Music Recommendation System | **Capstone Project II: Milestone Report I**
Amanda Strack

Problem:

Recommendation systems are vitals features for companies such as Amazon, Netflix and Google. The underlying goal of the recommendation system is to personalize content and to present the most relevant items to the company's audience. This creates a better user experience while also driving incremental revenue.In this project I will build a recommendation system for users to discover new music.

For my recommendation system I will use a combination of collaborative filtering and classification.The approach will begin with matrix factorization of the user listen count data.Then the latent factors found will join the spotify audio data as additional features. Finally, classification will be performed to provide the final song recommendation.

Client:

My client is any music streaming company such as Spotify. The client will be able to use this data-driven product feature to enable users to more easily discover songs that they are likely to enjoy. The same approach could also work for other types of recommendation systems such as movies, as long as there is metadata such as the Spotify audio features. This feature will ultimately benefit the company because it will enhance user experience and keep users engaged.

Data:

The first part of my dataset comes from the Million Song Dataset which contains two files: triplet_file and metadata_file. The triplet_file of 2,000,000 rows contains user_id, song_id and listen count. The metadata_file of 1,000,000 rows of songs contains song_id, title, release_by (release date) and artist_name. I then merged the two datasets by song_id, bringing the count of unique songs from 1,000,000 to 10,000 songs.

The next part of the dataset was called from the Spotify Web API. To do this I used an open source tool (playlist-converter.net) which can convert a list of songs into a Spotify playlist. I exported a CSV of the 10,000 song names from our previous dataset and created Spotify playlists of around 7,000 of the same songs. The other 3,000 songs were not found in the Spotify search. Using the Spotify API, I extracted the artist name, song title, and spotify ID of the tracks of these playlists. I then merged this data set with our previous dataset by song title and artist.

The last part of the dataset are audio features also called from the Spotify API. The audio features were extracted using the unique spotify ID's of our dataset and then were added in as additional columns. Below is a description of these audio features.

KEY	VALUE DESCRIPTION
duration_ms	The duration of the track in milliseconds.
key	The estimated overall key of the track.
mode	Mode indicates the modality of a track. Major is represented by 1 and minor is 0.
time_signature	The time signature is a notational convention to specify how many beats are in each bar.
acousticness	A confidence measure from 0.0 to 1.0 of whether the track is acoustic.
danceability	Danceability describes how suitable a track is for dancing based on a combination of musical elements including tempo, rhythm stability, beat strength, and overall regularity.
energy	Energy is a measure from 0.0 to 1.0 and represents a perceptual measure of intensity and activity.
instrumentalness	Predicts whether a track contains no vocals.
liveness	Detects the presence of an audience in the recording.
loudness	The overall loudness of a track in decibels (dB).
speechiness	Speechiness detects the presence of spoken words in a track.
valence	A measure from 0.0 to 1.0 describing the musical positiveness conveyed by a track.
tempo	The overall estimated tempo of a track in beats per minute (BPM).

