Week 1 Writeup - Identify the Problem Statement and Dataset

Ryan O'Hara

Problem Statement/Project Value:

With the development of music streaming services over the past decade, music has never been more accessible than it is today. Listeners have a much wider variety of artists and genres today than they did when music was radio and tapes. According to variety.com and multiple other sources, "Stats now show that an average of more than 100,000 songs are being uploaded to digital service providers every day… if each of those 100,000 tunes lasted just three minutes, it would take over 30 years for one person to listen through all of the music released to and through DSPs in a day" (variety.com). The 100,000-song estimate came from the CEO of Universal Music and Warner Music.

With so much music available at the touch of a button is a major upgrade for most music listeners. One no longer has to pay for albums to get a hold of one song, create CDs for the car, pay $1 per song in iTUNES, etc. Listeners can now pay a monthly premium and have the freedom to listen as much as they want and to try as many songs/artists as they want. In addition, these services allow new artists to release music to the public without paying high fees or obtaining a record label. Artists are now paid by stream and not record sold and this can also benefit artists: popular artists can make money on their songs for a longer period of time and avoid illegally downloading (rampant before Spotify) and new artists can make a little bit of money right away without a record label.

Although music streaming is ultimately a positive development in the space for both artists and listeners, it has created a problem on the other end of the spectrum: there are too many songs to pick from. Many listeners revert back to artists they grew up with, songs that are on the radio, and mainstream music. The vast amount of new music being released every second makes it hard for both listeners to experience new music from small/new artists and it also makes it very hard for new artists to make a name for themselves without a record label. This phenomenon is called "choice overload" or "the paradox of choice" (thedecisionlab.com). The paradox of choice is described as, "how people get overwhelmed when they are presented with many options. While we tend to assume that more choice is a good thing, research has shown that, in many cases, we have a harder tome choosing from a larger array of options.

Having too many options can be incredibly overwhelming – and can actually make us feel less in control… [in times like these] we often settle for the default choice or end up making no decision at all" (thedecisionlab.com). This is prevalent on music streaming services such as Spotify and entertainment streaming services such as Netflix. As a result, Spotify recently launched its "DJ X" that is designed to provide a mix of songs that a listener is familiar with as well as mix in some new songs that the listener might like based on listening trends. Although a good idea in practice, DJ X seems to revert heavily to previous songs a listener liked at some point in the past as well as genre and tends to overplay these songs. Another problem with this approach is the listener almost gets punished for exploring new music they may not like if they choose to explore music outside of the DJ feature. Even a couple streams from a particular artist can cause DJ to play them on repeat for weeks, even if the listener is not too keen on them.

For this project we are aiming to take a creative approach on developing a song recommendation system that recommends songs based on metadata, lyrics, and genre tags of the input song rather than past listening trends of the user. A user will be able to manually enter a song they enjoy (most likely from a mainstream artist) and get one to five recommendations based on the one-song input. We hope that this version of a recommendation model will drop a lot the bias caused by recommending songs based on old listening trends. This model approach gives listeners more control and customization of recommendations. In addition, our model is designed to recommend both popular (mainstream) songs as well as underground artists due to the many parameters we would like to added into the model. We hope that this approach will benefit non-mainstream artists and aid them in being discovered and not drowned out by the top 1% of creators.

Calculation of Potential Economic Value:

Assumptions:

- Based on Spotify's Q1 report, we are assuming there are 317 million free users and 239 paid subscribers on the platform.
- $0.50 per month revenue from free users (ads)
- $11.99 per month subscription → assuming a $5 profit per subscriber
- Assuming $0.01/min for ad listening → Lifetime value of average user at $30

- Assuming our model can increase free users listening time by 2/min per day (less than 1 song) and improves retention by 0.5%
- Assuming this model can convert 0.1% of free users to paid subscriptions annually
- Costs $5 million annually to develop, maintain, and update at scale

Value Calculation:

Ad Revenue:

