Spotify Recommendation Algorithm

Our Agenda For Today

- How We Came Up With Our Idea and Found Our Dataset
- Different Approaches to Recommendation Algorithms
- Our Spotify Recommendation Algorithm: What It Does and How It Works
- How Does Our Algorithm Perform?
- Additional Topics for Further Discussion and Improvement

How We Came Up With Our Idea and Found Our Dataset

What Brings Us Together?

Potential Datasets

- Art and Artists
- Netflix Shows
- Open Data on AWS
- Phishing
- Spotify Hit Predictor

Our Dataset – 169,909 Songs from 1921-2020

The datasets features are:

Acousticness Artists Danceability Duration

Energy Explicit ID

Instrumentalness Key Liveness Loudness

Mode Name Popularity

Release date Speechiness Tempo Valence

Year

Different Approaches to Recommendation Algorithms

We Decided Against Our Initial Idea of Collaborative Filtering Due to a Lack of User Data

- Collaborative Filtering through Matrix Factorization is a common aspect of services like Netflix or Spotify.
  - It relies on similarities in taste and between songs to determine possible likes/matches.

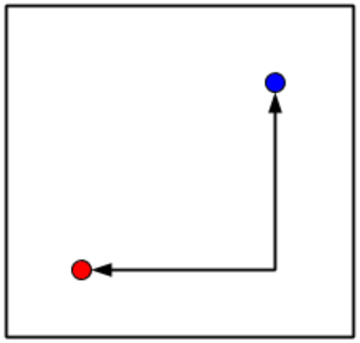
- We didn't use this, since this requires user data on song preference, which was not something we could get.

Preference	Song A	Song B	Song C	Song D
User A	1	2	1	1
User B		1	1	3
User C	1		3	2

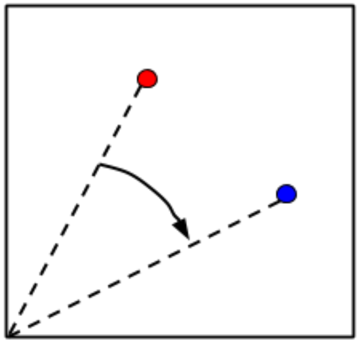
We Chose to Use Similarity Metrics for Our Recommendations

- To find song similarity, we needed to use metrics which calculated the distance between the numerical features of our data.
- Manhattan distance was our first choice, since it works better in higher dimensions, and is less sensitive to extreme values
- Cosine similarity, while not equivalent to Euclidean distance, gives us something similar.
  - More on that later…

