# Music Recommendation System (Capstone Project)

Kedar Kurpad, MIT-PE ADSP MAY24B

# Problem Definition

In today's fast-paced world, individuals often struggle to find time to engage with artistic and entertainment content despite technological advancements that simplify access to such content. For internet-based companies, maximizing user engagement is crucial, as revenue often correlates directly with the time users spend on their platforms.

To enhance user experience and drive engagement, these companies must accurately predict and recommend content that aligns with users' preferences. Spotify, a leading global audio content provider, faces the challenge of recommending relevant songs from an ever-expanding catalog. Given the vast amount of data on user preferences and song characteristics, identifying the most suitable recommendations has become a complex task.

The goal is to develop a recommendation system capable of suggesting the top 10 songs for a user based on their listening preferences. By leveraging extensive user and content data, this system will enhance user satisfaction and increase engagement on the platform.

**Key Questions**

1. What types of songs are users most likely to listen to? How can we predict the likelihood of a user enjoying a particular song?
2. What features of songs contribute to their popularity? Are there patterns in the types of songs that are frequently played or preferred?
3. How can we enhance our recommendation system to improve user engagement? What adjustments can be made to the recommendation algorithm to better meet user preferences?

**Possible Insights from Data Statistics**

1. Unique Users, Songs, and Artists: Total count of unique users, songs, and artists in the dataset to understand the breadth of the platform's content and user base.
2. Year-wise Songs Listened: Trends in song popularity and listening patterns over different years to identify shifts in user preferences and content consumption.
3. Most Played Song: Identifying the song with the highest play count to understand which songs are universally popular.
4. Maximum Number of Songs Played in a Year: Determining the highest number of songs a user has listened to in a single year to gauge the extent of user engagement.
5. Univariate and Bivariate Analysis:
    a. Univariate Analysis: Examining individual attributes like play counts, listening frequency, and song popularity.
    b. Bivariate Analysis: Exploring relationships between different variables, such as the correlation between song features and play counts, or between user demographics and song preferences.

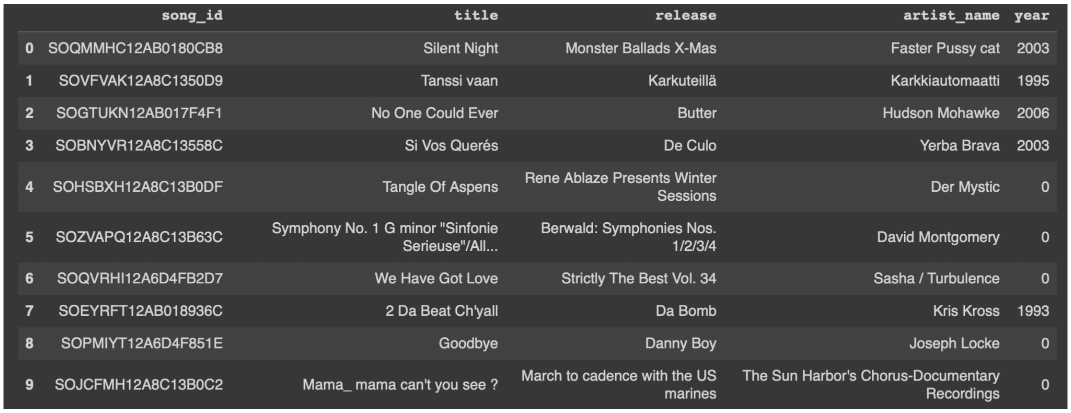
# Data Exploration

**Data Dictionary**

The primary dataset used here is the Taste Profile Subset from The Echo Nest, which is part of the Million Song Dataset. It includes two key files: one detailing song information such as song ID, title, release year, and artist name; and the other containing user data with user IDs, song IDs, and play counts.

