IUM 2023Z

Jan Filipecki (305969), Jakub Niezabitowski (304082)

Temat:

“Fajnie byłoby rozszerzyć nasz serwis o generowanie popularnych playlist – zestawów pasujących do siebie utworów, których słuchaniem zainteresowane będzie wiele osób”

Szczegółowy opis tematu:

W ramach projektu wcielamy się w rolę analityka pracującego dla portalu „Pozytywka” - serwisu muzycznego, który swoim użytkownikom pozwala na odtwarzanie ulubionych utworów online. Praca na tym stanowisku nie jest łatwa – zadanie dostajemy w formie enigmatycznego opisu i to do nas należy doprecyzowanie szczegółów tak, aby dało się je zrealizować. To oczywiście wymaga zrozumienia problemu, przeanalizowania danych, czasami negocjacji z szefostwem. Same modele musimy skonstruować tak, aby gotowe były do wdrożenia produkcyjnego – pamiętając, że w przyszłości będą pojawiać się kolejne ich wersje, z którymi będziemy eksperymentować.

Jak każda szanująca się firma internetowa, Pozytywka zbiera dane dotyczące swojej działalności – są to (analitycy mogą wnioskować o dostęp do tych informacji na potrzeby realizacji zadania):

- lista dostępnych artystów i utworów muzycznych,
- baza użytkowników,
- historia sesji użytkowników,
- techniczne informacje dot. poziomu cache dla poszczególnych utworów.

Problem biznesowy:

Opis kontekstu:

Serwis muzyczny “Pozytywka” umożliwia użytkownikom słuchanie swoich ulubionych utworów muzycznych. Firma chce rozszerzyć swoją usługę o generowanie playlist, zawierających zestaw pasujących do siebie utworów, co potencjalnie ma zwiększyć zainteresowanie serwisem.

Zadanie biznesowe:

Celem biznesowym jest zwiększenie sumarycznej liczby odsłuchań piosenek znajdujących się na playlistach, co potencjalnie przyczyni się do zwiększenia bazy użytkowników serwisu „Pozytywka”. Chcemy stworzyć model, który pozwoli nam na wygenerowanie zbiorów utworów, które są ze sobą spójne na przykład pod względem gatunku, tempa lub dynamiki.

Biznesowe kryterium sukcesu:

Model powinien co tydzień generować nową playlistę utworów, która docelowo ma generować więcej łącznych odsłuchań niż zbiór tych samych utworów w poprzedzającym tygodniu(Eksperyment A/B).

Zadanie modelowania:

Założenia:

- Czas trwania playlisty to maksymalnie jedna godzina
- Playlisty generowane mają być co tydzień
- Playlisty z poprzedzającego tygodnia zastępowane są nowo-wygenerowanymi playlistami
- Jeden utwór może występować w wielu playlistach
- Model generuje od 5 do 15 playlist

Opis zadań modelowania:

Model bazowy:

Pierwszym krokiem jest wyznaczenie *n* najczęściej powtarzających się wśród artystów gatunków muzycznych (*n* - docelowa liczba generowanych playlist).
Następnie do utworów dodajemy atrybut, będący listą gatunków, reprezentowanych przez ich autorów.
Model bazowy sprowadza się do grupowania utworów według najpopularniejszych gatunków. Jeśli dany utwór posiada w liście gatunków jeden z najczęściej powtarzających się, trafia do zbioru utworów tego gatunku. Następnie utwory w zbiorach sortowane są po atrybucie *popularity*, a ze zbioru wybierane są najpopularniejsze utwory, których łączny czas trwania nie przekracza godziny. To są nasze docelowe playlisty.
W przypadku niewystarczającej liczby utwór do spełnienia wymaganej godziny trwania, playlista uzupełniania jest najpopularniejszymi utworami, niezależnie od gatunku.