Manhattan Distance

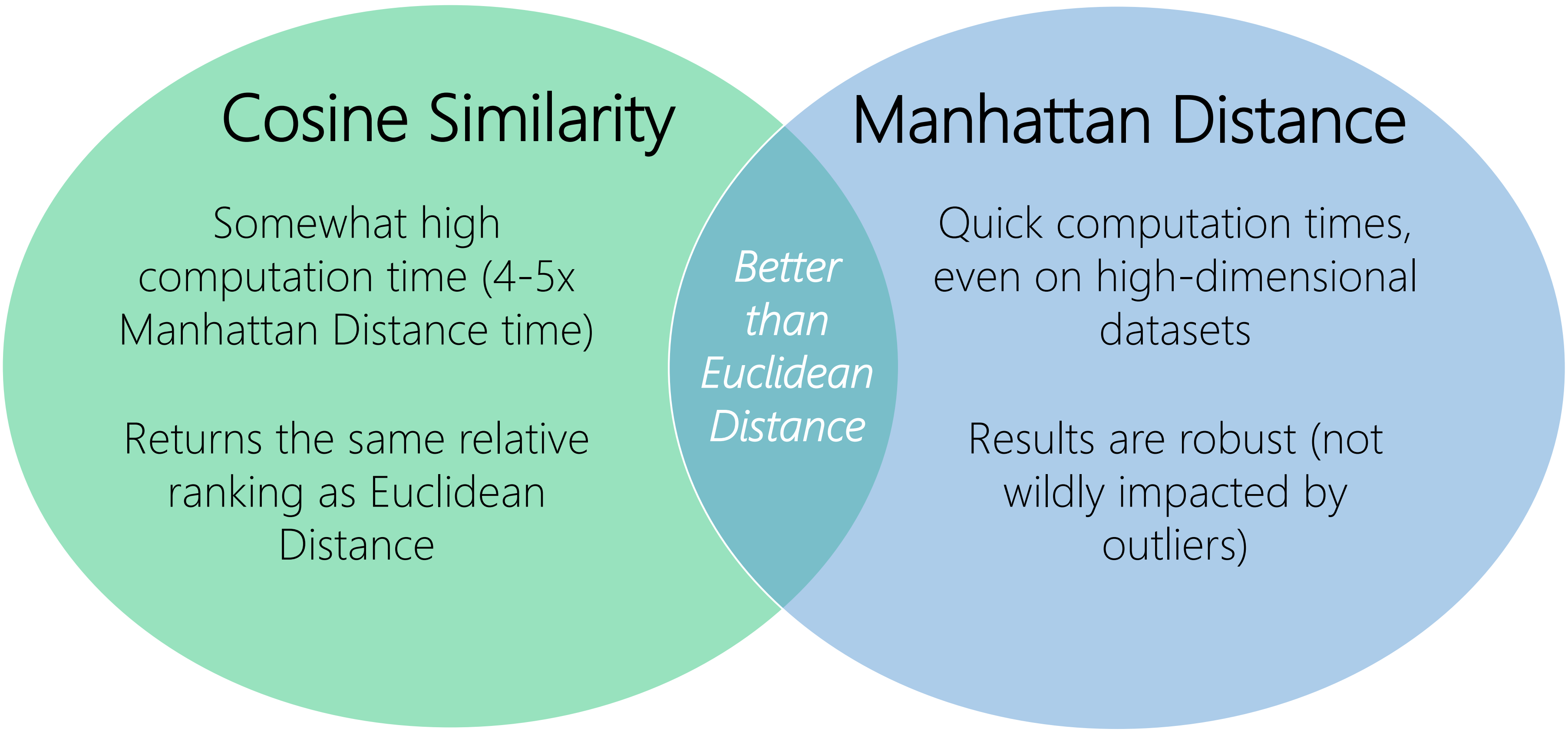


Cosine Similarity



Our Recommendation Algorithm:
What It Does and How It Works

We Built Two Different Recommendation Functions



Our Data Preprocessing Steps

Drop Unnecessary Columns

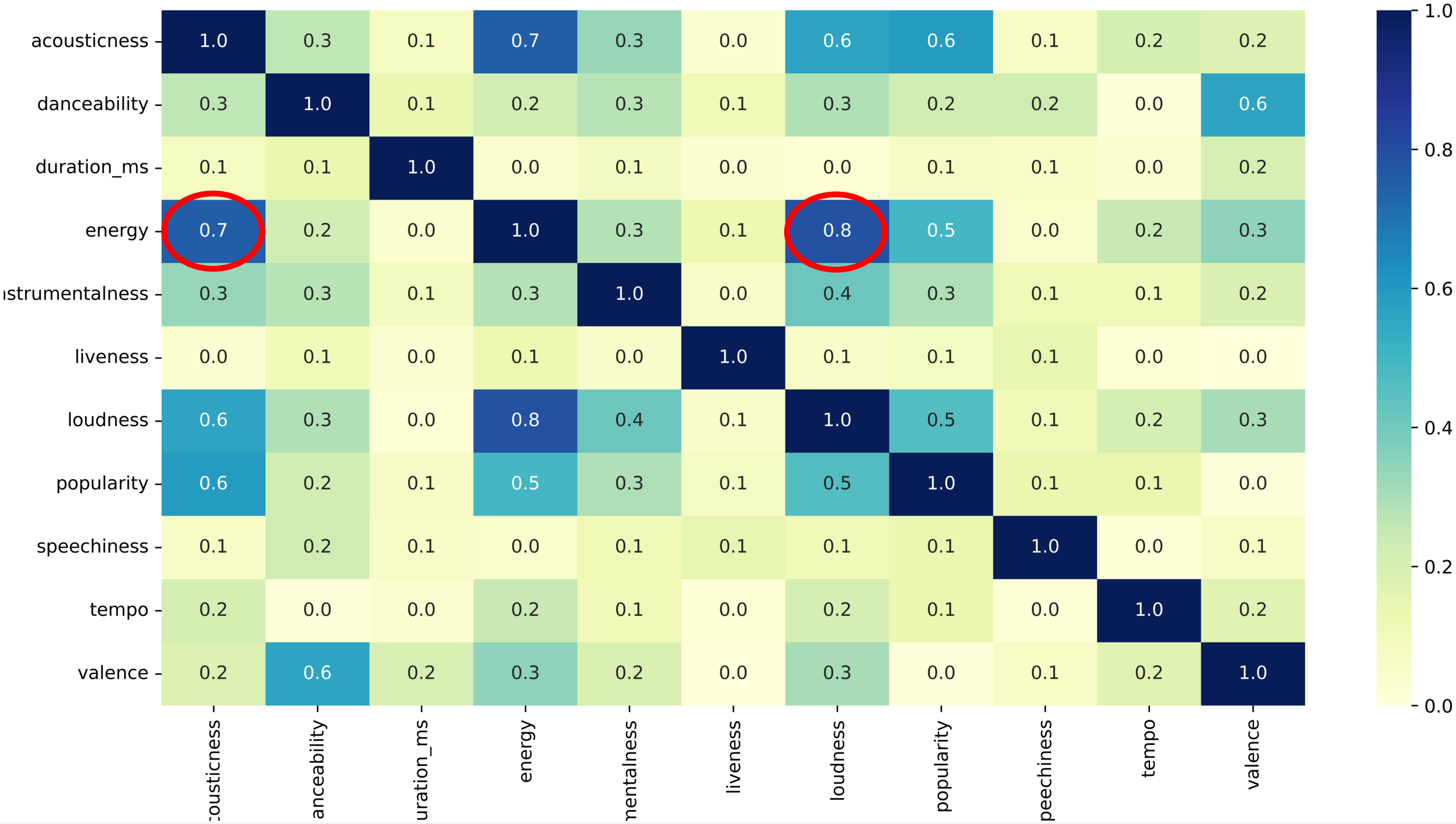
Drop Duplicate Data Entries

