IST 718 Final Project – Group 3
March 29, 2020

Music Recommendation Analysis

Title page (4 points) with summary of your findings and your recommendations.

Summary of Findings

The music streaming market is highly competitive with little differentiation beyond the core product (songs and artists). This results in rapid loss of market share and business viability in ancillary areas-primarily song recommendation algorithms to encourage streaming and product consumption beyond the intended consumer time frames. Industry analysis by Pandora and MusicWatch found the accuracy of song recommendations is a primary differentiator.[i]

The key attribute that drives prediction accuracy is the length of a user's membership measured in days, which infers a larger amount of behavior and preference data for a particular user. Additional analysis showed that location, both country and city, and age of users impacted accuracy of users. Prediction error was primarily due to false positives where a song was recommended, but the user did not listen to more than once.

The XGBoost algorithm was the most performant of explored models with a weighted precision and recall of 69%. This model demonstrated similar error predicting a user would like a song when only afforded a single listen. Additional analysis with association rules indicated some genres showed Country music had the highest predictability. Decision trees were least helpful.

Recommendations

As user data was a critical factor in prediction accuracy, new data should be integrated into analysis. While the current model does not exceed 75% accuracy, continuous integration and continuous development (CI/CD) will improve performance over time. Recommendations include:

1. Deploy the existing model and trigger retraining based on future data integration
2. Perform A/B testing on discrete and ensemble methods to compare performance
3. Determine additional information that can be integrated into the model (e.g. song rating, sentiment analysis in discussion forums)
4. Develop genre specific models to account for variations in user behaviors
5. Develop region specific models to account for variations in user behaviors

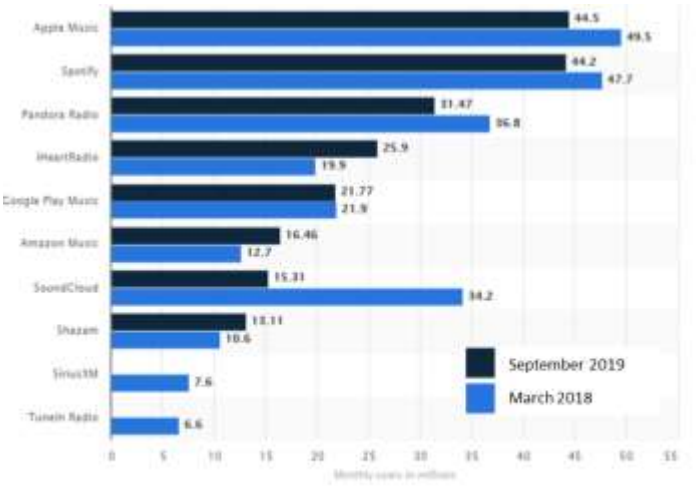
Continued experimentation and parameter tuning with the existing models and data should also be integrated into the song recommendations strategy.

Contents

Specification

Revenue from streaming services reached $8.8 billion in 2019, and now accounts for 79.5% of all recorded music revenues.[ii] In this same timeframe, the top 5 music streaming service providers held 79 percent market share.[iii] From March 2018 to September 2019 the streaming service providers Soundcloud, Shazam, SiriusXM and TuneIn Radio lost significant market share, and potentially long term viability.

As the top providers compete for a larger share of revenues it is critical they "*must get playlists right*" through relevant recommendations based in each users' preferences.[iv] Improving the relevance of recommendations requires a considerable amount of data to identify patterns and preferences of listeners.



Top U.S Music streaming services (millions of users)

Problem Statement

The music streaming market is highly competitive where a lack differentiation beyond the core product (songs) results in rapid loss of market share and business viability. Companies that cannot leverage their user and service data to provide highly relevant listener song recommendations risk irrelevance in as little as eighteen months.

Hypothesis

Analysis of data about the listener, songs and services enables a music streaming service to predict if a given user will listen to a song more than once.

Data

The data was sourced from the music recommendation system Kaggle competition hosted by *The ACM International Conference on Web Search and Data Mining* (WSDM) based on a donated dataset from KKBOX. The competition can be found at: https://www.kaggle.com/c/kkbox-music-recommendation-challenge/data.

The data is read in from .7z file formats and is saved into csv files for analysis:

```
################################################
####             Set up workspace             ####
################################################
#### Setup Workspace, import data and set key variables ####
#kaggle competitions download -c kkbox-music-recommendation-challenge

def setupEnv (dirVar):
    Archive('data/train.csv.7z').extractall(dirVar)
    Archive('data/test.csv.7z').extractall(dirVar)
    Archive('data/song_extra_info.csv.7z').extractall(dirVar)
    Archive('data/songs.csv.7z').extractall(dirVar)
    Archive('data/members.csv.7z').extractall(dirVar)
    df_songs = pd.read_csv('data/songs.csv', encoding = 'utf-8', na_values=['NaN'])
    df_song_extra_info = pd.read_csv('data/song_extra_info.csv', encoding = 'utf-8', na_values=['NaN'])
    df_members = pd.read_csv('data/members.csv', encoding = 'utf-8', na_values=['NaN'])
    print(df_song_extra_info.shape)
    df_train = pd.read_csv('data/train.csv', encoding = 'utf-8', na_values=['NaN'])
    df_test = pd.read_csv('data/test.csv', encoding = 'utf-8', na_values=['NaN'])
    return df_songs, df_song_extra_info, df_members, df_train, df_test
```

The initial data was imported from csv files into dataframes for analysis:

df_train		df_test		df_songs		df_song_extra_info		df_members	
Shape: (7377418, 6)		Shape: (2556790, 6)		Shape: (2296320, 7)		Shape: (2295971, 3)		Shape: (34403, 7)	
msno	obj	id	int	song_id	obj	song_id	obj	msno	obj
song_id	obj	msno	obj	song_length	int	name	obj	city	int
source_system_tab	obj	song_id	obj	genre_ids	obj	isrc	obj	bd	int
source_screen_name	obj	source_system_tab	obj	artist_name	obj			gender	obj
source_type	obj	source_screen_name	obj	composer	obj			registered_via	int
target	int	source_type	obj	lyricist	obj			registration_init_time	int
				language	float			expiration_date	int