**song_data**

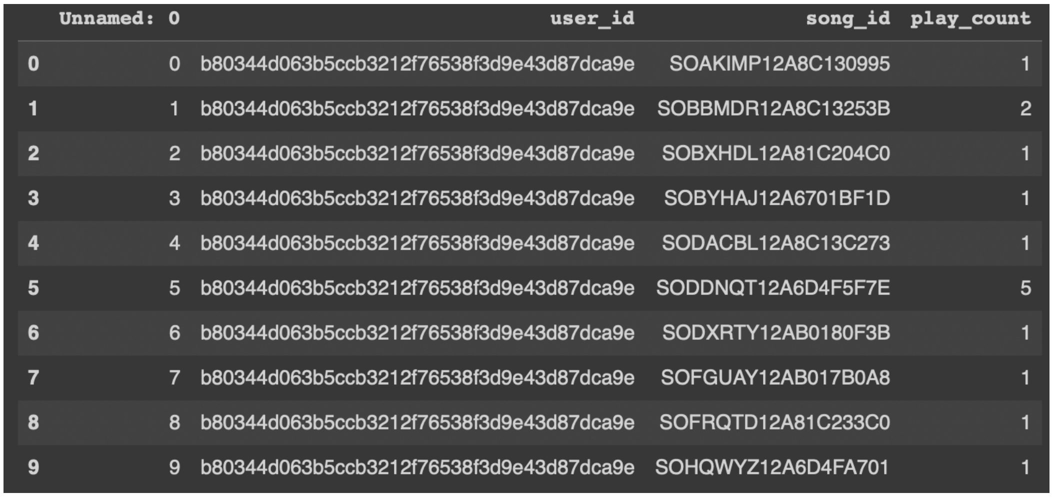
- song_id: A unique id given to every song
- title: Title of the song
- Release: Name of the released album
- Artist_name: Name of the artist
- year: Year of release



| | song_id | title | release | artist_name | year |
|---|---|---|---|---|---|
| 0 | SOQMMHC12AB0180CB8 | Silent Night | Monster Ballads X-Mas | Faster Pussy cat | 2003 |
| 1 | SOVFVAK12A8C1350D9 | Tanssi vaan | Karkuteillä | Karkkiautomaatti | 1995 |
| 2 | SOGTUKN12AB017F4F1 | No One Could Ever | Butter | Hudson Mohawke | 2006 |
| 3 | SOBNYVR12A8C13558C | Si Vos Querés | De Culo | Yerba Brava | 2003 |
| 4 | SOHSBXH12A8C13B0DF | Tangle Of Aspens | Rene Ablaze Presents Winter Sessions | Der Mystic | 0 |
| 5 | SOZVAPQ12A8C13B63C | Symphony No. 1 G minor "Sinfonie Serieuse"/All... | Berwald: Symphonies Nos. 1/2/3/4 | David Montgomery | 0 |
| 6 | SOQVRHI12A6D4FB2D7 | We Have Got Love | Strictly The Best Vol. 34 | Sasha / Turbulence | 0 |
| 7 | SOEYRFT12AB018936C | 2 Da Beat Ch'yall | Da Bomb | Kris Kross | 1993 |
| 8 | SOPMIYT12A6D4F851E | Goodbye | Danny Boy | Joseph Locke | 0 |
| 9 | SOJCFMH12A8C13B0C2 | Mama_ mama can't you see ? | March to cadence with the US marines | The Sun Harbor's Chorus-Documentary Recordings | 0 |

**count_data**

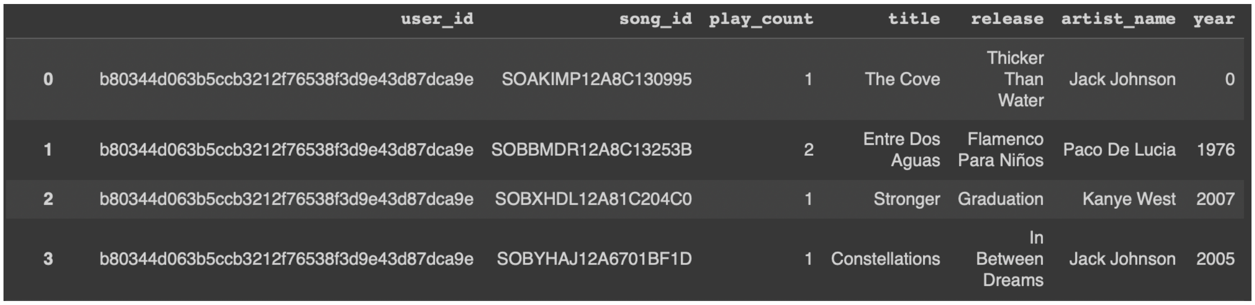
- user _id: A unique id given to the user
- song_id: A unique id given to the song
- play_count: Number of times the song was played