- (+ 2/min/day) X (30 days/month) X ($0.01/min) = $0.60 / user / month
    - To be conservative we will estimate the true value is 10% (adjusting for skips, loading, etc.)
        - True value = ($0.06 /user / month) X (12 months) X (317 million users) = $228.24 million

Converted Subscribers

- (0.1% conversion rate) X (317 million users) X ($5.00 / month) X (12 months) = $19.02 million

Retention Rate

- (0.5% less churn) X (556 million users) X ($30 / user) = $83.40 million

Overall

- $224.24 million + $19.02 million + $83.04 million - $5 million = +$325.3 million per year

(sourced from the group 1 writeup for consistency)

Project Plan:

Week 1 – Identify the Problem/Project Setup

- Identify a Problem Statement – Create a song recommendation system based on user input and not user past trends
- Value Statement – More streamlined recommendations, recommendations are not biased towards user listening trends, and helps smaller artists get noticed
- Find an initial dataset
- Create and set up GitHub repository

- Written report 1

Week 2 – Data Ingestion and Basic Data Exploration

- Loading initial dataset into Jupyter Notebook
- Verify Schema
- Inspect Data (types, completeness, integrity) + Basic cleaning (NaN, missing values, duplicates)
- Identify target/predictor variables
- Written report 2 + Notebook 2

Week 3 – Exploratory Data Analysis

- Visualize data
- Describe the data (count, mean, median, etc.)
- Correlation analysis across variables
- Data/variable definitions
- Week 3 report + notebook

Week 4 – Data Prep (Cleaning)

- Clean lyric data (remove stopwords)
- Encode genres (one-hot encoding)
    - Reduce genre dimensions
- Check missing values and NaN (completed in week 2)

Week 5 -Data Engineering and Making Data Model Ready

- Create features
    - Power = Energy X Loud
    - Sqrt_BPM
      Sqrt_Acoustic
    - Circular Camelot encoding
    - One-Hot Encode updated genre tags
- Normalize numeric variables
- Week 5 report + notebook

Week 6 – Baseline Model (simple)

- Implement a basic/simple approach to KNN (cosine distance)
    - Using only lyric embeddings
        - Keeps it simple and gives us an idea of the effect lyrics can have on recommendations
- Test the model by generating recommendations for 100 random songs and evaluating “relevant”/”good” – 1 or “irrelevant”/”bad” - 0 based on the input song.
- Week 6 report and notebook

Week 7 – More Complex Model

- Generate embeddings with Glove-Twitter API
    - Twitter API is believed to work better with lyrics due to slang and vulgar language commonly used in music
- Still use KNN and only lyrics
    - Help to determine which embedding API to use moving forward
- Testing the model on 100 random songs
- Week 7 report and notebook

Week 8 – More Complex Model

- Create a more advanced/data integrated KNN model
- Keeping Twitter embeddings
- Adding music features/metadata (Popularity, sqrt_BPM, sqrt_Acoustic, Camelot_sin, Camelot_cos, Power, etc.)
    - Weighing music features/metadata 50%
    - Weighing lyric embeddings 50%
    - Cosine distance
- Testing the model on 100 random songs
- Week 8 report + notebook

Week 9 – Most Complex Model, Model Comparison and Winning Model Selection

- Final approach – KNN (cosine distance)
    - Integrate genre tags as 10% distance
    - Music metadata – 45% distance
    - Lyric embeddings (Twitter) – 45% distance
    - Test on 100 random songs

- Compare results to the other models
    - Week 6 – not evaluated for accuracy (data changes were made)
        - Basic – not the winning model
    - Week 7 (lyrics only – Twitter) – 49 ± 5%
    - Week 8 (All features except genre) – 43 ± 5%
    - Week 8.2 – (All features except genre, lyrics weighed more) – 49 ± 5%
    - Week 9 (lyrics 45%, metadata 45%, genre 10%) – 60 ± 5%
- Winning model = Week 9 model – lyrics (45%), metadata (45%), genre tags (10%)
- Summarize Results
- Week 9 report + notebook