The data is then cleaned:

- Remove all non-English entries (as indicated by the float 52 in df_songs
- Remove all non-Roman characters and punctuation from artist_name, composer and lyricist in df_songs
- Replace all spaces with "_"
- Create new feature for membership days (expiration_date - registration_init_time)
- Remove all columns with NA values contributing to more than 1% of total values
- Remove all ages younger than 15 years and older than 55

The df_train dataframe is then merged with features from df_songs, df_song_extra_info and df_members and the df_test dataframe is merged with features from df_songs, df_song_extra_info and df_members:

Dataframe	Description
df_train_clean	Shape: (336702, 14) • msno: unique user ID • song_id: unique song id • source_system_tab: music app tab where the event was triggered • source_type: first play source (album, playlist etc) • song_length: in ms • genre_ids: genre category. Some songs have multiple genres and they are separated by | • artist_name: name of person that performed the song • composer: name of person that wrote the music for the song • lyricist: name of person that wrote the lyrics for the song • city: city where user lives • age: age of the user • registered_via',

	• membership_days: number of days the listener has been a member of the service
	• count_song_played (new feature)
	• count_artist_played (new feature)
	• target: target variable for prediction. Values:
	  ○ 1 = member has listened to a song 2 or more times
	  ○ 0 = member has listened to a song < 2 times

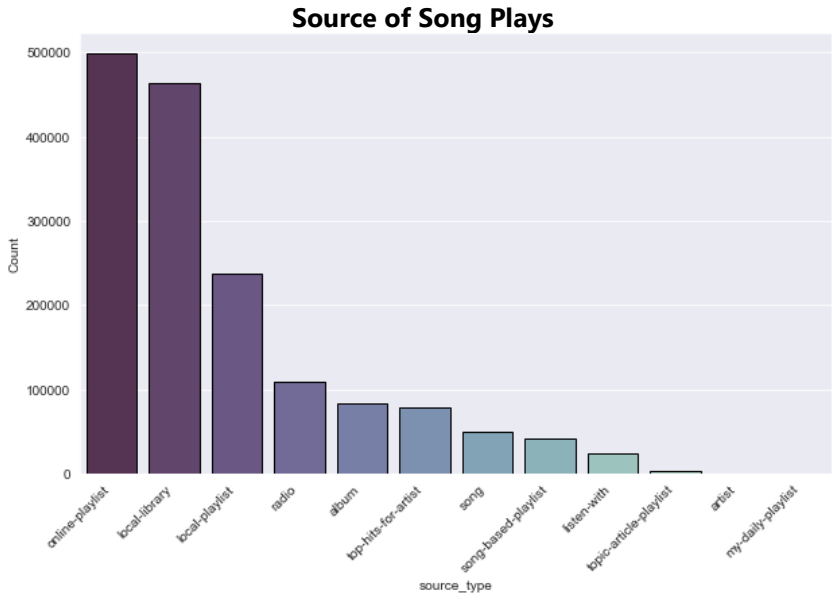
df_test_clean	Shape: (118280, 14) • id: row id • msno: unique user ID • song_id: unique song id • source_system_tab: music app tab where the event was triggered • source_type: first play source (album, playlist etc) • song_length: in ms • genre_ids: genre category. Some songs have multiple genres and they are separated by | • artist_name: name of person that performed the song • composer: name of person that wrote the music for the song • lyricist: name of person that wrote the lyrics for the song • city: city where user lives • age: age of the user • registered_via', • membership_days: number of days the listener has been a member of the service • count_song_played (new feature) • count_artist_played (new feature)

Additional resources used in this analysis include:
https://www.kaggle.com/myonin/music-recommendation-random-forest-xgboost
https://www.kaggle.com/rohandx1996/recommendation-system-with-83-accuracy-lgbm
https://www.kaggle.com/freshwater/basic-of-lgbm

Observations

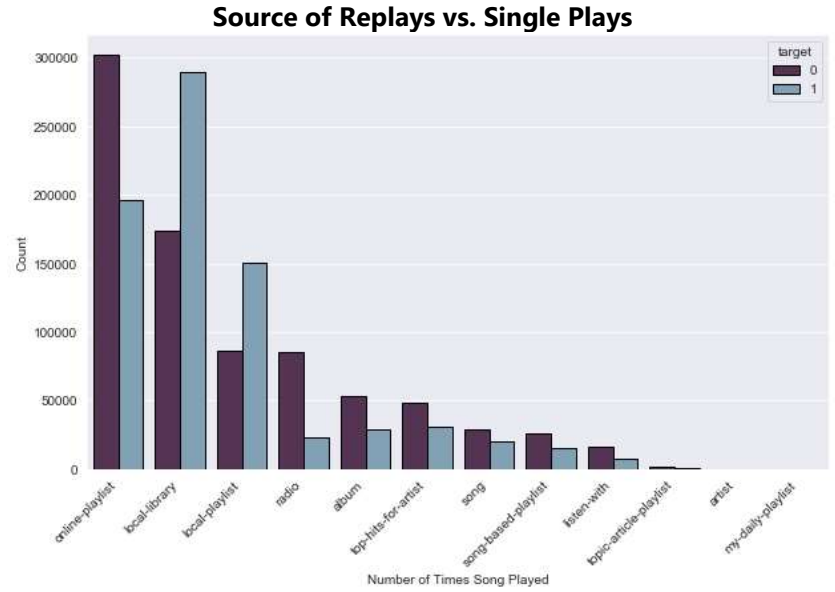
Observation (4 points)—What can you conclude from your observation and visualization of the data?

Analysis of the cleaned dataset was conducted across key attributes to identify patterns that may help inform the final models.



