Spotify's Top 200 Charts:
Unsupervised Learning & Recommendations

The Group Chat
Members: Nate, Lisa, Geo, Sam, Mateo

Overarching Question

Using the songs that entered weekly top 200 charts between 2021 to 2022 from the Spotify API we ask the question:

Is there a relationship between audio features and, if so, how are they correlated?

Additionally, how can we use these features to cluster songs based on audio tracks of songs represented by numeric features?

Hypothesis

The distribution of certain genres will vary among different groups of songs. These differences in distributions will allow us to perform unsupervised learning on the data to cluster the songs into different groups/listening personas.

For example, the mean tempo of Pop artists will be higher than that of Ballad singers since Pop songs tend to be more upbeat and fast.

If numerical data is extracted from the songs then models can be trained to cluster/classify songs into different groups since there will be enough difference between certain features. This approach of comparing audio features between two groups can then be applied to other projects, such as comparisons of the audio of living entities to classify them.

01

Exploratory Data Analysis

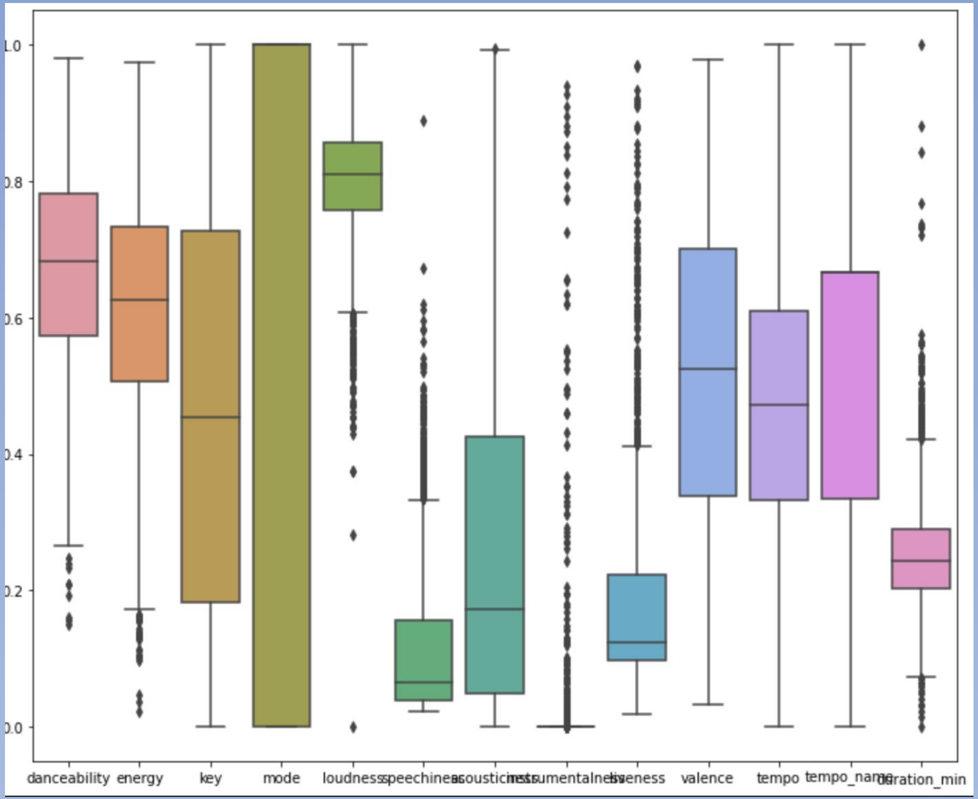
Correlation Between Features

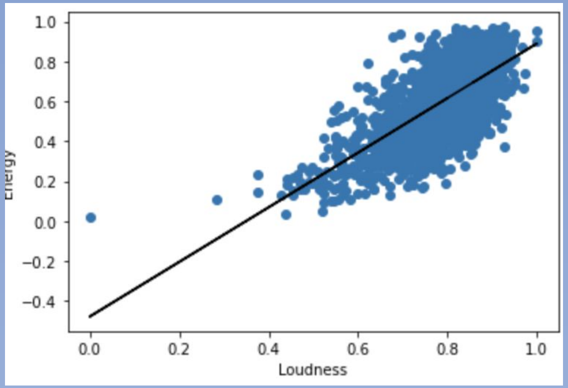
Correlation Heatmap of Audio Features

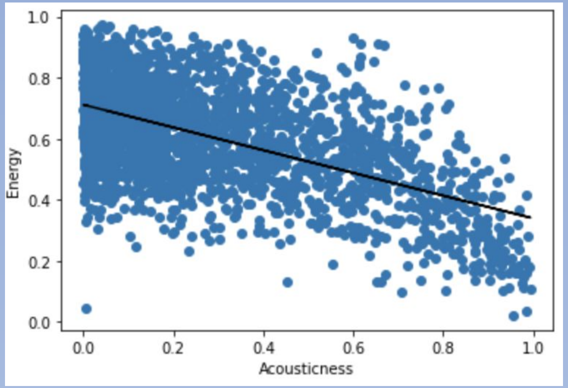
	danceability	energy	key	mode	loudness	speechiness	acousticness	instrumentalness	liveness	valence	tempo	duration_min
duration_min	-0.16	0.04	-0.02	0.03	0.02	-0.07	-0.01	0.02	0.01	-0.14	-0.0	1.0
tempo	-0.1	0.08	-0.01	0.02	0.05	0.07	-0.07	-0.02	-0.01	0.03	1.0	-0.0
