APPLIED DATA SCIENCE

16971-A

FINAL PROJECT REPORT

SONG RECOMMENDATION ENGINE

TEAM E:
Mandar Limaye
Padmil Khandelwal
Steffie Rego

Table of Contents

1. Executive Summary

The project aims at building a song recommendation engine for users given their listening history. We aim to understand the user's song preference using various features of the songs previously listened to by the user. Music streaming services could benefit from this approach as it gives them a platform to understand user behavior and recommend songs accordingly. Additionally, record labels and artists can perform targeted marketing for their newly released songs using data of users provided by the recommendation engine. Our project also aims at solving the 'cold-start problem' a prevalent issue in this space, by recommending new songs to the users given their previously listened songs.

One of the key aspects of our project is that it takes into consideration the lyrics of the songs to understand whether similarity in lyrics plays a role in a user's song selection along with other song features. In order to achieve this, an NLP technique called topic modeling was performed that clustered the words into multiple groups and assigned topics to each group. Using these topics, the percentage contribution of each topic in a particular song listened was determined. The project aims to use these lyrics contributed features along with traditional song features and artist features to build a user profile for every user. This profile is used to understand user's music taste and determine his likelihood to listen to a new song. This, in turn, forms the basis of our recommendation engine.

The project uses songs, lyrics, and user metadata from the Million Song Dataset. Corresponding train and test datasets were build using multiple iterative steps to ensure same users and different songs in both these sets. Feature engineering was performed using multiple aggregated statistics of user-song interaction and user profiles generated using past listening habits to determine the optimal features for classification. The likelihood of a user listening to a new new song was then analyzed using multiple classification models. Based on the structure of our experiment and evaluation matrix, we observed that the Decision Tree Classifier performed better than the rest of the models.

2. Motivation

In today's digital era, we have various music streaming platforms like Spotify, SoundCloud and Apple Music. These platforms have a total user base of 59.76m users and have generated a revenue of US$5,300m, 79.8% of the total music industry revenue in 2019 [1]. All these streaming services have recommendation engines with their own unique predictive and classification algorithms.

Personalized Music Taste

- Spotify alone accounts for 30m unique songs with thousands of playlists. In 2018, Spotify confirmed that almost 20,000 new tracks are added per day to the platform.
- Currently, there are 400,000+ artists on Spotify and 45% of them do not belong to any major labels [2]. Only certain percentage of these songs make their way to the 8000 influential playlists on Spotify
- With the millennial generation developing personalized music taste, it becomes extremely important for businesses to release similar music on their platform.
- Despite the demand for different music being high, a large number of songs released on the platform do not reach their intended audience.

Understanding User Inclination

- The taste of music constantly changes with time and experiences. With users responding to different songs, the concept of genre-blending came into the music industry. Therefore, it is utmost necessary for the platforms to understand the user profile.

- In such scenarios, it becomes critical for businesses to understand the user's willingness to listen to a new song. Most of the recommendation engines use a collaborative approach by leveraging popular listening patterns of other users to recommend new songs.
- However, the power of predictive modeling and natural language processing can empower these companies to comprehend the similarity between lyrics and other features of different songs. It can help them to generate better personalized playlist after newly released songs are added to the platform.

This content-based filtering approach in the recommendation engine systems has been a black box. Therefore, we wanted to understand the basic functioning of these systems and devise ways to improve it.

3. Objective & Goals

The objective of the project was to build a model to understand the factors that influence the choice of a user in song selection, given the users' past listening habits. Subsequently, the goal of this project was to study recommendation engines and understand if lyrics play a role in determining users' choices.

Output: Binary output classifying weather a user will listen to newly released song

Hypotheses to Test:

- The user's song selection can be predicted using the features of the songs previously mapped to the user.
- The high popularity of the artist corresponds to a higher probability of the song being selected by the user.
- The similarity of song lyrics is likely to contribute more to the selection of songs by the user.