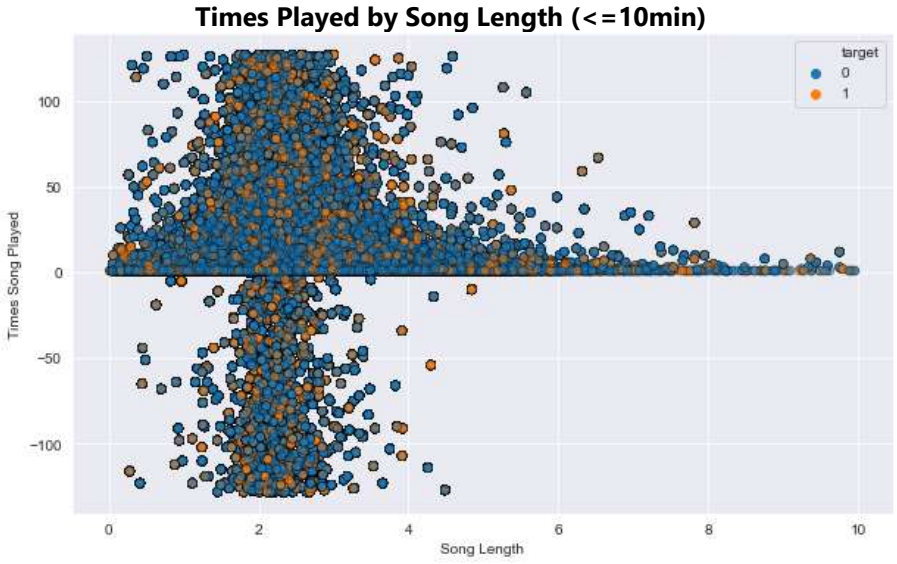
The primary sources for triggering a listening event (user listening to a song) are personalized sources such as playlists and local libraries.

This analysis is consistent with findings by MusicWatch that nine in ten U.S. streaming service users had created a playlist in the previous three months and almost half said they had listened to their favorite "way more than 10 times."[v]



Further analysis of the source of a triggered listening event shows that personal sources contribute to the most single events (listened to only once) and repeated listens but the local library and local playlist have a higher instance of repeated plays (target = 1) than single plays (target = 0).

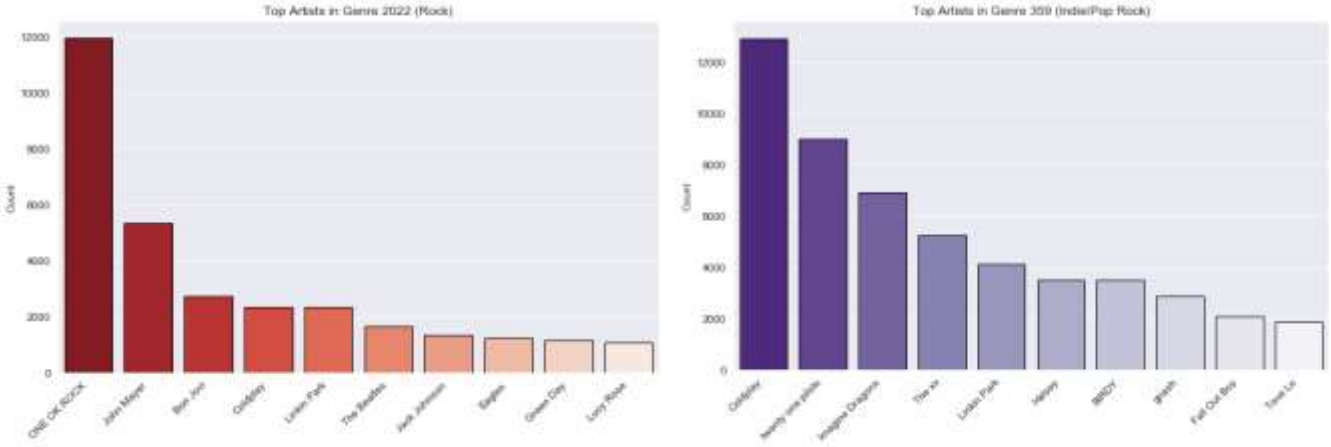
All other sources have a higher number of single listens than repeated listens.



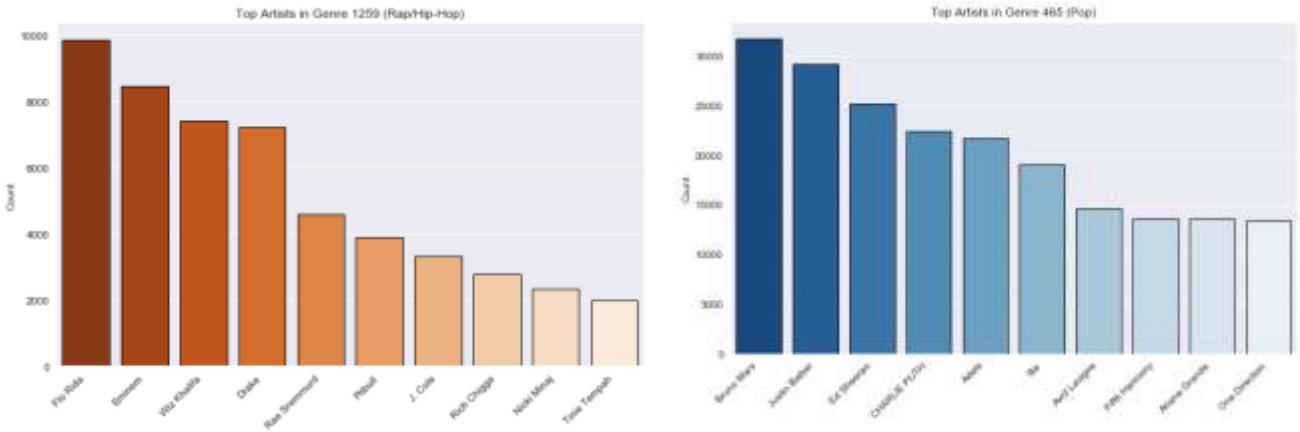
Analysis of relative number of times a song is played by song length shows for both single plays and repeated plays the highest number of chosen songs are clustered around 2 minutes.