| | Unnamed: 0 | user_id | song_id | play_count |
|---|---|---|---|---|
| 0 | 0 | b80344d063b5ccb3212f76538f3d9e43d87dca9e | SOAKIMP12A8C130995 | 1 |
| 1 | 1 | b80344d063b5ccb3212f76538f3d9e43d87dca9e | SOBBMDR12A8C13253B | 2 |
| 2 | 2 | b80344d063b5ccb3212f76538f3d9e43d87dca9e | SOBXHDL12A81C204C0 | 1 |
| 3 | 3 | b80344d063b5ccb3212f76538f3d9e43d87dca9e | SOBYHAJ12A6701BF1D | 1 |
| 4 | 4 | b80344d063b5ccb3212f76538f3d9e43d87dca9e | SODACBL12A8C13C273 | 1 |
| 5 | 5 | b80344d063b5ccb3212f76538f3d9e43d87dca9e | SODDNQT12A6D4F5F7E | 5 |
| 6 | 6 | b80344d063b5ccb3212f76538f3d9e43d87dca9e | SODXRTY12AB0180F3B | 1 |
| 7 | 7 | b80344d063b5ccb3212f76538f3d9e43d87dca9e | SOFGUAY12AB017B0A8 | 1 |
| 8 | 8 | b80344d063b5ccb3212f76538f3d9e43d87dca9e | SOFRQTD12A81C233C0 | 1 |
| 9 | 9 | b80344d063b5ccb3212f76538f3d9e43d87dca9e | SOHQWYZ12A6D4FA701 | 1 |

**Observations & Insights**
After checking data types and missing values from each dataset, we can see our "count" dataset is complete with no missing values, and the data types consist of two integers and two objects. Our "song" dataset has several missing values - 17 missing titles and 7 missing releases, and the data types consist of four objects and one integer.

We drop unnecessary columns and merge the datasets to arrive at a combined data frame for our analysis:



| | user_id | song_id | play_count | title | release | artist_name | year |
|---|---|---|---|---|---|---|---|
| 0 | b80344d063b5ccb3212f76538f3d9e43d87dca9e | SOAKIMP12A8C130995 | 1 | The Cove | Thicker Than Water | Jack Johnson | 0 |
| 1 | b80344d063b5ccb3212f76538f3d9e43d87dca9e | SOBBMDR12A8C13253B | 2 | Entre Dos Aguas | Flamenco Para Niños | Paco De Lucia | 1976 |
| 2 | b80344d063b5ccb3212f76538f3d9e43d87dca9e | SOBXHDL12A81C204C0 | 1 | Stronger | Graduation | Kanye West | 2007 |
| 3 | b80344d063b5ccb3212f76538f3d9e43d87dca9e | SOBYHAJ12A6701BF1D | 1 | Constellations | In Between Dreams | Jack Johnson | 2005 |

We then perform the following cleaning / pre-processing steps:

- Get the column containing the users
- Create a dictionary that maps users(listeners) to the number of songs that they have listened to
- We want our users to have listened at least 90 songs
- Create a dictionary that maps songs to its number of users(listeners)
- We want our song to be listened by at least 120 users to be considered

This allows us to work with a much more manageable dataset, with the following metrics:

- 400730 rows
- 7 columns

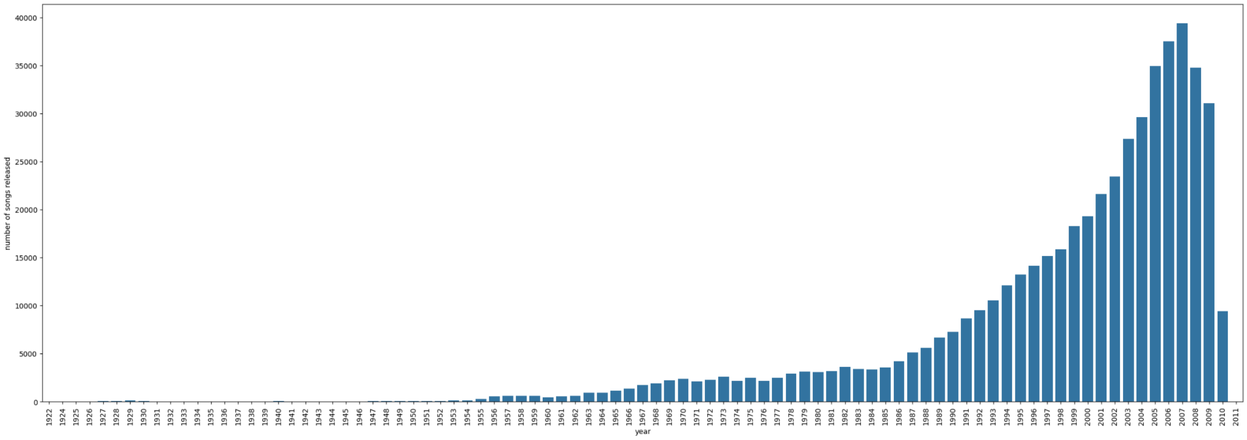
By examining single variables (univariate analysis) we can surface some preliminary insights about our dataset:

- 3156 unique user IDs
- 9998 unique songs
- 3374 unique artists

We can then see examine multiple variables at the same time to see how they might relate to each other (multivariate analysis):

- Top Songs
    - Dog Days Are Over (1634 plays)
    - Sehr kosmisch (1583)
    - Use Somebody (1463)
    - Secrets (1427)
    - Fireflies (1291)
- Most Active Users

    - 32542 (1157 plays)
    - 75144 (1032)
    - 31704 (981)
    - 10807 (903)
    - 7834 (896)
- Years With Most Songs in Dataset
    - 2007 (39414)
    - 2008 (34770)
    - 2009 (31051)
    - 2010 (9397)
    - 2011 (1)



We can see that there is significant recency bias in this dataset, which displays a left-skewed distribution. This may also indicate that more content in general was being produced in later years.

# Building Various Models

1. **Popularity Based**

   To build a recommendation system based on song popularity, we'll focus on calculating two key metrics: the average number of times each song is played and the total number of plays for each song.

   We'll then combine these metrics into a single dataset, which helps identify which songs are most popular.

   Next, we'll create a function to recommend the top songs based on their average play counts. This function also allows us to set a minimum play count threshold to ensure we recommend only those songs that have been played enough to be considered popular.

   Here's how it works:

   i. Average and Total Plays Calculation: Compute the average and total plays for each song.
   ii. Filter by Minimum Plays: Apply a filter to include only songs with a minimum number of plays.
   iii. Sort and Recommend: Sort the songs by their average play count and select the top ones.
   iv. In practice, we'll use this function to recommend the top 10 songs that meet a minimum play count threshold of 5.
   v. This is a simple and highly interpretable system, however we can do significantly better by applying more advanced technologies and algorithms to surface better tailored recommendations to specific users

2. **Collaborative Based (User-User, Item-Item)**

   a. In our user-user similarity-based collaborative filtering approach, we focus on recommending songs by leveraging the similarity between users. The process involves the following steps:

      i. Data Preparation: We use a dataset that includes user IDs, song IDs, and play counts to build a model.
      ii. Similarity Calculation: Compute the similarity between users based on their listening history. This helps identify users with similar tastes.
      iii. Prediction Generation: Use the similarities to predict which songs a user might like based on the preferences of similar users.
      iv. Evaluation: We measure the effectiveness of the recommendations using metrics like RMSE (Root Mean Squared Error). This helps in assessing the accuracy of our recommendations.
      v. Hyperparameter Tuning: Adjust parameters to optimize model performance, including factors such as the number of clusters in the co-clustering algorithm and the similarity measures used.

         1. The RMSE result suggests that on average, predictions deviate roughly 1.1 units from the actual values
         2. The precision result suggests that roughly 39% of recommended songs are relevant to the user

3. The recall result suggests that the model successfully captures about 60% of all relevant items in the test set. (better than precision, but still room for improvement)
4. The F1 result combines precision and recall to provide a general performance score of 0.47, again indicating significant room to grow
5. After fine-tuning our model using GridSearch CV, a technique for systematically searching for the best settings for our model, we achieved a significant boost in performance. The best RMSE score obtained was 1.0529, reflecting improved accuracy in our recommendations. The optimal parameters identified for the model were: {'k': 30, 'min_k': 9, 'sim_options': {'name': 'pearson_baseline', 'user_based': True, 'min_support': 2}}. These adjustments enhanced the model's ability to predict user preferences more accurately.