4. Stakeholders

Our major stakeholders would be the music streaming platforms. Additionally, record labels and artists can use this to target their songs to selected users.

5. Use Case Scenario

Collaborative filtering suffers from the "cold start problem". It is impossible to recommend a new track to any existing users without the song gaining userbase on the platform. Our product aims to solve this problem using content-based filtering which uses song features and their lyrics. The firm (typically the music streaming platform) will input user listening habits and the new songs not visible to the user. Our product will generate which songs can be recommended to these users based on their listening habits, increasing the listening time per user for the firm.

6. Dataset

6.1 Metadata

Tracks Metadata: 1,000,000 tracks, 999,056 songs, 44,745 artists
Lyrics Metadata: 237,662 tracks, 5000 unique words, 107.76 average words per song

User data (Before Processing):
TRAIN: 100000 users, 157251 songs
TEST: 100000 users, 159717 songs

6.2 Selection of Subset Data

In order to create a uniform subset of train and test datasets for our classifications problem such that they both have the same users but different songs (No Song Overlap per User), we performed the following steps:

1. Filter songs for which lyrics are available and
   TRAIN: 98801 Users
   TEST: 98870 Users

2. Find 400 Popular Songs in TRAIN and TEST
3. Randomly select 200 songs in popular TRAIN that do not exist in popular TEST
4. Filter users that have only listened to > 10 songs
5. Filter users such that same users exist in TEST and TRAIN

Our base assumption in the training dataset is that the user has been exposed to all the songs in the dataset. For our problem statement we also assume that the user's music taste remains constant. The final dataset is created such that, every user is mapped to all songs from dataset and given a status 0/1 depending on whether the user has listened to the song or not.

Final Dataset:
TRAIN: 31 Users, 93 Songs, 369 user records with status 1, 2514 user records with status 0
TEST: 31 Users, 80 Songs, 277 user records with status 1, 2203 user records with status 0

7. Project Plan (Flowchart)

The following flowchart explains the working of the project

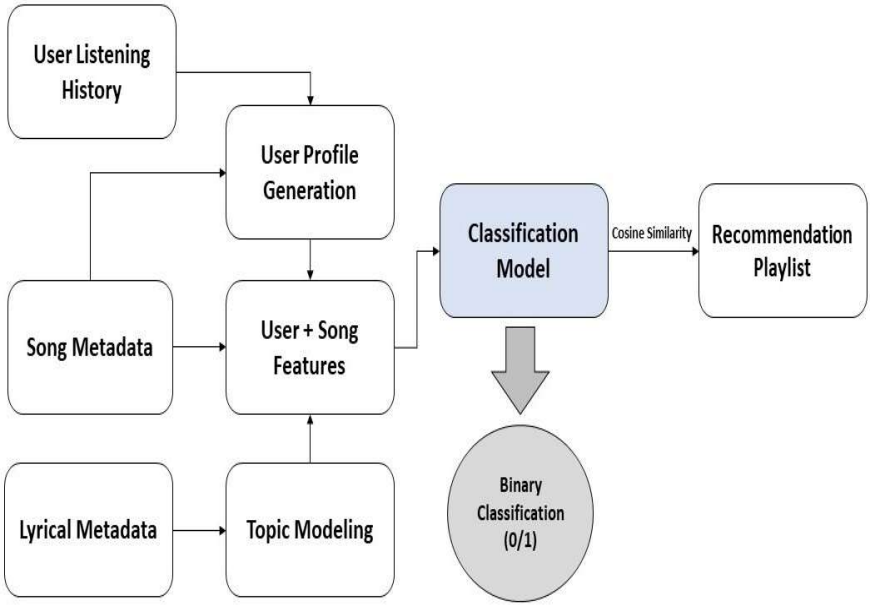

Figure 1: Flowchart indicating the internal working of the project

8. Exploratory Analysis

Following are some of the exploratory analysis we conducted on the dataset in order to understand the distribution of data after pre-processing:

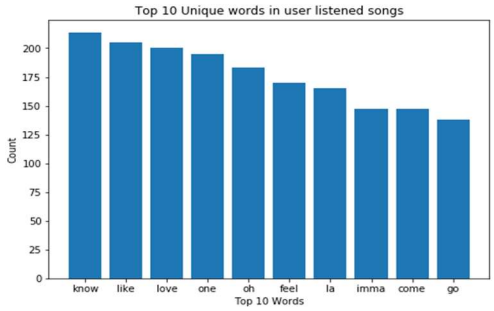

Figure 2 : Top 10 unique words in song lyrics listened to by users

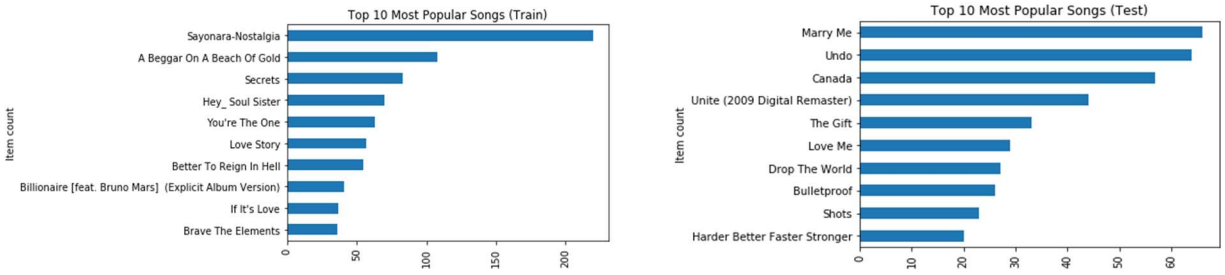

Figure 3: Top 10 Most popular Songs in TRAIN & TEST dataset

9. Feature Engineering

9.1 Topic Modelling

To find out the topics per track and the percentage contribution of each topic in a particular track we performed topic modelling using Latent Dirichlet Allocation. [3]

- Data Pre-processing:

1. Stop Words and Punctuation Removal:
   These words are removed as they do not add value to the context of the document. This was done using NLTK library.

2. Lemmatization and Stemming:
   Breaking down similar extensions of one word to root form using WordNet lemmatizer and Porter Stemmer.

3. Created Bag of Words and document for each track using the words in each track and the frequency of their Occurrence (Figure 4)

	track_id	bagWord	doc
0	TRAALAH128E078234A	[know, know, know, know, one, one, one, one, o…	know know know know one one one one one never …

Figure 4: Snippet of dataframe containing Bag of Words and Document for a particular track

4. Created Dictionary and Corpus needed for topic modelling:
   The two main inputs to the LDA topic model are the dictionary(id2word) and the corpus. The dictionary is created using corpora from the **gensim** library of python. This dictionary is used to create a corpus which is a term document frequency matrix using doc2bow.

- Building the Topic model:

1. Using the LdaModel from the **gensim** library of python to train the LDA model and determine the distribution of words per topic. We choose **10 topics** as input to the model.

```
[(0,
  '0.030*"like" + 0.023*"come" + 0.014*"got" + 0.013*"love" + 0.012*"get" + 0.010*"go" + 0.009*"think" + 0.009*'
 (1,
  '0.026*"keep" + 0.019*"got" + 0.017*"man" + 0.015*"babi" + 0.013*"know" + 0.012*"time" + 0.012*"wait" + 0.012
```

Figure 5: Snippet of topics and corresponding words in the topic

2. Compute model Perplexity:
   Model perplexity is a measure to judge how good a given topic model is. The lower the score the better the model performance.
   Final Perplexity = -6.32

- Determining topic contribution per track:

  In order to build a song feature profile for each track, we determined the topic contribution per track. i.e. the percentage contribution of each of the 10 topics per track(Figure 6)

	Topic_0	Topic_1	Topic_2	Topic_3	Topic_4	Topic_5	Topic_6	Topic_7	Topic_8	Topic_9	Text	track_id
0	0.0000	0.0000	0.7917	0.0	0.0000	0.1129	0.0241	0.0000	0.0000	0.0691	know know know know one one one one one never ...	TRAALAH128E078234A
1	0.0000	0.0000	0.0000	0.0	0.0000	0.0000	0.0000	0.0000	0.9974	0.0000	like like come come come one one one one one o...	TRABFDT12903CADD73

Figure 6: Snippet of dataframe with topic contribution for a track

9.2 User Profile

To create a user profile for every user in the dataset we used the following formula:

$$(UserFeature)_i = \sum[(Songfeature)_i \times WeightedListenCount]$$

The user feature is assigned to every record corresponding to that user. This is repeated for every Song feature such that Number of song features = Number of user features.

9.3 Final Dataset Features:

The following are the features of our final dataset after aggregating measures and performing Natural Language Processing techniques

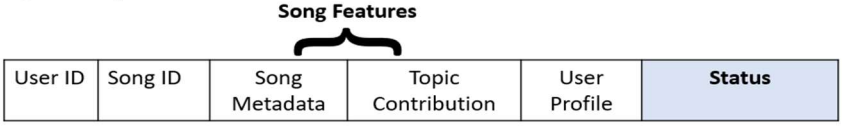

Figure 7: Representation of final features in TRAIN and TEST datasets

10. Modelling and Performance:

10.1 Evaluation Metric Selection

As explained earlier that the model is imbalanced with around 90% data points with class 0 and 10% with class 1. Thus, instead of accuracy we have selected AUC and F1 as our evaluation metrics. But we will also give a priority to TPR if model outputs are quite similar. From our predictions, to create the final recommendation for

a user we are using top five cosine similarity to suggest songs to the user. In order to further assess our final recommendation, we are using an absolute score out of 5 to determine the performance.

10.2 Approach 1: Cosine Similarity as a Feature

Initially we added an input feature that incorporated the lyrical similarity between the songs of the user by vectorizing the textual feature using TF-IDF. We calculated the sum of weighted cosine similarity between a particular song and all the previous songs listened by that specific user. This feature gave us a similarity score based on the user's listening habits which was used as input along with other song features. After trying five models along with 10-fold cross CV our model performances were the following:

Model	TPR(Recall)	FPR	Accuracy	Precision	F1
Support Vector	0.006	0.009	0.765	0.167	0.012
Naive Bayes	0.476	0.444	0.538	0.242	0.321
Decision Tree	0.268	0.224	0.660	0.264	0.266
Random Forest	0.287	0.215	0.671	0.285	0.286
K-Nearest Neighbours	0.165	0.126	0.712	0.281	0.208

Figure 8: Performance of all models using Cosine Similarity Approach

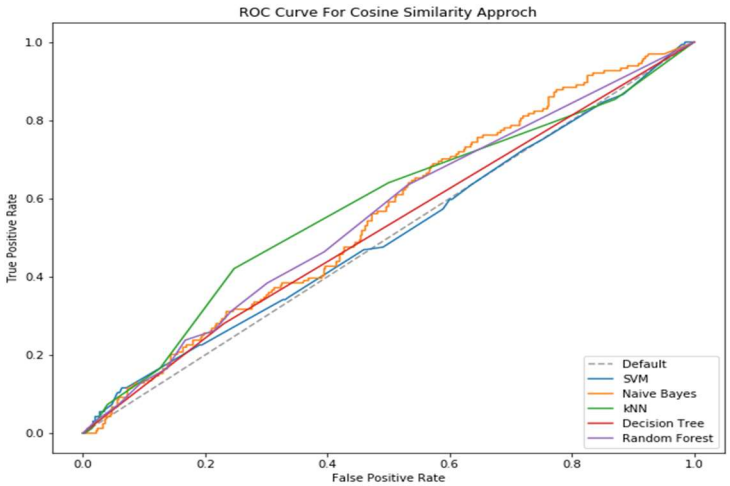

Figure 9: ROC for different models using Cosine Similarity Approach

Observation: The model was not performing as per our expectation as you can see in the figure 9. Next, to improve our performance we decided to engineer user features by user profile generation method as explained in section 9. The final input space used is given in Figure 7.