Convert Categorical Variables

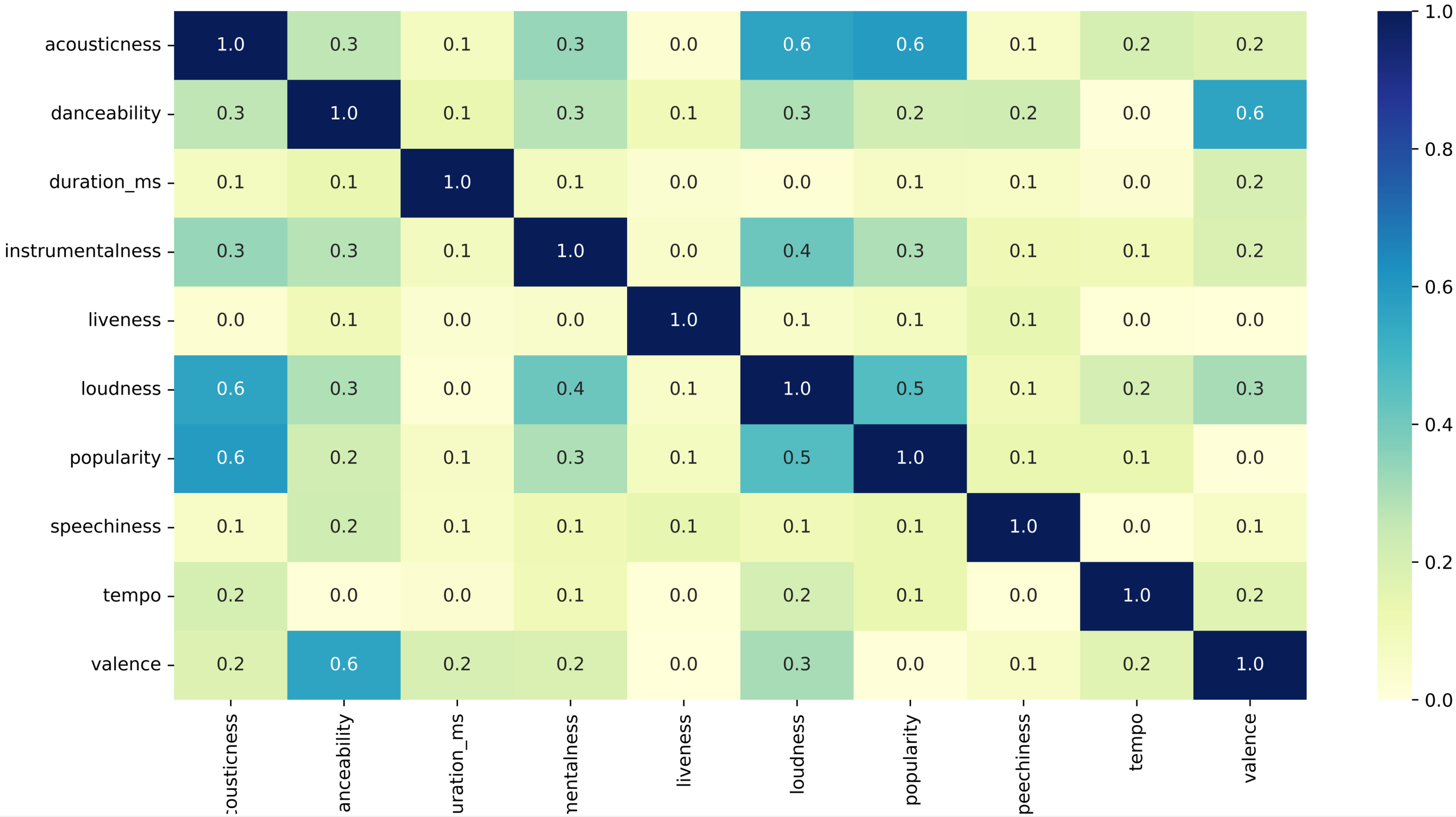
Min-Max Scaling to Normalize

End Result: a Clean Dataset

Dealing with High Correlations Between Features



After Dropping the Energy Column

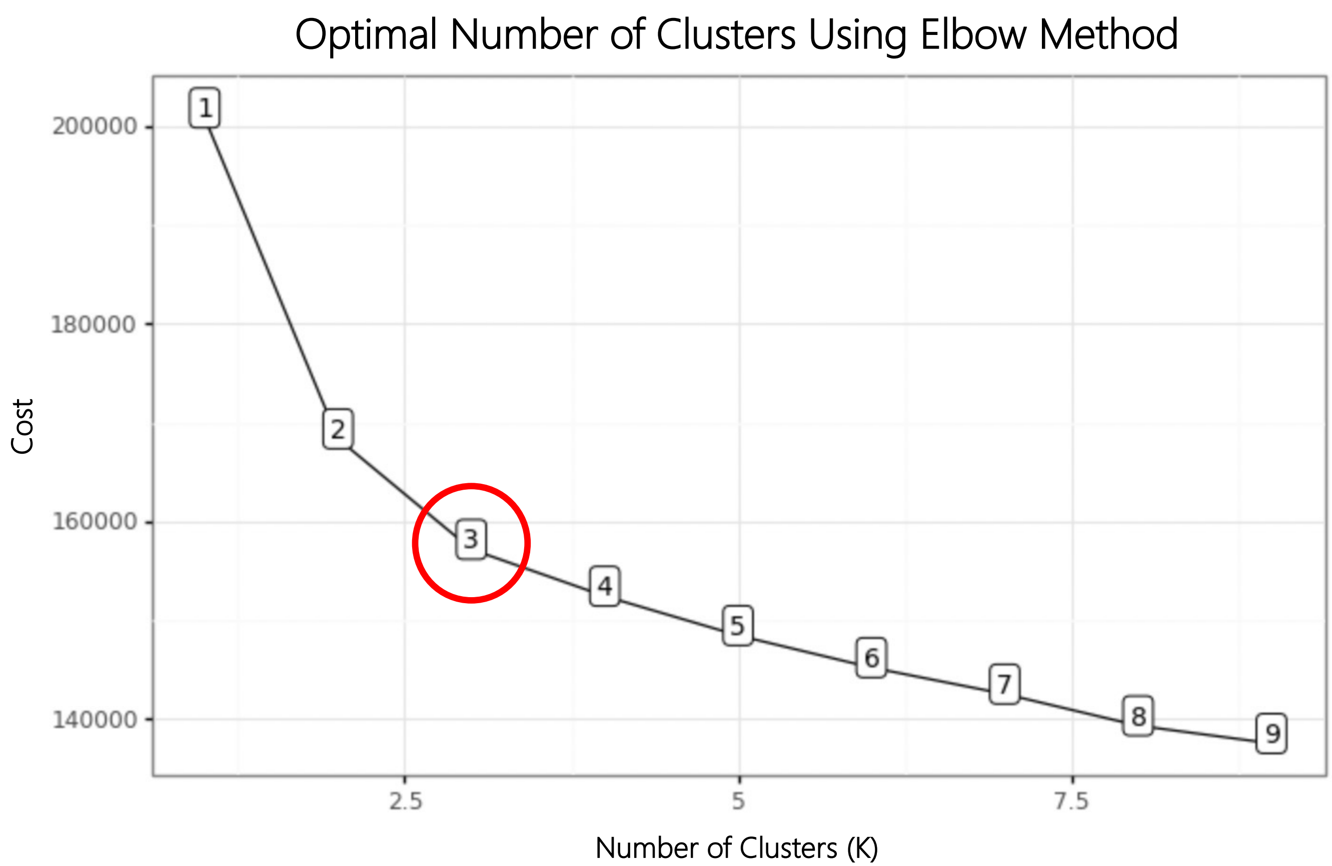


Exploring Our Data: KPrototypes Clustering