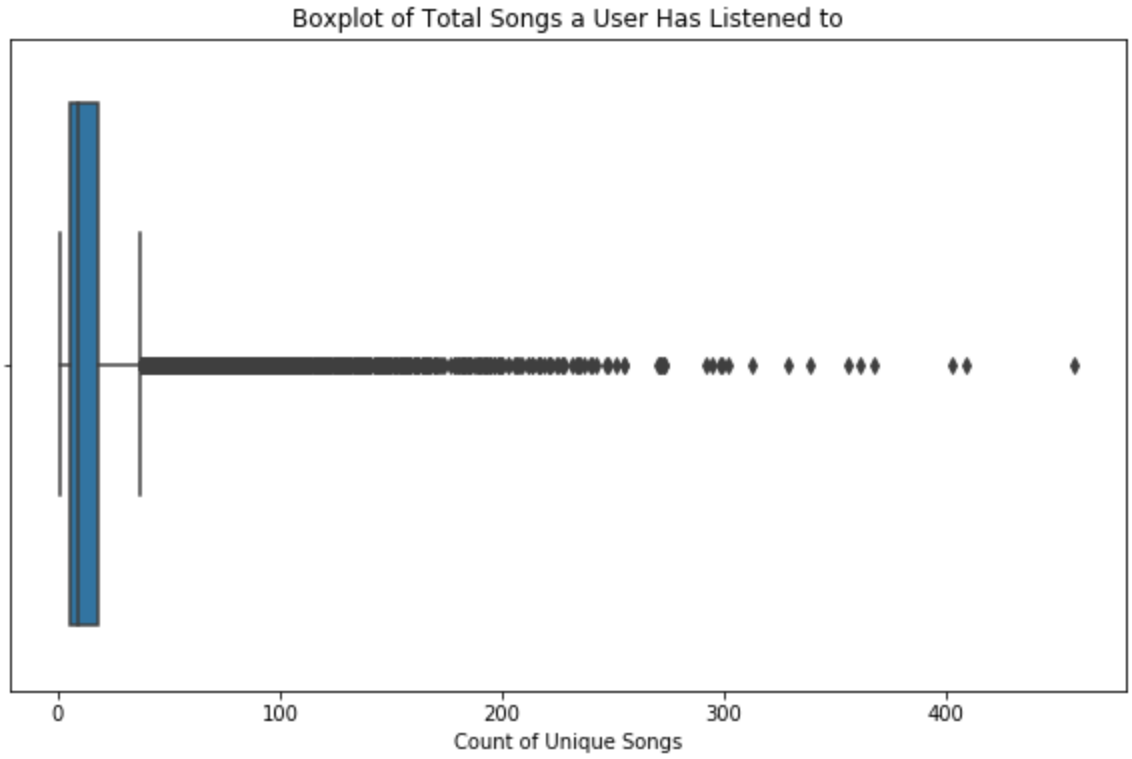
Exploratory Data Analysis:

We will explore the dataset consisting of user's listening activity. In this EDA we explore the dataset without the Spotify audio features. The purpose of this analysis is to understand the users listening behavior and see if there are any outliers that may skew our recommendations or make our dataset invalid. There are four particular areas we will explore:

- How many different songs does a user actually listen to?
- How many times is each song listened to?
- How many unique users listen to each song?

How many different songs does a user actually listen to?

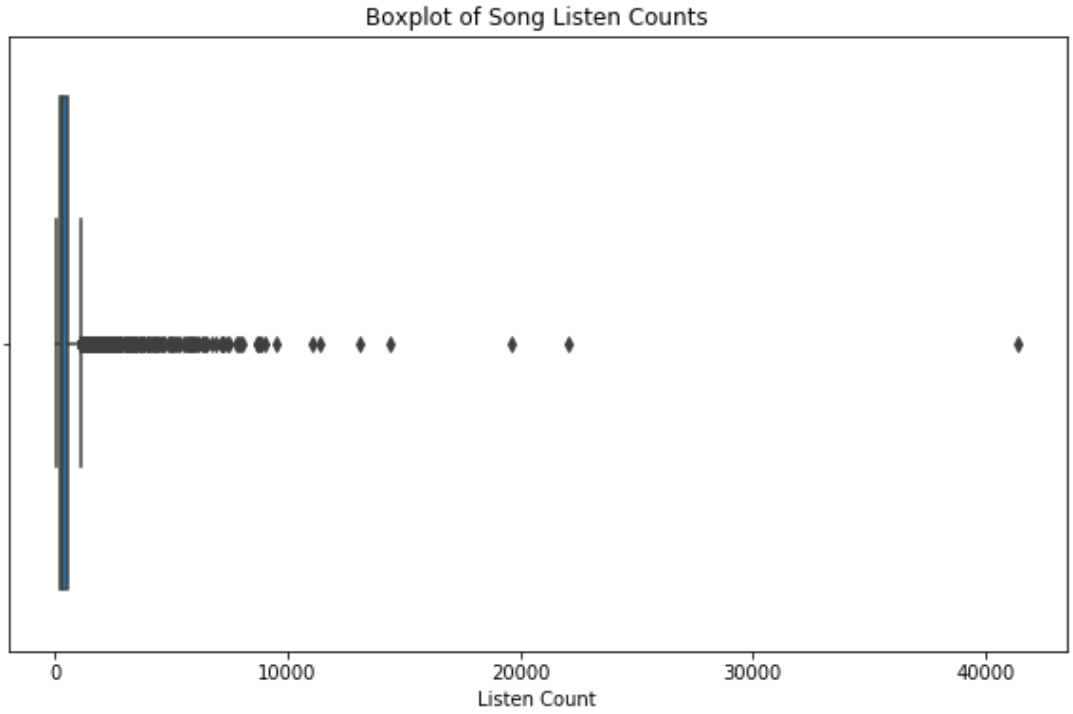
First we explore the distribution of how many different songs each user listens to. The purpose of this is to make sure that our data makes sense with real life user listening habits and to also make sure the data is valid enough to train our recommendation machine.