10.3 Approach 2: User Profile Generation

We used six classification models to identify the best performing model. Figure 10 shows the confusion matrix for all the classification models. We used GridSearchCV for hyperparameter tuning to select the optimal parameters for every classification model. We also used 10-fold cross validation to train all the models with multiple parameters. We have also made sure that in each cross validation, the user profile was generated based on the specific training data. This was done to make sure that user profile does not include songs in the test fold and there is no data leakage.

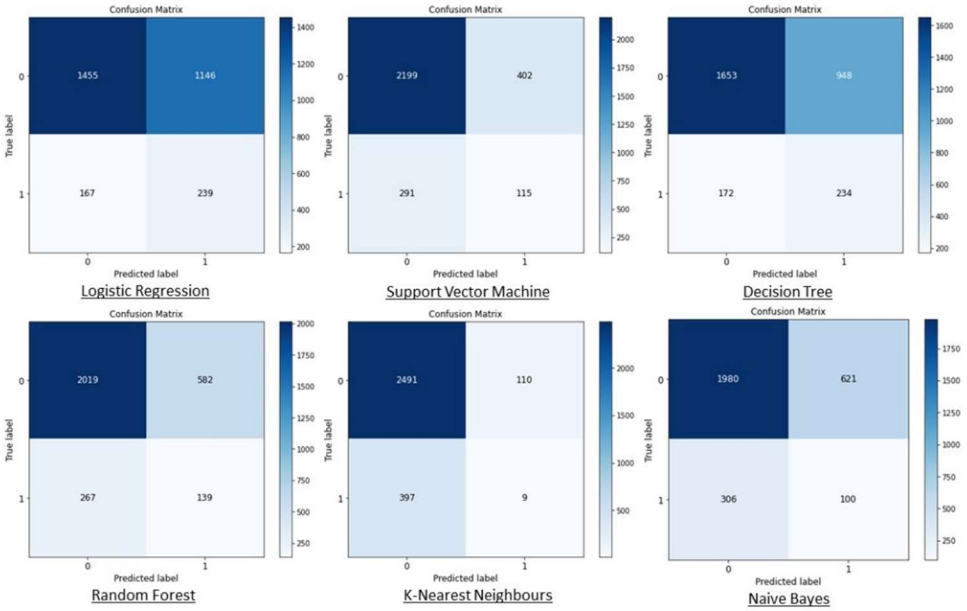

Figure 10: Confusion Matrix of all models using User Profile Approach

Model	TPR (Recall)	FPR	Accuracy	Precision	F1	AUC Score
Logistic Regression	0.589	0.441	0.563	0.173	0.267	0.574
Support Vector	0.283	0.155	0.770	0.222	0.247	0.563
Naive Bayes	0.246	0.239	0.692	0.139	0.178	0.504
Decision Tree	0.576	0.365	0.628	0.198	0.289	0.605
Random Forest	0.342	0.224	0.718	0.193	0.250	0.562
K-Nearest Neighbours	0.022	0.042	0.831	0.076	0.034	0.490