We chose to use KPrototypes clustering for our data to improve the computation speed and accuracy of the song recommendation functions.

	Accepts Numerical Data	Accepts Categorical Data	Time Complexity	Interpretability
KMeans	✓	✗	Decent	Very Good
KModes	✗	✓	Very Good	Unusable
KPrototypes	✓	✓	**Awful**	Decent

Choosing the Correct Number of Clusters

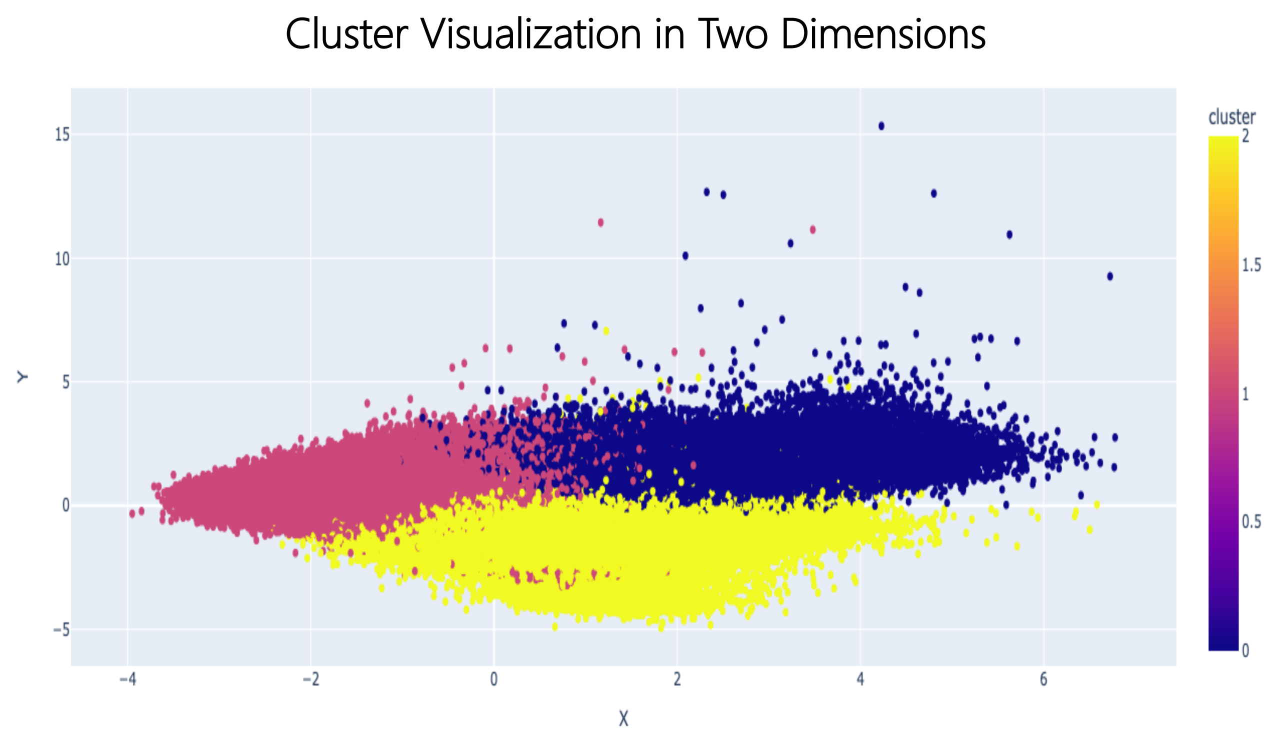


Visualizing Our Clusters Using PCA

0 Cluster 0: classical music, symphonies, podcasts, radio, and white noise

1 Cluster 1: music those using our recommender are likely to listen to

2 Cluster 2: older music and songs in non-English languages



How Does Our Algorithm Perform?