Overall data shows that many more songs receive a single listen (Target = 0) but events resulting in single or repeated plays demonstrate a similar clustering.

Analysis of top bands by number of listens to their songs



Rock and Indie Pop/Rock genres show some overlap (e.g. Coldplay) which shows artists can be included in multiple genres. Rock genre shows a significanly higher song listen count for ONE OK ROCK (Japanese rock band, formed in Tokyo in 2005) which may indicate country level dynamics that impact popularity.



Hip-hop and Rap and pop genres show a linear distribution across the top bands by song listen count.

Top artists across selected genres

POP

Rap/Hip-Hop

Rock

Indie Rock/Pop

Analysis

Analysis (10 points)—Explain what types of analysis you used. Explain how you arrived at the model(s) you used for your recommendation.

The training data is split into a train and test set with a 70%, 30% split.

```
#####################################################
####       Create Training/Test Split            ####
#####################################################
target = train.pop('target')
train_data, test_data, train_labels, test_labels = train_test_split(train, target, test_size = 0.3)
```

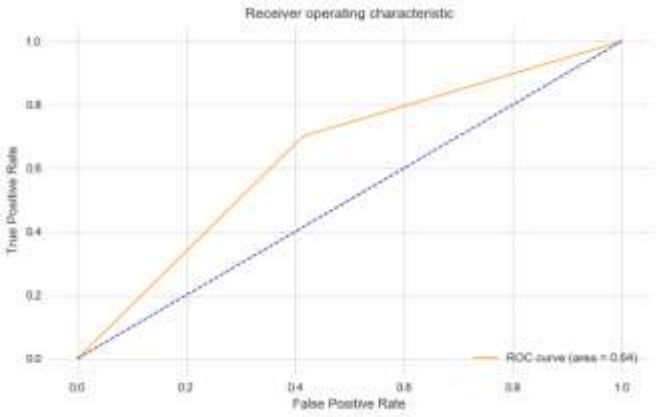
Random Forest Classifier

A Random Forest algorithm is chosen for its ability to handle large datasets with high dimensionality, the ability to manage missing values without significantly impacting the outcome and to minimize overfitting by averaging results.

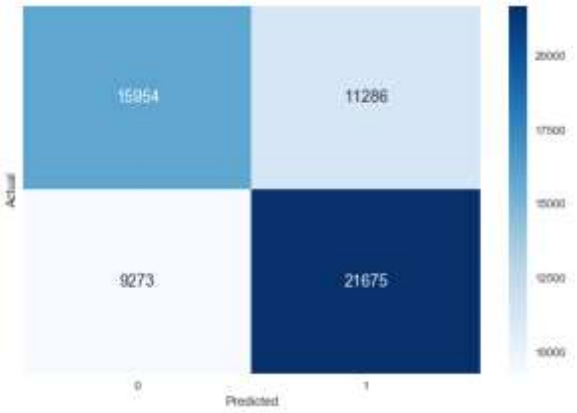
A random forest classifier is built:

	precision	recall	f1-score	support
0	0.62	0.59	0.60	27165
1	0.66	0.68	0.67	31023
accuracy			0.64	58188
macro avg	0.64	0.64	0.64	58188
weighted avg	0.64	0.64	0.64	58188

The Random Forest Classifier performed better on repeated listens (f1 = 67%) than single listens (f1 = 60%)

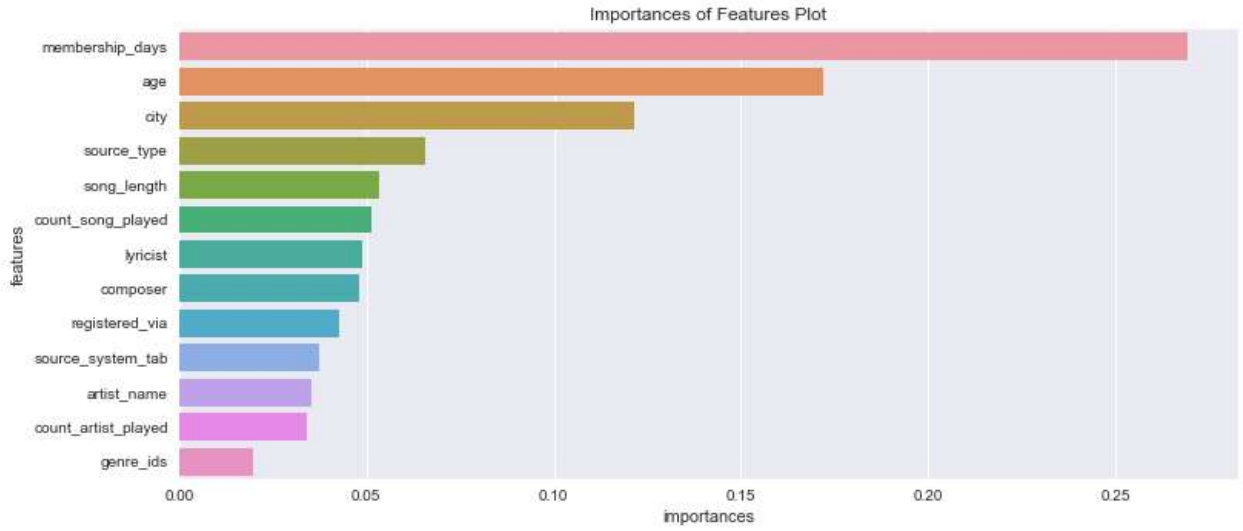


The Receiver operating characteristic plot (ROC) shows the ROC curve is close to the Area Under the Curve plot which indicates this is not a performant model.



Additional analysis of the confusion matrix shows there are a significant number of misidentified single listen where target = 0 (11,266).