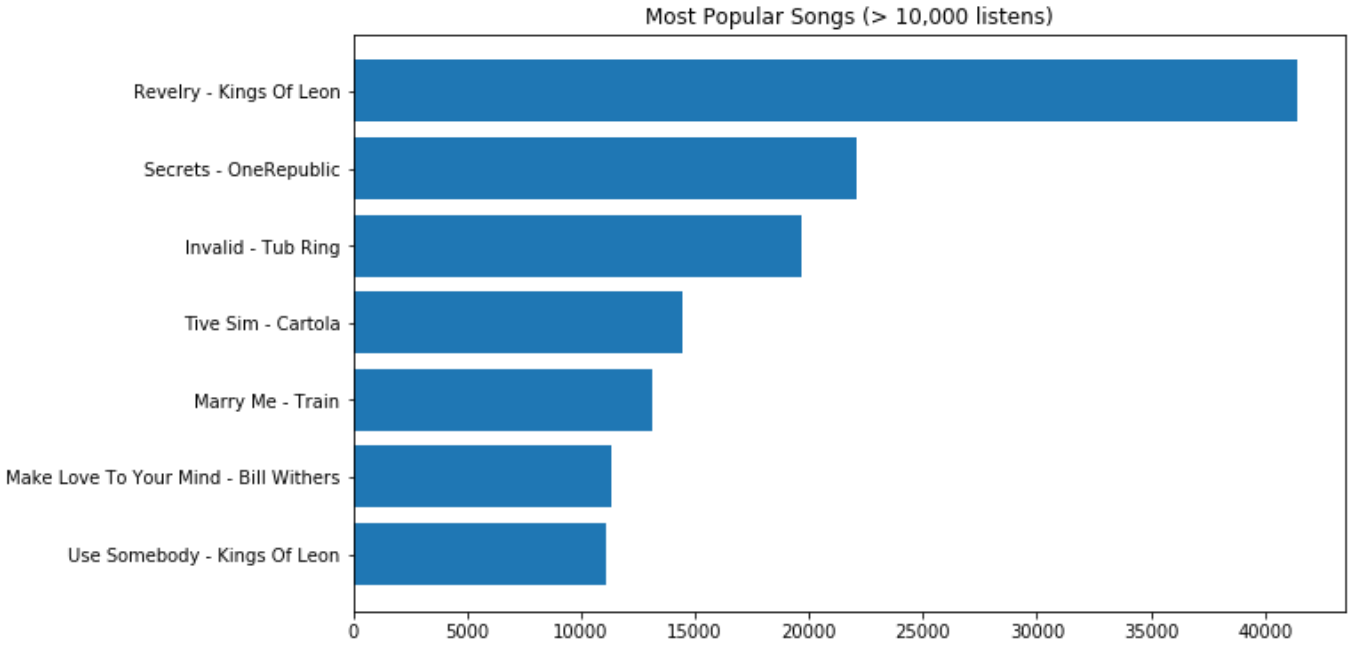
It does appear that there are quite a few outliers in regards to the amount of songs a user listens to. However, since we are building a recommendation system, outliers are treated differently as it is valid data of human activity. In this case, what we are concerned about is having a dataset that would not allow us to train a recommendation system. In our data, only 3357 out of 74,904 users only listened to one song (5%) so we can conclude that our data is valid for our model.

