ilość nieprawidłowych wartości według kolumn w pliku artists.jsonl:
    - {'id': 88, 'genres': 77}
- ponieważ nie wykryto zależności dla większości tych zmian, można je zignorować. Jednak w przypadku
    - kolumny track_id w pliku sessions.jsonl, błędy wydają się zależne od session_id

      algorytm wykorzystujący session_id do przewidzenia czy track_id będzie błędnie podane, ma dokładność 0.9996 w porównaniu do algorytmu wykorzystującego losowe dane o dokładności 0.81094.

2981 rekordów w tracks.jsonl o takiej samej nazwie, ale z miejscami niezgodnymi danymi

na przykład dla utworu Pagtingin, są dwa rekordy:

{"id": "7178ubXeY1sFOqdNkKrUwb", "name": "Pagtingin", "popularity": 53, "duration_ms": 227370, "explicit": 0, "id_artist": "4DAcJXcjX0zlQAZAPAx4Zb", "release_date": "2019-05-03", "danceability": 0.669, "energy": 0.42, "key": 2, "loudness": -8.464, "speechiness": 0.0253, "acousticness": 0.506, "instrumentalness": 0.0, "liveness": 0.404, "valence": 0.523, "tempo": 95.049}

{"id": "5l9g7py8RCblcvbZgGQgSd", "name": "Pagtingin", "popularity": 66, "duration_ms": 227370, "explicit": 0, "id_artist": "4DAcJXcjX0zlQAZAPAx4Zb", "release_date": "2019-05-12", "danceability": 0.669, "energy": 0.42, "key": 2, "loudness": -8.464, "speechiness": 0.0253, "acousticness": 0.506, "instrumentalness": 0.0, "liveness": 0.404, "valence": 0.523, "tempo": 95.049}

Po analizie, błąd wydaje się mieć związek z kolumną “id” pliku track.jsonl, oraz “release-date”.

Algorytm przewidujący możliwość podwojenia na podstawie tych dwóch kolumn ma szansę przewidzenia błędu równą 0.8026, w porównaniu do przewidywania na podstawie losowych liczb o dokładności 0.6212.

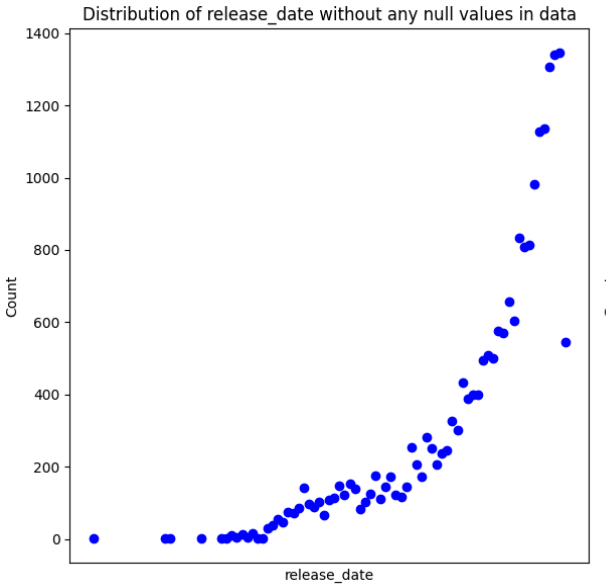
trzy różne formaty release date,

na przykład:

- release_date: "1956-01-01"
- release_date: "1956-03"
- release_date: "1929"

Rozkład dla samego roku wydaje się być prawidłowy, ale potrzebujemy dokładniejszych danych.

Napotkaliśmy również anomalię w release_date, wykazującą że ostatni sprawdzany rok ma nieprawidłowo małą liczbę wydań, ale wynika to tylko z faktu że dane nie były pobierane pod koniec roku. Ostatni wynik jest z 2021-04-10.