b. In our item-item similarity-based collaborative filtering approach, we recommend songs by examining similarities between items rather than users. The process involves:
   i. Data Preparation: We utilize a dataset containing song IDs, user IDs, and play counts to create the recommendation model.
   ii. Similarity Calculation: We compute the similarity between songs based on their play history to identify items with similar listening patterns.
   iii. Prediction Generation: Recommendations are generated based on the similarity between items, suggesting songs that are similar to those a user already enjoys.
   iv. Evaluation: We assess the quality of our recommendations using metrics like RMSE (Root Mean Squared Error) to gauge prediction accuracy.
   v. Hyperparameter Tuning: We fine-tune the model parameters to enhance performance, including aspects such as the number of factors and similarity measures used.
      1. After employing GridSearchCV to find the optimal model settings, we noticed a significant depreciation in recommendation accuracy (RMSE score: 5.150919469234268). User-User remains the better candidate at this time.

   *vi. Disclaimer: During my analysis, my item-item collaborative filtering model produced a suspiciously high RMSE score (in excess of 5), potentially indicating a coding error in my notebook. Further investigation is required on my end to fully understand this error.*

**3. Matrix Factorization**

a. Model-based Collaborative Filtering, often referred to as matrix factorization, creates personalized recommendations by analyzing a user's past behavior without relying on additional information. The term "matrix factorization" comes from the technique's ability to decompose a large matrix of user-item interactions into smaller, latent feature matrices. These latent features capture hidden patterns within the data, allowing the system to predict and suggest content that aligns with each user's unique preferences.

   i. Without optimization, we yield an RMSE of 1.0328, our best result so far.
   ii. After hyperparameter optimization, we arrive at an even better RMSE of 1.019.

4. **Content Based**
   a. In this content-based filtering approach, we attempt to build a recommendation system by incorporating additional features of the songs, such as the title, release, and artist name, rather than relying solely on play counts. The process involves the following steps:
      i. Data Preparation: We concatenate the title, release, and artist name into a single text column to represent the song's features. This allows us to capture more context about each song.
      ii. Text Preprocessing: We tokenize the text data, removing any non-alphabetic characters and stopwords, and apply lemmatization to normalize the words. This process ensures that the content is clean and ready for further analysis.
      iii. Feature Representation: We use TF-IDF (Term Frequency-Inverse Document Frequency) Vectorization to convert the text data into a numerical matrix, where each song is represented by its unique features. This matrix helps in quantifying the importance of different words in the context of the entire dataset.
      iv. Similarity Calculation: We compute the cosine similarity between the TF-IDF vectors of the songs. This measure helps us identify how similar different songs are based on their content features.
      v. Recommendation Generation: For a given song, we find the top 10 most similar songs by comparing their cosine similarity scores. These recommendations are based purely on the content of the songs, providing personalized suggestions that align with the user's existing preferences.
         1. This method could be useful for introducing users to new songs that share common characteristics with what they already enjoy, but it has shortcomings in terms of interpretability and evaluation.
         2. Content-based filtering can sometimes lack diversity in recommendations and may not account for the broader context of user preferences, as it relies heavily on item features without considering user interactions or preferences from other users

# Comparison of Various Techniques

To evaluate the effectiveness of the different recommendation techniques, we consider key metrics such as RMSE (Root Mean Squared Error), precision, recall, and the F1 score. These metrics provide insights into the accuracy, relevance, and overall performance of the recommendations.

1. RMSE (Root Mean Squared Error): RMSE measures the average deviation between predicted and actual ratings. A lower RMSE indicates higher accuracy in predictions.
2. Precision: Precision measures the proportion of recommended items that are relevant to the user. A higher precision value suggests that the system makes more relevant recommendations.
3. Recall: Recall measures the proportion of relevant items that were successfully recommended. A higher recall indicates that the system captures a larger portion of the user's preferences.
4. F1 Score: The F1 score is the harmonic mean of precision and recall, providing a balanced evaluation of the system's performance.