valence	0.34	0.34	0.07	0.02	0.27	0.01	-0.06	-0.08	0.03	1.0	0.03	-0.14
liveness	-0.08	0.13	0.0	0.04	0.06	0.04	-0.05	-0.0	1.0	0.03	-0.01	0.01
instrumentalness	-0.1	-0.06	0.02	-0.0	-0.19	-0.07	0.06	1.0	-0.0	-0.08	-0.02	0.02
acousticness	-0.27	-0.58	-0.03	0.07	-0.44	-0.13	1.0	0.06	-0.05	-0.06	-0.07	-0.01
speechiness	0.23	0.05	0.04	-0.12	-0.03	1.0	-0.13	-0.07	0.04	0.01	0.07	-0.07
loudness	0.24	0.7	0.05	-0.02	1.0	-0.03	-0.44	-0.19	0.06	0.27	0.05	0.02
mode	-0.14	-0.05	-0.14	1.0	-0.02	-0.12	0.07	-0.0	0.04	0.02	0.02	0.03
key	0.06	0.07	1.0	-0.14	0.05	0.04	-0.03	0.02	0.0	0.07	-0.01	-0.02
energy	0.18	1.0	0.07	-0.05	0.7	0.05	-0.58	-0.06	0.13	0.34	0.08	0.04
danceability	1.0	0.18	0.06	-0.14	0.24	0.23	-0.27	-0.1	-0.08	0.34	-0.1	-0.16

1
0.5
0
−0.5

Distributions & Correlations of Audio Features







02

Data Cleaning

Missingness Analysis

Since we used the Spotify API there was no missing data. However we did need to standardize the numeric features.

We verified this by using Pandas to count the number of NaN values

Check for missingness

```
for col in charts.columns:
    num_NaN = charts[col].isna().sum()
    print("Are there any NaN's?")
    if num_NaN == 0:
        print('No')
```

```
Are there any NaN's?
No
Are there any NaN's?
No
Are there any NaN's?
No
Are there any NaN's?
No
Are there any NaN's?
No
Are there any NaN's?
No
Are there any NaN's?
No
Are there any NaN's?
No
Are there any NaN's?
No
Are there any NaN's?
No
Are there any NaN's?
No
Are there any NaN's?
No
Are there any NaN's?
No
Are there any NaN's?
No
Are there any NaN's?
No
```

How we chose variables:

- **PCA?** Not good for categorical, did not give us significant and easily interpretable components.
- **Solution:** Correlation Threshold

Why a threshold?