Model docelowy:

W przypadku modelu docelowego chcemy poprawić dwa mankamenty modelu bazowego. Pierwszym z nich jest sam atrybut *popularity*, który jak dowiedzieliśmy się od „Pozytywki" jest wartością globalną, co oznacza, że nie jest otrzymywany na podstawie wewnętrznych danych firmy, a raczej wynikiem agregacji popularności w wielu serwisach streamingowych. Może to zaburzać wyniki naszego modelu, dlatego dla pewności wprowadzić chcemy dodatkowy wewnętrzny atrybut

popularności, który wyznaczany będzie na podstawie ilości akcji odtworzeń i pominięć w bazie sesji użytkowników, co nie tylko da lepszy obraz wewnętrznych danych firmy, ale również będzie aktualizował dane o wiele szybciej.
Drugim problemem jest fakt, że gatunek utworów nie musi bezpośrednio reprezentować utworów, gdyż jest to cecha artystów, co jest jedną z podstawowych wad bazowego modelu. Kolejnym mankamentem jest zbytnia ogólność terminu „gatunek" co sprawia, że utwory z tego samego gatunku potencjalnie mogą do siebie nie pasować. Zaradzić temu chcemy wykorzystując inne atrybuty utworów.
Gdy dane zostały już rozszerzone o niezbędne atrybuty, możemy przystąpić do pierwszego etapu modelu docelowego - klasteryzacji. Klasteryzację chcemy przeprowadzić korzystając z metody DBSCAN. Pozwoli nam ona pogrupować utwory o podobnych cechach w dokładniejszy sposób niż w modelu bazowym. Warto zaznaczyć, że klastry budowane będą bez wykorzystania atrybutów popularności.
Następnie, podobnie jak w modelu bazowym, sortujemy utwory w klastrach. Tym razem robimy to jednak na bazie sumy odpowiednio przeskalowanych wartości dwóch atrybutów - atrybutu *popularity* dostarczonego przez „Pozytywkę" oraz naszego wewnętrznego atrybutu popularności, wyznaczonego na podstawie danych sesji. Po posortowaniu utworów, wybieramy w każdym klastrze wszystkie najpopularniejsze utwory, których łączny czas nie przekracza godziny. Parametry metody klastrowania dobierane będą w taki sposób, żeby suma długości utworów w klastrach zawsze była większa od godziny.

Analityczne kryterium sukcesu:

- Jakość wygenerowanych zbiorów chcemy mierzyć za pomocą metryki Silhouette, która mierzy podobieństwo utworów do innych w danej playliście, czyli swoistą spójność playlist, oraz podobieństwo w porównaniu do utworów w innych playlistach, czyli czy zbiory odpowiednio separują różne utwory od siebie. Metryka będzie mierzona bez wykorzystania atrybutów popularności oraz gatunków autorów. Chcemy zauważyć co najmniej 10% poprawę jakości w tej metryce.

$$\frac{SC_D}{SC_B} \geq 1.1$$

  SC_D - współczynnik Silhouette dla playlist wygenerowanych przez model docelowy

  SC_B - współczynnik Silhouette dla playlist wygenerowanych przez model bazowy

- Suma liczby odsłuchań utworów w tygodniu po umieszczeniu ich na playliście jest większa o co najmniej 10%, niż w tygodniu poprzedzającym

$$\frac{\sum_{i=0}^{|X|} x_i}{\sum_{j=0}^{|Y|} y_j} \geq 1.1$$

X - zbiór liczb odsłuchań utworów w tygodniu, kiedy były na playliście

Y - zbiór liczb odsłuchań utworów w tygodniu przed dodaniem do playlisty

Wstępna propozycja narzędzi:

- **Python** - język programowania
- **SQLite3** - biblioteka Pythona, która umożliwia przechowywanie danych firmy w formie jednego pliku bazodanowego
- **Pandas** - biblioteka Pythona, która umożliwia łatwe wczytywanie i manipulowanie danymi; Bardzo przydatna podczas analizy danych
- **Plotly** - biblioteka Pythona, wykorzystywana do wizualizacji i analizy danych
- **Scikit-learn (sklearn)** - biblioteka Pythona, która umożliwia wykonanie klasteryzacji DBSCAN
- **Numpy** - biblioteka Pythona, która ułatwi nam wyliczanie analitycznego kryterium sukcesu naszych modeli

Analiza danych:

Analiza danych znajduje się w załączonym do dokumentacji pliku .ipynb.