How many times is each song listened to?

Next we look at the number of times each song is listened to by users. We want to see if there are any outliers so that our analysis is not skewed when recommending songs. For example, we wouldn't want a song to be recommended to users only because it's listen count is considerably higher than all other songs.



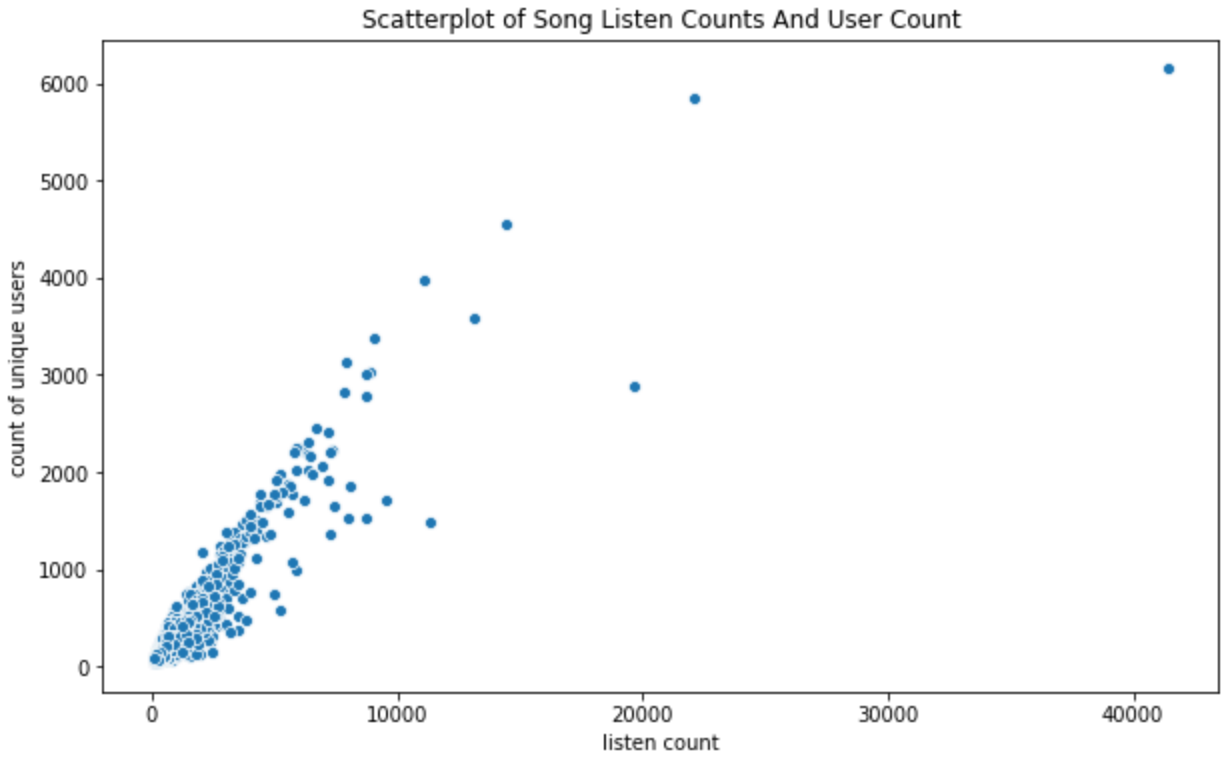
From the boxplot you can see that there are a few songs with very high listen counts. Below is a closer look at the songs with a listen count greater than 10,000.



These songs would be considered outliers, however we do not want to remove them from our data. But we do want to keep these songs in mind during our modeling stage so that they will not be recommended to users solely due to their popularity.