Figure 11: Performance of all models using User Profile Approach

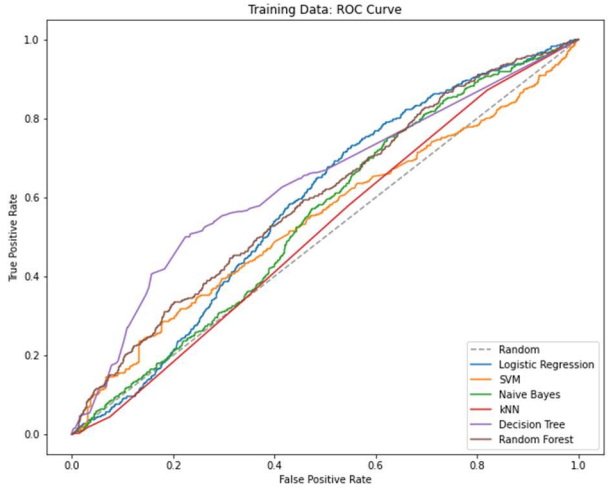

Figure 12: ROC for different models using User Profile Approach

Observation: As you can see in Figure 11 & 12, we saw that our logistic regression and decision tree models were performing the best. After comparing the AUC & F1 score, we concluded that **Decision Tree** is the optimal model for our problem statement.

11. Results and Interpretation

11.1 Model Performance on Test Data

We used the trained decision tree classifier to predict the likelihood of the user listening to a song in the test data. During this step, the user profile was generated for each user using all the songs that user has listened to in the train data (There is no overlap of songs between train and test)

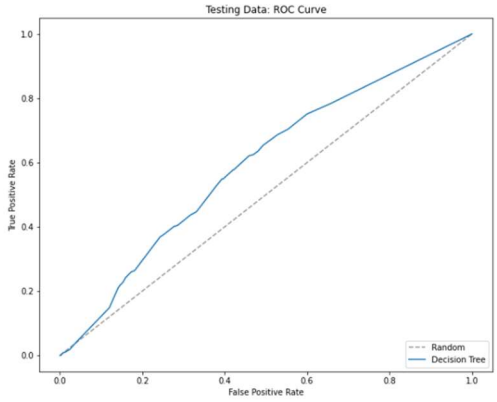

Figure 13: ROC of Test Data using User Profile Approach

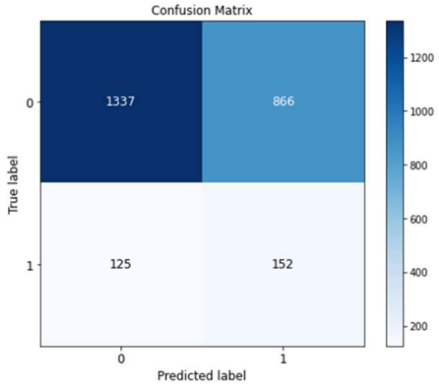

Figure 14: Confusion Matrix on Test Data

TPR (Recall)	FPR	Accuracy	Precision	F1	AUC Score
0.549	0.393	0.601	0.1493	0.2347	0.578

Figure 15: Test Data Performance

As you can see in the Figure 13 & 15, the model performed well on an unknown test dataset. Further to test the hypothesis of our experiment, we calculated the feature importance of all the features in the classification model.