- Easy to access and evaluate
- Pipeline objects allow for easy implementation due to custom transformers if we decide to convert to supervised

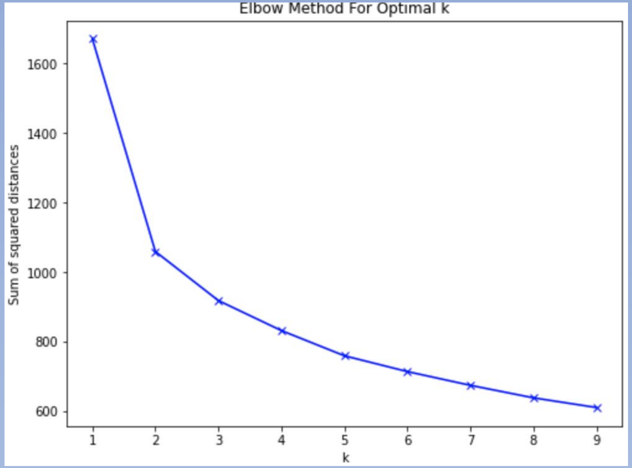
03

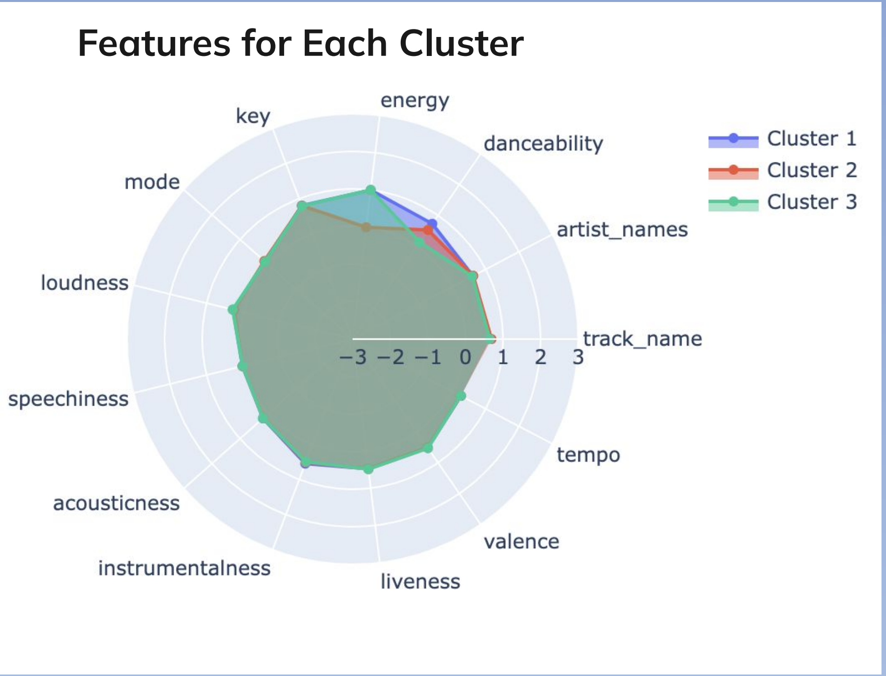
Different Models/ Approaches

K-Means Clustering

High level concept: creating k clusters where the center is a circular shape and points are assigned to their closest centroid.

Through the elbow method, we found that 3 was the best hyperparameter.