- **loudness w pliku tracks.jsonl jest ujemną wartością**
  np. {"id": "63kd4m3VFxcJjPVVtbVNAu", "name": "Hello, Dolly!", "popularity": 53, "duration_ms": 147000, "explicit": 0, "id_artist": "19eLuQmk9aCobbVDHc6eek", "release_date": "1964-10-25", "danceability": 0.0, "energy": 0.405, "key": 0, "loudness": -9.935, "speechiness": 0.0, "acousticness": 0.842, "instrumentalness": 0.00114, "liveness": 0.198, "valence": 0.0, "tempo": 0.0}

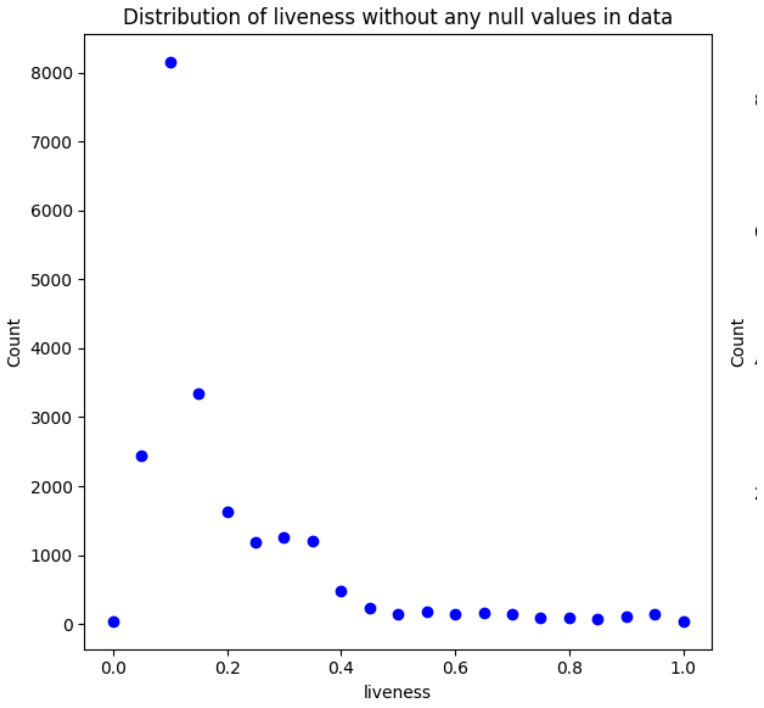
- **w pliku users.jsonl występują znaki specjalne we wszystkich kolumnach z tekstem**

  np. {"user_id": 564, "name": "Andrzej \u0141ukowiak", "city": "Warszawa", "street": "al. Spokojna 13/83", "favourite_genres": ["reggaeton", "latin", "k-pop boy group"], "premium_user": false}

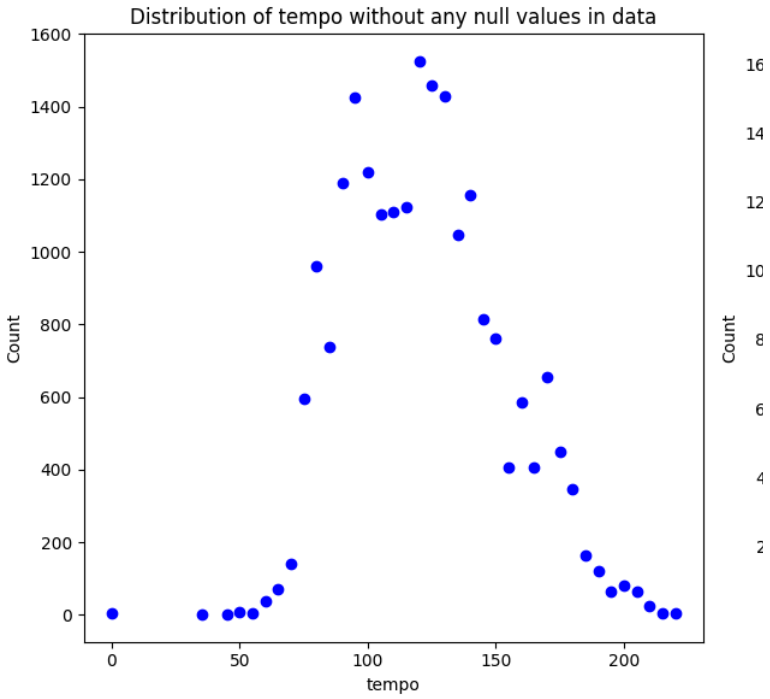
- **Prosimy przeanalizować wyniki niektórych kolumn dla pliku tracks.jsonl**