Analysis of importance of features shows membership days (how long the member has had an account) is the most impactful attribute with age and city influencing predictions. Values with <0.05 include the artists name and the number of times an artist has been played overall. Additionally, genre does not have a significant impact on predictions.



User ID {mnso} and the song id (song_id) have been removed from the analysis as they will not scale analysis across new users or songs. Additionally, features with importance of < 0.05 are removed due to their limited impact on analysis.

XGBClassifier

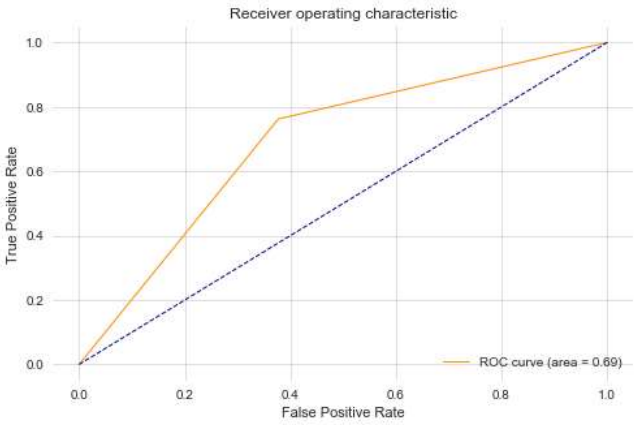
Boosting ensemble techniques weight votes with a collection of classifiers by averaging the prediction over a collection of classifiers and combining a set of heterogeneous classifiers.[vi]

The XGB Classifier was chosen due to its speed and focus on minimizing potential loss for all possible splits and creates a new branch.[vii]

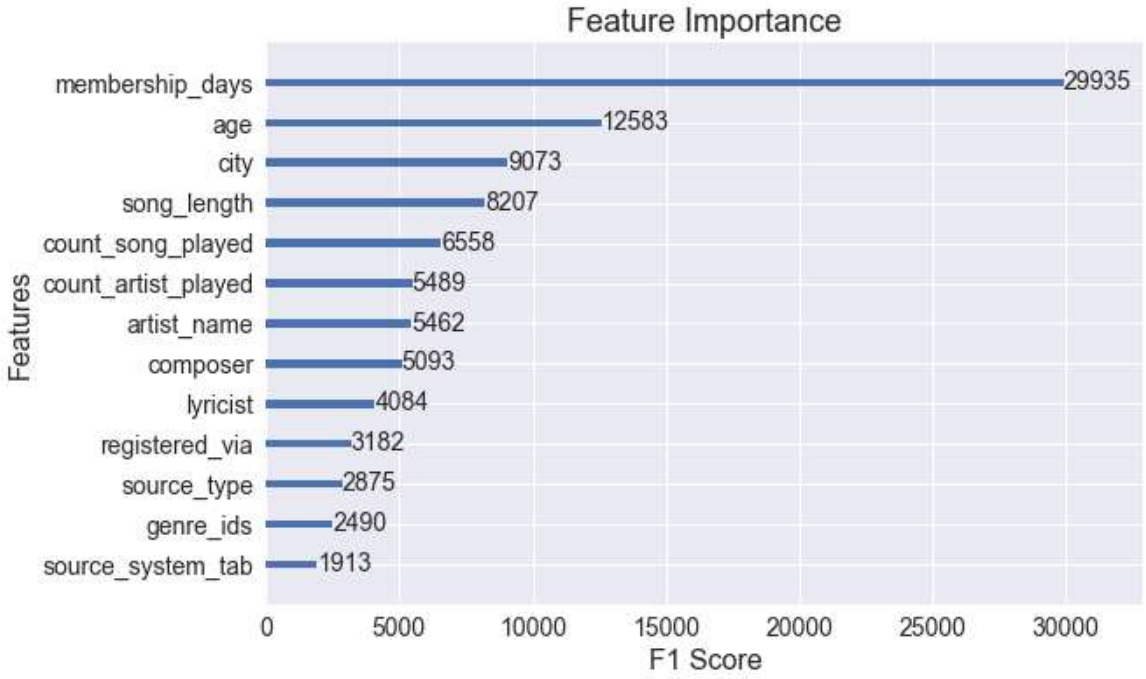
```
# Create model
print('Training Model')
model2 = xgb.XGBClassifier(learning_rate=0.1, max_depth=10, min_child_weight=10, n_estimators=250)
model2.fit(train_data, train_labels)
```

	precision	recall	f1-score	support
0	0.69	0.62	0.65	27165
1	0.69	0.75	0.72	31023
accuracy			0.69	58188
macro avg	0.69	0.68	0.68	58188
weighted avg	0.69	0.69	0.69	58188

n_estimators = dictates how many subtrees will be trained and helps to manage overfitting.
max_depth = maximum tree depth each individual tree h can grow to
learning_rate = manages through local minima



The XG Boost Classifier performed better on repeated listens (f1 = 72%) than single listens (f1 = 65%) with a weighted average of 69%. This level of accuracy provides nominal confidence in prediction. The Receiver operating characteristic plot (ROC) shows the ROC curve is close to the Area Under the Curve plot which indicates this is not a significantly performant model.



Feature importance for XGBoost classifier is similar to the Random Forest results. Number of membership days has the most significant impact on predictions and model performance.