**Performance of Different Techniques:**

1. Popularity-Based Approach
    a. RMSE: N/A (Not applicable as this method does not predict ratings)
    b. Precision & Recall: N/A
    c. Strengths: This approach is simple and highly interpretable, providing recommendations based on what is popular across the user base. It is easy to implement and understand.
    d. Limitations: It lacks personalization and may not align with individual user preferences, as it does not consider user-specific behavior.
2. Collaborative Filtering (User-User)
    a. RMSE: 1.0529 (after optimization)
    b. Precision: ~39%
    c. Recall: ~60%
    d. F1 Score: 0.47
    e. Strengths: It captures the preferences of users with similar tastes, leading to personalized recommendations.
    f. Limitations: Precision is relatively low, and the model may struggle with sparse data. While recall is better, there's still significant room for improvement.
3. Collaborative Filtering (Item-Item):
    a. RMSE: 5.1509 (after optimization)
    b. Precision & Recall: Not specified, but lower performance is implied.
    c. Strengths: This method is effective when the user-item interactions are dense, providing recommendations based on item similarity.
    d. Limitations: The high RMSE indicates poor prediction accuracy, or potentially a coding error on my part. Further investigation is needed.

4. Matrix Factorization:
    a. RMSE: 1.019 (after optimization)
    b. Precision & Recall: Not specified, but performance is implied to be superior.
    c. Strengths: This approach captures latent factors in the data, enabling it to discover hidden patterns and make highly personalized recommendations.
    d. Limitations: Requires significant computational resources for optimization and may not be as interpretable as simpler methods.
5. Content-Based Filtering:
    a. RMSE: N/A (Not applicable as this method focuses on item features rather than predicting ratings)
    b. Precision & Recall: Not specified
    c. Strengths: It introduces users to new songs with similar characteristics to what they already enjoy, focusing on content rather than user behavior.
    d. Limitations: The method may lack diversity and not fully account for broader user preferences. It may also struggle with interpretability and evaluating recommendations.

**Best Performing Technique:**

- Matrix Factorization emerges as the best-performing technique in terms of RMSE, achieving the lowest error score of 1.019 after hyperparameter optimization. This suggests that it is the most accurate in predicting user preferences.

**Scope for Improvement:**

1. Collaborative Filtering: Further optimization of parameters or combining user-user and item-item methods could enhance performance. Addressing sparsity through techniques like dimensionality reduction or matrix completion might also improve precision and recall.
2. Content-Based Filtering: Enhancing diversity in recommendations by incorporating collaborative signals or user interaction data could address some of its limitations.
3. Matrix Factorization: Although it performs well, combining it with other methods like content-based filtering (a hybrid approach) could further refine the recommendations and improve interpretability.

In conclusion, while matrix factorization currently outperforms other methods in terms of RMSE, each technique has its strengths and weaknesses. There is potential to further improve performance by combining methods, fine-tuning parameters, and addressing specific limitations inherent in each approach.

# Proposal For Business Solution

I recommend implementing an optimized matrix factorization model as the core of our recommendation system. This model has demonstrated superior performance, achieving the lowest RMSE of 1.019 after hyperparameter tuning.

The matrix factorization model excels at capturing latent patterns within user-item interactions, making it highly effective for generating personalized recommendations. Its relatively low RMSE signifies accurate predictions, establishing it as a reliable choice for aligning with user preferences. Additionally, this method offers greater interpretability compared to more complex techniques, facilitating a clearer understanding of how recommendations are generated.

**How This Solves the Problem:**
The primary challenge is to deliver accurate, personalized recommendations that resonate with user preferences. By leveraging matrix factorization, the system can uncover hidden relationships in the data, enabling it to recommend songs that are more likely to appeal to individual users. This tailored user experience is essential for enhancing engagement and satisfaction.

The optimized matrix factorization model has proven effective, with performance metrics indicating its readiness for deployment in a production environment. Its interpretability, combined with robust predictive accuracy, makes it a viable solution for real-world applications. We can deploy this model confidently, knowing it will enhance user experience through precise and personalized recommendations.

**Potential Expansions/Enhancements:**
While matrix factorization serves as an excellent starting point, future iterations could explore hybrid models that combine matrix factorization with content-based filtering or collaborative methods. This approach could further enhance recommendation accuracy and address any remaining gaps in user satisfaction.

Thank you for reviewing my analysis of this rich dataset. Please direct any questions, follow-ups, and feedback to kedarkurpad@icloud.com.