DS-GA 1004 BIG DATA

GROUP 42

Kritik Seth
NYU
Center for Data Science
New York, NY, USA
kls8193@nyu.edu

Saahil Jain
NYU CDS
Center for Data Science
New York, NY, USA
sbj7913@nyu.edu

DATA PREPROCESSING

1. Joined interactions and tracks dataset on “recording_msid”
2. Imputed the null values in “recording_mbid” with corresponding value with “recording_msid”
3. Created a new dataframe with columns- “user_id”, “recording_mbid”, “timestamp”.

TRAIN VALIDATION SPLIT

We used the following two methods to split the data into train and validation

Method - 1

1. A new column named "row_number" was created, which assigned a unique number starting from 0 to each interaction of a user, ordered by timestamp from oldest to most recent.
2. A new column named "max_row_number" was created for each user, containing the value of the last row in the "row_number" column.
3. A new column named "split_index" was created by multiplying "max_row_number" with a fraction indicating the ratio of "train_data" to "all_data".
4. The train dataset was created by selecting all rows for every "user_id" where "row_number" was less than "split_index".
5. The validation dataset was created by selecting all rows for every "user_id" where "row_number" was greater than or equal to "split_index".

Method - 2

1. A new column named "row_number" was created, which assigned unique numbers to each interaction randomly (starting from 0) for each user. [Repeat steps 2-5]

Therefore, the train dataset was split into train and validation with a 70/30 ratio by creating a new column for the row number, determining the maximum row number for each user, calculating a split index, and then creating separate datasets based on the split index for each user.

BASELINE MODEL

We tried multiple popularity baseline models such as average play per user, total plays per user, through total unique interactions, and time-based but the best popularity baseline in terms of highest MAP was time-based popularity baseline model.

Dataset was grouped by “recording_mbid” and a time decay function was calculated and applied as follows.

$$w(t) = e^{\left(\frac{-\lambda \cdot t}{86400}\right)}$$

here t is an estimate of the amount of time music has been available, λ is the decay parameter and 86400 is the number of seconds in a day. This estimate of t was calculated by grouping on “recording_mbid” and subtracting the lowest timestamp for each music from the highest. We multiplied this time decay function with the number of unique users who interacted with the song to calculate our “popularity” parameter. The dataset was then sorted on “popularity” (highest to lowest) and top 100 rows from “recording_mbid” were selected to be our recommendations.

Our intuition for this is that if a song that was released yesterday becomes as popular (has the same number of interactions) as a song that has been there for a year, it is probably what is more popular in recent times and would be a better recommendation.

Time-based approach was selected and hyperparameter tuning was performed. Fourteen values of λ were considered ranging from 1e-07 to 0.5. These values were looped over and for each value of λ MAP on Train and Validation dataset was calculated. The value of λ the highest MAP on the Validation dataset was selected. Best λ = 1e-05. This was observed on the data split randomly.

PERFORMANCE

Performance metric that was used to evaluate the performance of our baseline model was Mean Average Precision.

Results:

Dataset	Decay Rate (λ)	MAP
Train	1e-05	0.00031802
Validation	1e-05	0.00028663