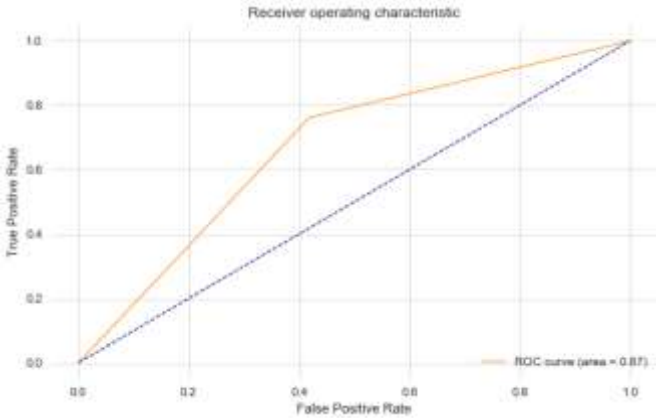
CatBoostClassifier

The CatBoost Classifier was chosen as it is a performant classifier that can easily handle categorical features.[viii]

```
# Create model
print('Training Model')
model3 = CatBoostClassifier(learning_rate=0.1, depth=10, iterations=300,)
model3.fit(train_data, train_labels)
```

	precision	recall	f1-score	support
0	0.68	0.58	0.63	27240
1	0.67	0.76	0.72	30948
accuracy			0.68	58188
macro avg	0.68	0.67	0.67	58188
weighted avg	0.68	0.68	0.67	58188

The Category Boost Classifier performed better on rep eated listens (f1 = 72%) than single listens (f1 = 63%) with a weighted average of 67%.



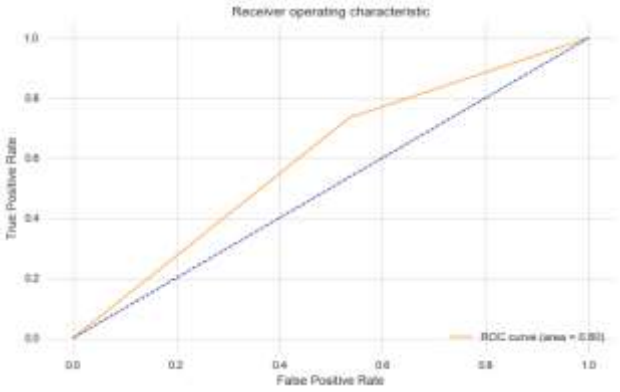
The Receiver operating characteristic plot (ROC) shows the ROC curve is close to the Area Under the Curve plot which indicates this is not a significantly performant model.

Naïve-Bayes

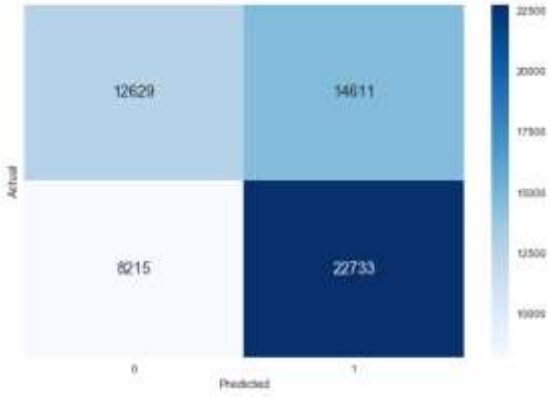
Naïve-Bayes was chosen due to the fast training and low resource usage.

	precision	recall	f1-score	support
0	0.61	0.46	0.53	27240
1	0.61	0.73	0.67	30948
accuracy			0.61	58188
macro avg	0.61	0.60	0.60	58188
weighted avg	0.61	0.61	0.60	58188

The Category Boost Classifier performed better on repeated listens (f1 = 67%) than single listens (f1 = 53 %) with a weighted average of 60%. These results show this is the least performant model.



The Receiver operating characteristic plot (ROC) shows the ROC curve is close to the Area Under the Curve plot which indicates this is not a significantly performant model.



Analysis of the confusion matrix shows similar results there are a significant number of misidentified single listen where target = 0 (11,266).

Comparison of the models shows no models exceed a weighted f1 value of 70%, with a lower performance on single listens due to false positives.

Random Forest	XGB Classifier	CatBoost Classifier	Naïve-Bayes
Accuracy (f1) Single Listen: 60% Repeated Listen: 67% Weighted: 64% **Features** membership_days age city	**Accuracy** (f1) Single Listen: 65% Repeated Listen: 72% Weighted: 69% **Features** membership_days age city	**Accuracy** (f1) Single Listen: 63% Repeated Listen: 72% Weighted: 67%	**Accuracy** (f1) Single Listen: 53% Repeated Listen: 67% Weighted: 60%

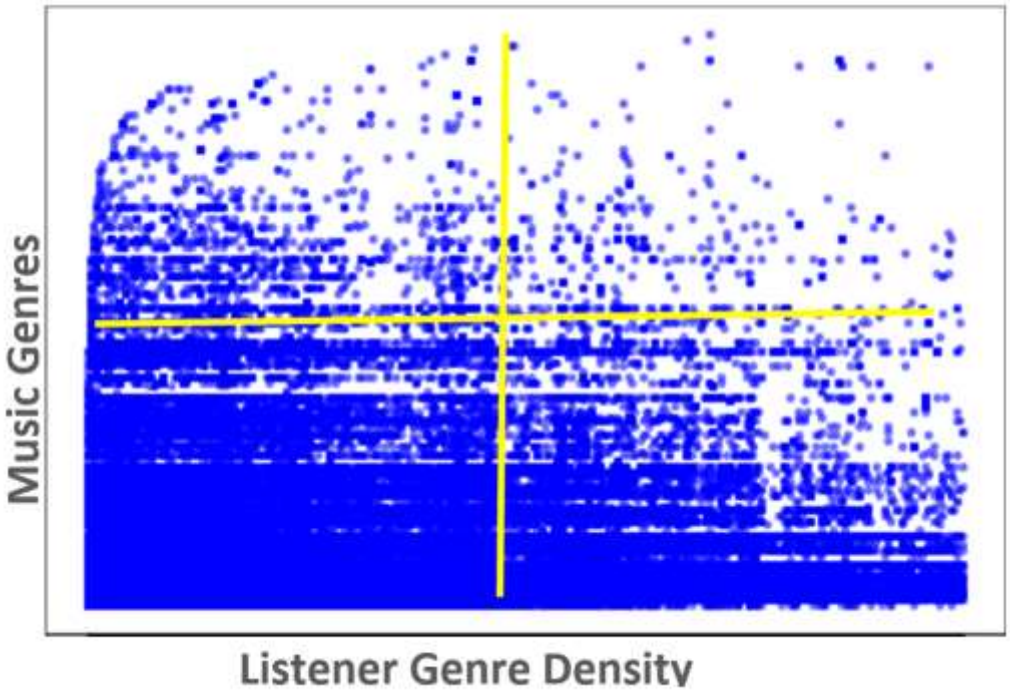
Association Rules

Where association rules are less useful in terms of predicting likelihood of listening to a follow-song; the market basket approach is helpful in terms of assessing relationships amongst music tastes in terms of antecedents and consequents. For the training set rules created by making baskets for each user based on total genre counts in their listening basket. The following illustrates an interesting finding of *country* music playing a key role in top association rules generated.