11.2 Hypothesis Results

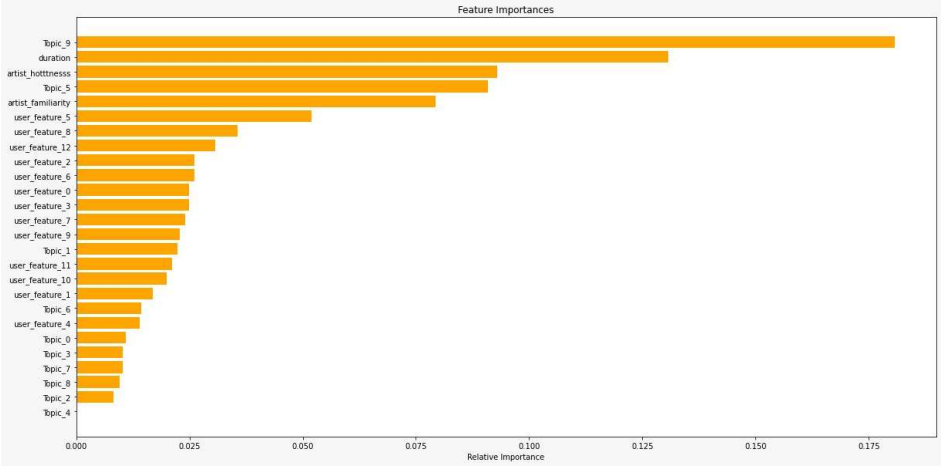

Figure 16: Feature Importance

We successfully conducted the experiment that showed user's song selection can be predicted using the features of the songs previously mapped to the user On the other hand, as you can see in the Figure 16, contribution of some topics such as Topic_9 and Topic_5 do play an important role to determine the user's likelihood to listen to a song. Along with lyrics, artist hotness and familiarity also play a huge role in user's song selection.

11.3 Final User Recommendation

From our predicted output labels, we created a list of recommendations for a user. To rank this as top 5 we used cosine similarity between user profile vector and song vector.

```
Generating Recommendations for User  cff21116daa6af7719ed41e4b540feeaf5f31803
--------------------------------------------
Harder Better Faster Stronger by Daft Punk
The Gift by Angels and Airwaves
My Name Is by Eminem
Drop The World by Lil Wayne / Eminem
Victoria (LP Version) by Old 97's
--------------------------------------------
User Listens to  3 out of  5 songs
********************************************
```

Figure17: Recommendation for a user

Result: On average if we recommend 5 songs to every user, we were able to predict more than 2 (2.4) of them correctly. Figure 17 shows a snapshot of recommendation for one of the users in the dataset.

12. Stakeholder Utility

The model will return a binary output for each user about their likelihood to listen to a particular song given their past listening history. This model will then be used to

13. Challenges

The following challenges were faced while building the recommendation engine:

1. Determining the optimal number of topics for LDA (topic modelling)
2. Selection of subset for TRAIN and TEST to balance the bias of 0/1 distribution
3. Feature engineering to find out the appropriate features for classification
4. Selection of optimal thresholds on operating curve

14. Risks & Mitigation

1. Model Overfitting:
To mitigate the problem of overfitting in our classification model, we performed hyperparameter tuning to reduce the bias caused due to large amount to dimensionality ratio in the dataset.

2. Imbalance output class:
As the average song count per user was 12, which is quite low compared to the total songs in the dataset, we selected AUC and F1 as our evaluation metrics for classification instead of accuracy.

3. Lack of user feature:
Due to lack of user features in the original dataset, we created user profile by using feature engineering as explained in section 9.2

15. Future Work

We see that there is lot of potential to continue the work in this domain. One of the biggest weak links in the experiment was lack of user features. However, if features like user demography were available, then more holistic view of user can be taken into consideration while recommending songs.

On the other hand, it would be interesting to build a collaborative filtering engine using the timeseries data of user's listening history. It is a well-known fact that user's music taste changes with time. Therefore, the recommendation accuracy can be increased if we incorporate a feature that can account for change in music taste

References:

[1].“Music Streaming - United States: Statista Market Forecast.” *Statista*, www.statista.com/outlook/209/109/music-streaming/united-states.

[2]. Ingham, Tim. “Nearly 40,000 Tracks Are Now Being Added to Spotify Every Single Day.” *Music Business Worldwide*, 29 Apr. 2019, www.musicbusinessworldwide.com/nearly-40000-tracks-are-now-being-added-to-spotify-every-single-day/.

[3]. Prabhakaran, Selva. “Topic Modeling with Gensim (Python).” *Machine Learning Plus*, https://www.machinelearningplus.com/nlp/topic-modeling-gensim-python/#16buildingldamalletmodel.