You Be The Judge! Song Recommendations for Classmates

A	B	C	D
Until You Come Back to Me – Aretha Franklin Feel Like I Do – Disclosure I Like It – DeBarge Ultimate – Denzel Curry Blue World – Mac Miller	Streetcar – Daniel Caesar Nights – Frank Ocean Die for You – The Weeknd Open – Kehlani Garden – SZA	More – Chief Keef Love Sosa – Chief Keef 15 – Taylor Swift Going Bad – Drake Poundcake – Drake/Jay-Z	Sunflower – Post Malone Beautiful People – Ed Sheeran I Don't Care – Ed Sheeran As Long As You Love Me – Justin Bieber Scared of the Dark – Lil Wayne
Manhattan Right Now – Lil Uzi Vert Yours Truly – Post Malone Cosine For However Long – Bryson Tiller Without Me – Halsey, ILLENIUM	Manhattan Love Don't Change – Jeremih To Die For – Kygo, St Lundi Cosine Hurting – Kygo The Worst – Jhene Aiko	Manhattan Cocoa Butter Kisses – Chance the Rapper Broken Clocks – SZA Cosine Clique – Kanye West 3500 – Travis Scott	Manhattan Graveyard – Halsey Like I Can – Sam Smith Cosine Reborn – Kids See Ghosts Canyon Moon - Harry Styles

Get Your Own Custom Song Recommendations!

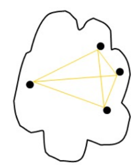
Using Google API, we can import data from our Google Sheet to our Colab File and automate the recommendation system!

Measuring Accuracy Quantitatively is Difficult Without True Labels

- Common error metrics such as MAE, RMSE, MSE require there to be a **predicted label** and an **actual label**.
  - Because our dataset only contains information regarding the songs/artists/albums, we were unable to product any meaningful quantitative accuracy measure.

- However, given more resources, there are some steps we could take to create an accuracy system:

1. Internal Validation
- Cohesion across samples
- Compute the similarity between several records via 'Euclidian distance' in featurespace



cohesion

$$\text{Cohesion}(C_k) = \sum_{x \in C_k;\, y \in C_k} \text{similarity}(x, y)$$

2. External Validation / Twin Sample Validation
- Creating a sample of records that is expected to exhibit similar behavior to the training data
- Perform unsupervised learning on the 'twin-sample'
- Compare similarity between two sets results
  - F1, Jaccard Similarity.... any kind of External Validation method

Additional Topics for Further Discussion and Improvement

Possible Additions and Improvements to Our Project

Up-to-Date Song Dataset

1. Our current dataset stops at 2020. We looked into using Spotify's API to pull up-to-date song data, but found it was more trouble than it was worth.

Web-Based Front-End

2. Would allow users to input their songs and receive real-time results through a sleek user interface. Unfortunately, we couldn't get it working in time.

Artist Recommender

3. Similar to the song recommender. It would use naïve weighting for each artist and artist-relative popularity for each artists' songs.

Possible Ways to Improve the Underlying Algorithm

We feel our algorithm performs **as best as can reasonably expected** in most general-case scenarios, considering we used unsupervised learning to generate recommendations from imperfect data.

A few suggestions for potential improvements we could make to the underlying algorithm are:

An alternative, numerical representation of the 'key' feature

1 **Significant Loss of Interpretability for a Negligible Improvement in Recommendation Accuracy**

Choosing a different number of clusters

2 **Choosing 3 Clusters was 'Optimal' According to the Elbow Method**

Filtering Dataset to Only the 10,000 Most Popular Songs

3 **We Would Miss Out on Recommending Similar Songs that are Less Popular**

Thank You!

Any Questions?