How many unique users listen to each song?

Next we looked at the distribution of the number of unique users that listen to each song. We explored this to see if the songs with high listen counts were caused by certain users listening to these songs an extreme amount of times as this could potentially skew our data.



The scatterplot shows that there is a relationship between number of listens and number of users. With the exception of a couple of outliers, the number of users that listen to a song increases as the number of listens increases. This tells us that the listen count of our popular songs is not skewed by a few users with extreme listen counts.

EDA Key Findings:

This analysis tells us that we have a valid dataset to begin training our recommendation engine. There are some songs that are 'outliers' in regards to listen count however we will not remove these outliers. We will just keep these songs in mind during the modeling stage and make sure they are not recommended to a lot of users solely due to their popularity.

Recommendation System:

Approach:

We will use a combination of matrix factorization and classification to produce song recommendations for a particular user. For our model we will randomly split the dataset into three sets. There will be two test data sets and one validation dataset. The first data set will be used to perform matrix factorization to extract user and item latent factors. The second dataset will be used to train our classification model. And lastly, our validation set will be used to evaluate our model.

Matrix Factorization:

Matrix factorization is to find out two matrices such that when you multiply them you will get back the original matrix. Matrix factorization can be used to discover latent features underlying the interactions between two different kinds of entities. The purpose of performing matrix factorization in our project is to extract latent factors of the users and the songs.

song	& Down - Boys Noize	' Cello Song - Nick Drake	'97 Bonnie & Clyde - Eminem	'Round Midnight - Amy Winehouse	'Round Midnight - Miles Davis	(Antichrist Television Blues) - Arcade Fire	(I Just) Died In Your Arms - Cutting Crew	(If You're Wondering If I Want You To) I Want You To - Weezer	(Nice Dream) - Radiohead	(The Symphony Of) Blase' - Anberlin	...
user_id											
00003a4459f33b92906be11abe0e93efc423c0ff	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...
00005c6177188f12fb5e2e82cdbd93e8a3f35e64	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...
00030033e3a2f904a48ec1dd53019c9969b6ef1f	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...
0007235c769e610e3d339a17818a5708e41008d9	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...
000a5c8b4d8b2c98f7a205219181d039edcd4506	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...

The intuition behind using matrix factorization is that there should be some latent features that determine how many times a user listens to a song. For example, two users would listen to the same song if they both like the genre of the song. Therefore if we can discover these latent features, we can add them as additional features to our dataset since the features associated with the user should match with the features associated with the song.

	user_id	u1	u2	u3	u4	u5	u6	u7
0	00003a4459f33b92906be11abe0e93efc423c0ff	2.292281e-08	8.675921e-06	0.000000	0.000055	0.000029	7.304217e-07	7.148265e-08
1	00005c6177188f12fb5e2e82cdbd93e8a3f35e64	0.000000e+00	0.000000e+00	0.000000	0.000196	0.000159	6.174749e-06	2.112216e-07
2	00030033e3a2f904a48ec1dd53019c9969b6ef1f	0.000000e+00	0.000000e+00	0.000000	0.000000	0.492211	0.000000e+00	0.000000e+00
3	0007235c769e610e3d339a17818a5708e41008d9	4.859820e-08	6.610512e-06	0.000002	0.000011	0.000012	2.942126e-05	1.186700e-05
4	000a5c8b4d8b2c98f7a205219181d039edcd4506	4.055804e-06	9.234415e-07	0.000052	0.000006	0.000018	3.937425e-07	3.131225e-05

	song	s1	s2	s3	s4	s5	s6	s7
0	Lights Of Ayodhya - Yulara	3.736008e-06	0.000069	0.000000e+00	0.000911	0.002214	0.036572	0.000100
1	Ironmasters - The Men They Couldn't Hang	6.799389e-07	0.000016	0.000000e+00	0.000091	0.000303	0.000071	0.000040
2	Chasing Cars - Snow Patrol	2.748346e-07	0.000018	5.600788e-07	0.000085	0.000606	0.000119	0.000012
3	Secrets - OneRepublic	2.552698e-07	0.000004	1.755655e-04	0.000471	0.001095	0.000043	0.000014
4	You'd Be So Nice To Come Home To - Julie London	9.558672e-07	0.000034	5.485172e-06	0.000166	0.000479	0.000121	0.000252