  Poniżej podaje problemy i pytania w sprawie danych razem z rozkładem wyników

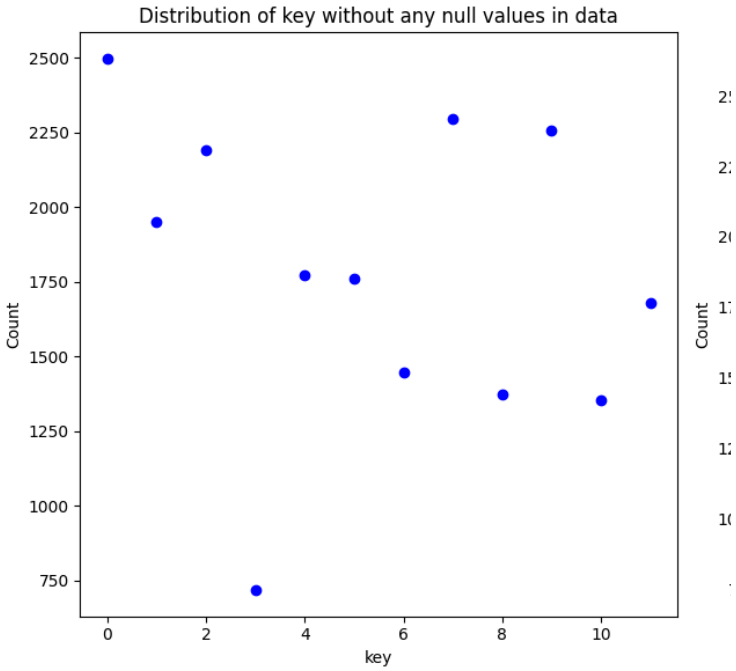
  - W “liveness” widać znaczący odchył dla wartości z przedziału [0.2; 0.4] (rozkład liveness z zaokrągleniem z precyzją do dwóch miejsc po przecinku, skalowanym do wielokrotności 5)



- czy “tempo” na pewno jest dobrze podane w danych?
  (rozkład tempa zaokrąglonego do najbliższej wielokrotności 5)



- Czy mógłby pan powiedzieć w jaki sposób dobierana jest wartość klucza? Czyte dane na pewno są prawidłowo przydzielane?
  (rozkład wartości klucza)



Dla pewności, byłaby możliwość otrzymania informacji o przewidywanych zakresach danych w kolumnach?

- ## session_id nie powinno być unikalną wartością?

  Jeśli nie, to dlaczego session.jsonl nie posiada unikalnego klucza id?

  przykładowe dane z pliku sessions.jsonl:

  {"session_id": 124, "timestamp": "2023-10-25T10:13:53.258944", "user_id": 551, "track_id": "1Duym1IVQurgKHHSqpOWhY", "event_type": "play"}

  {"session_id": 124, "timestamp": "2023-10-25T10:15:27.065944", "user_id": 551, "track_id": null, "event_type": "skip"}

  {"session_id": 124, "timestamp": "2023-10-25T10:15:32.065944", "user_id": 551, "track_id": null, "event_type": "advertisment"}

  {"session_id": 124, "timestamp": "2023-10-25T10:15:32.065944", "user_id": 551, "track_id": "6XFduBZIOMyOV8sCcHuYzb", "event_type": "play"}

  {"session_id": 124, "timestamp": "2023-10-25T10:16:19.994944", "user_id": 551, "track_id": "6XFduBZIOMyOV8sCcHuYzb", "event_type": "like"}

  {"session_id": 124, "timestamp": "2023-10-25T10:17:47.679944", "user_id": 551, "track_id": null, "event_type": "skip"}

  Jeśli istnieje dodatkowa tablica z sesjami, gdzie session_id jest unikalną wartością, która na przykład mówi kiedy się rozpoczęła lub zakończyłą dana sesja, poprosilibyśmy do niej dostęp

- ## linie w users.jsonl mogą być zakończone kolumną id

  {"user_id": 578, "name": "Olgierd Drgas", "city": "Szczecin", "street": "ul. Towarowa 86", "favourite_genres": ["rock en espanol", "hip hop", "electropop"], "premium_user": false, "id": -1}