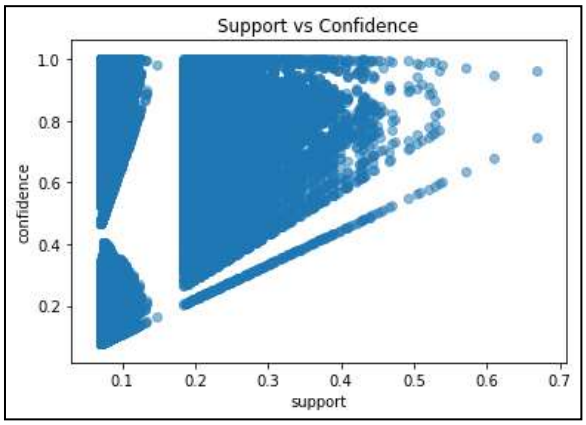
The following illustrates an interesting finding of "country" music playing a key role in top rules generated out of 300 possible genres.

Top Rules	antecedents	consequents	antecedent support	consequent support	support	confidence	lift
0	Country	HipHop	0.347539634	0.588840848	0.301461808	0.867417062	1.473092541
1	HipHop	Country	0.588840848	0.347539634	0.301461808	0.511958042	1.473092541
2	Country	Contemporary	0.347539634	0.518797612	0.281737698	0.810663507	1.562581417
3	Contemporary	Country	0.518797612	0.347539634	0.281737698	0.543058973	1.562581417
4	Country	Dupstep	0.347539634	0.696191064	0.318632901	0.916824645	1.316915271
5	Dupstep	Country	0.696191064	0.347539634	0.318632901	0.457680251	1.316915271
6	Country	Electronic	0.347539634	0.557092856	0.278402306	0.801066351	1.437940448
7	Electronic	Country	0.557092856	0.347539634	0.278402306	0.499741296	1.437940448
8	Country	Pop Rock	0.347539634	0.471607988	0.253325098	0.728909953	1.545584406
9	Pop Rock	Country	0.471607988	0.347539634	0.253325098	0.537151838	1.545584406

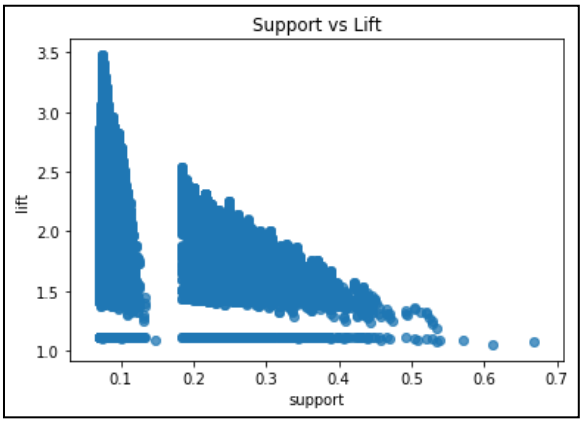
The following graphic is a scatterplot of total music genres (y-axis) in the KDSM kaggle music challenge, group 3 used in their project, with a 1.5M data points graphed on the x-axis. It provides a very dense view but does illustrate in the upper right quadrant the music categories "least likely" to be followed listening to a prior music category.



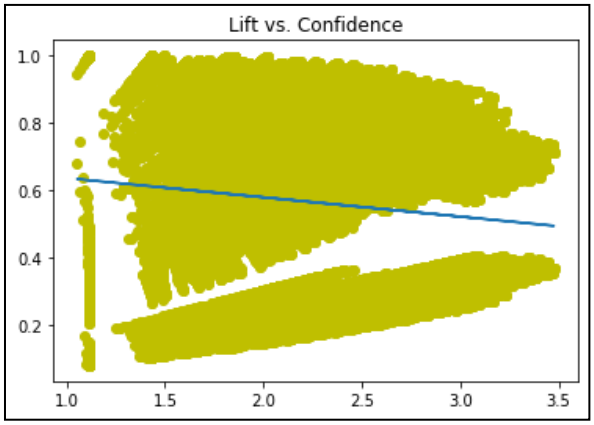
Assessing "Support vs. Confidence" resulted in a narrowing of rule support (red arrow) helping to illustrate while initially there is very high support for the rules generate the frequency of rules (x-axis) rapidly diminishes from 0 to 1 (y-axis) informing the rules are not very robust.



Assessing "Support vs. Lift" helps visualize relationship of rules that appear more than once, i.e. support values greater than 1, versus the frequency basket i.e. number of genre songs. The following illustrates a majority of rules with less than 25% support given a listener's portfolio genre. While there are a large number of rules greater than 1 there is unfortunately a low frequency of them suggesting prediction outcomes would not be robust.



The following compares "Lift vs. Confidence" with Confidence on the y-axis. Confidence explains how likely one item is listened to, i.e. purchased, when another song is listened. The result illustrates a trending towards lower confidence suggesting the presence of lower quality rules with reduced prediction power.