Classification:

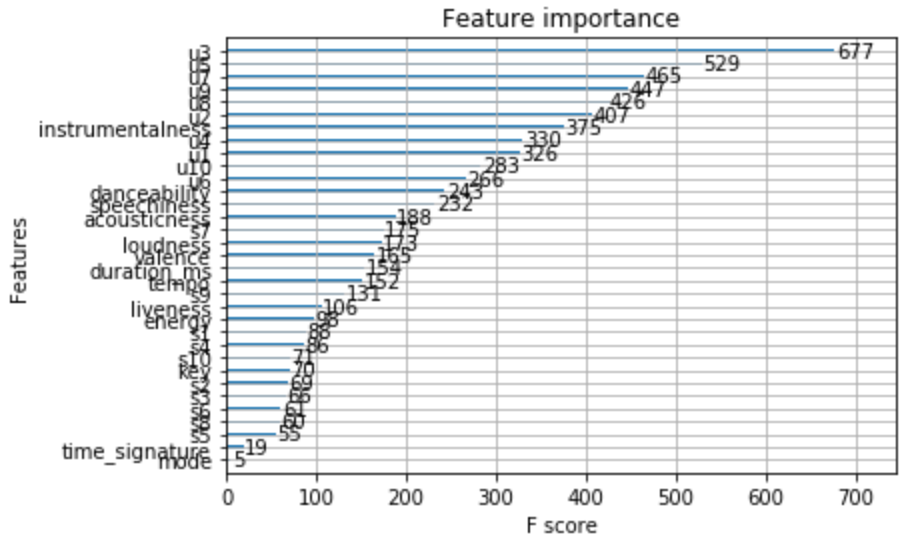
After adding on the latent factors found from matrix factorization, we then perform classification on the data set, ignoring the user_id and song_id columns. The column 'listen_count' is transformed into classes of 'one' and 'one_plus'. Since the listen counts are highly skewed we will only perform a binary classification.

XGBoost:

We use XGBoost as our classification technique. Without tuning the hyperparameters, the base model gives us an AUC score of 0.6026.

Feature Importance:

Now that we have extracted these latent factors from matrix factorization. Let's see how much of an effect these features have on our model. To do this, I used the feature_importances_ feature of our trained XGBoost model.



You can see that the user latent factors have very high 'importance' in comparison to our other features. However, our song or item latent factors have low 'importance' in compared to our other

features. We found that dropping the feature with the lowest F score ('mode') gives us the optimal AUC score.

Evaluation Metrics:

To evaluate our model we will use ROC AUC score because we have a binary classification with skewed classes. AUC - ROC curve is a performance measurement for classification problem at various thresholds settings. ROC is a probability curve and AUC represents measure of separability. It tells how much model is capable of distinguishing between classes.

Final Recommendation System:

At prediction time, if we want to know if a user will listen to a song we will join the user features and the song features of that song and predict. The function ‘get_top_songs’, takes in a user id as an argument and returns five recommended songs.

```
get_top_songs('f1ccb26d0d49490016747f6592e6f7b1e53a9e54')
```

	song_id	song	pred
3067	SOBJYFB12AB018372D	Also Frightened - Animal Collective	0.712179
924	SOCJWZY12A67021D18	Hallelujah - Rufus Wainwright	0.704709
1325	SOYFQVR12AB018ACA5	On Melancholy Hill - Gorillaz	0.696341
2837	SOAVQRP12A8C13120B	Kiwi - Maroon 5	0.695366
3117	SOWIGII12A58A7A939	Taxi Cab - Vampire Weekend	0.694435