Week 10 – Data Centric AI

- Enhance model data
    - Removed underrepresented genres (integrated them into more popular tags)
    - Introduced “Artist” to the metadata (later taken out due to artist bias)
    - Addition of “subgenre” to the metadata
- With the adjustments the model lost about 2% accuracy. The tradeoffs were not enough to justify keeping these changes
- Week 10 report + notebook

Week 11 – Explain the model, analyze risk, bias, and ethics

- Explain the winning model
    - 45/45/10 weight split
- Protected Categories
    - No explicit protected categories in the model
    - Proxies – Artist, lyrics (slang – cultural ties)
- Bias
    - Vernacular, dialect, slang
    - Genres (all have different cultural and geographic roots)
    - Recognize that not all bias is harmful
- Risks
    - Fair representation of all artists
- Week 11 writeup + notebook

Week 12 – Save and Deploy the Model

- Pickle the model
- Environmental Dependencies
- Deployment: Spyder App
- Week 12 report + notebook

Week 13 – Put it All Together

- Constructing the final report
- Constructing the final coding script

Data Selection:

Our original raw dataset can be found under the “Playlist” folder. These csv files were gathered using a website (Tunebat.com) that takes Spotify playlist links and returns a file with all the songs information (Title, Artist, Genre) and metadata (BPM, Acoustic, Loud, Danceability, etc.). Data was pulled from 17 different highly rated playlists covering a wide variety of genres and time periods. Upon data collection, our initial dataset consists of about 9,000 songs.

Lyrics for the songs were gathered from genius.com using the genius API and were attached to the csv file rows.

Next steps will involve cleaning the data set. This will include deleting duplicate rows (songs), songs without lyric data, songs without metadata and remixes to start. The goal is to clean the dataset and get it down to a more manageable 3,000-4,000 songs.

Model Selection:

Our model is going to use the KNN algorithm framework, using cosine distance. The cosine distance metric “is used mainly to calculate similarity between two vectors. It is measured by the cosine of the angle between two vectors and determines whether two vectors are pointing in the same direction. It is often used to measure document similarity in text analysis” (kdnuggets.com). With our dataset, containing a complex vector for each song (including lyric data), we thought the cosine distance metric best suited our problem statement.

Our model will give the top 5 recommendations for a song given input. Our model will not consider past user listening trends and will be more focused on an input/output model rather

than attempting to cater the whole listening library to each individual user. This gives the user more freedom to explore genres that they may not have been exposed to in the past.

This model is an example of unsupervised learning due to the lack of labels in the data. Our model is not a traditional “right” or “wrong” model and introduces a high level of subjectivity. We will use KNN to plot all the songs in a high dimensional space and use cosine distance to find the closest songs to each other. Given this, we still need to measure accuracy, and we will do so by manually grading if a recommendation is relevant or not relevant for each random input (100 random songs per testing phase). This still is not completely supervised as the model will have no validation set and will not be able to adjust weights in real-time training.

Works Cited

“Choice Overload Bias.” The Decision Lab, thedecisionlab.com/biases/choice-overload-bias. Accessed 15 Aug. 2025.

“Most Popular Distance Metrics Used in KNN and When to Use Them.” KDnuggets, www.kdnuggets.com/2020/11/most-popular-distance-metrics-knn.html. Accessed 15 Aug. 2025.

Sharma, Amit. “How Much Does 1 Million Monthly Listeners on Spotify Pay?” Deliver My Tune, 30 Dec. 2024, blog.delivermytune.com/how-much-does-1-million-monthly-listeners-on-spotify-pay/.

Willman, Chris. “Music Streaming Hits Major Milestone as 100,000 Songs Are Uploaded Daily to Spotify and Other Dsps.” Variety, Variety, 7 Oct. 2022, variety.com/2022/music/news/new-songs-100000-being-released-every-day-dsps-1235395788/.