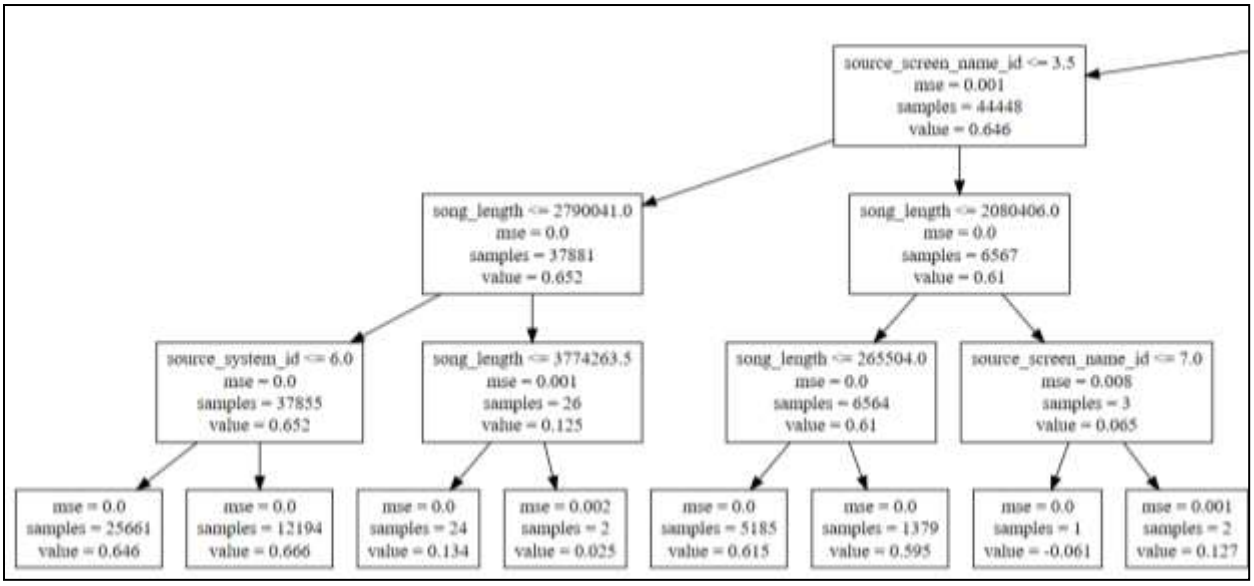
Decision Tree

A decision tree approach was tested to see if any insight could be found from this modeling technique. A rather wide tree resulted with low proportion scoring indicating the overall approach as unfavorable. Graphing was performed with the graphviz application using the web version at http://www.webgraphviz.com/. The following details the learnings.

Referring to the following is a tree constructed using "source type" as the main branching indicator. While some of the features resonate with prior model's more successful features this effort resulting tree branching is:

- Source Type (library, radio, etc)
  - Source screen name (discover genre, my library, etc)
    - Song length

Zoom in View:



Prediction outcomes were very poor so further work with the model was abandoned.

```
Full Tree Proportion of Training Set Variance Accounted for:  0.082

Full Tree Proportion of Test Set Variance Accounted for:  0.082

C:\Users\17574\Anaconda3\lib\site-packages\ipykernel_launcher.py:2:
```

Recommendations

Recommendation (8 points)—What should your customer do?

The models provide a better than 60% accuracy but result in a significant number of false positives that would result in recommending a song that the user does not like and only listens to once. While this may impact user confidence in recommendations, analysis shows performance will include as more user data is integrations. The following is recommended to manage these improvements:

1. Deploy the existing model and trigger retraining based on future data integration
2. Perform A/B testing on discrete and ensemble methods to compare performance
3. Determine additional information that can be integrated into the model (e.g. song rating, sentiment analysis in discussion forums)
4. Develop genre specific models to account for variations in user behaviors
5. Develop region specific models to account for variations in user behaviors

Additional work in the existing models should also continue through:

- Integration of additional user and song data (e.g. ratings and user sentiment)
- Explore ensemble models
- Analyze user age impact on prediction performance

Additional analysis should include exploration of ensemble approaches to integrate multiple models.

References

[i] Glenn Peoples, What Do Users Really Want In Their Premium Streaming Music Service?, HypeBot, Retrieved 3/23/2020 from https://www.hypebot.com/hypebot/2017/03/what-do-users-really-want-in-their-premium-streaming-music-service-glenn-peoples.html

[ii] Sarah Perez (2020), Streaming services accounted for nearly 80% of all music revenue in 2019, Tech Crunch https://techcrunch.com/2020/02/26/streaming-services-accounted-for-nearly-80-of-all-music-revenue-in-2019/

[iii] Amy Watson (2020), Most popular music streaming services in the United States in March 2018 and September 2019, by monthly users, Statista, https://www.statista.com/statistics/798125/most-popular-us-music-streaming-services-ranked-by-audience/

[iv] Glenn Peoples, What Do Users Really Want In Their Premium Streaming Music Service?, HypeBot, Retrieved 3/23/2020 from https://www.hypebot.com/hypebot/2017/03/what-do-users-really-want-in-their-premium-streaming-music-service-glenn-peoples.html

[v] Glenn Peoples, What Do Users Really Want In Their Premium Streaming Music Service?, HypeBot, Retrieved 3/23/2020 from https://www.hypebot.com/hypebot/2017/03/what-do-users-really-want-in-their-premium-streaming-music-service-glenn-peoples.html

[vi] Daniel Chepenko (2019), Introduction to gradient boosting on decision trees with Catboost, Towards Data Science, https://towardsdatascience.com/introduction-to-gradient-boosting-on-decision-trees-with-catboost-d511a9ccbd14

[vii] Gabriel Tseng (2018), Gradient Boosting and XGBoost, Medium, https://medium.com/@gabrieltseng/gradient-boosting-and-xgboost-c306c1bcfaf5

[viii] Sunil Ray (2017), CatBoost: A machine learning library to handle categorical (CAT) data automatically, Analytics Vidhya, https://www.analyticsvidhya.com/blog/2017/08/catboost-automated-categorical-data/