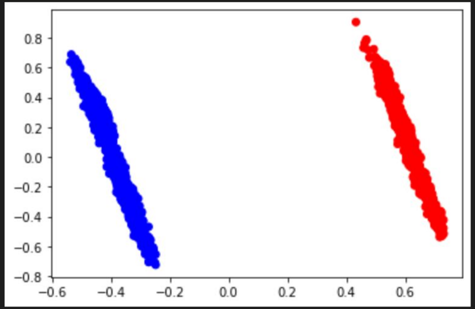
Gaussian Mixture Model

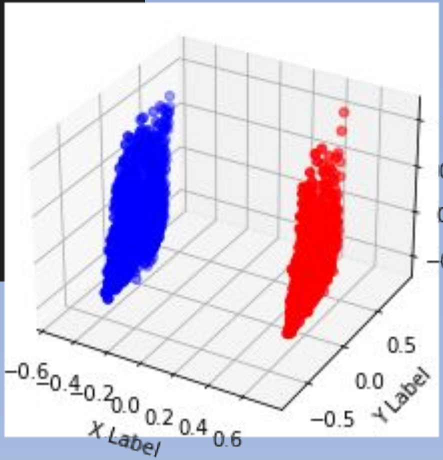
High level concept: assuming the data set represents multiple Gaussian (normal) distributions, clustering based on each point's probability of belonging to each distinct distribution. Soft clustering method

Using the optimal K, we observed 2 distinct groups, meaning 2 different distributions were discovered once the data was transformed by principal component analysis.

The lower dimensionality still portrays 72% of the variance due to the 3 principal components.

```
plt.scatter(df_pca['PC1'], df_pca['PC2'], c=labels, cmap = 'bwr')
plt.show()
```





Silhouette Score = .53

K-Nearest Neighbors

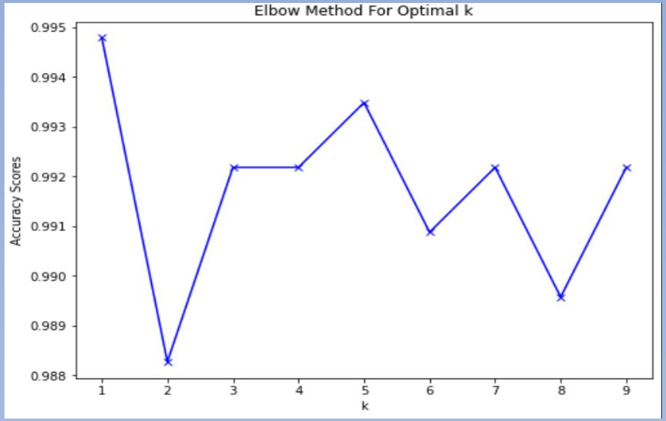
High level concept:

Supervised learning by calculating the distance between a given data point and its k-nearest neighbors in the training dataset.

Then, the model makes predictions based on the majority class **or** the average value of the nearest neighbors.

We performed k-nearest neighbors clustering with train-test split of 0.3 for the test data.

- > 0.99 accuracy for nearly all values of nearest neighbors.
- dataset is imbalanced and possibly overfit the model.
- high accuracy could be explained by the fact that the clusters themselves were very similar, with the data points within each cluster being tightly packed together.



Hard vs. Soft Clustering

Hard Clustering (K-means/KNN): each point belongs to one centroid or cluster.

Soft Clustering (GMM): each point belongs to multiple clusters with weights since it is a distribution.

What does this mean?

We used hard clustering and soft clustering methods on the off chance that the points would be close enough to be closely related to multiple clusters. However, it was unnecessary. In a scenario such as belonging to playlists based on genre, it would be useful to use soft clustering for multi-labeling tasks.

Evaluation:

With no ground truth label and with an unsupervised method such as clustering, an empirical evaluation is difficult to depict.

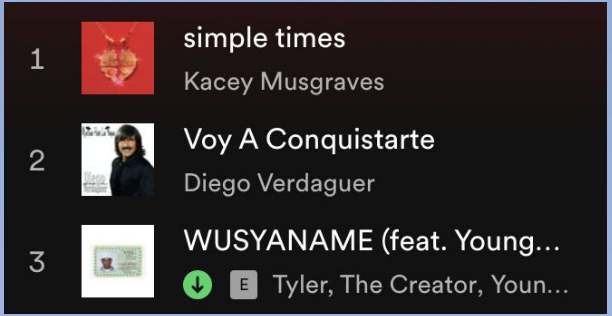
- Gaussian Mixture model: the silhouette score was used as a mathematical evaluation of its performance.
- K-Means: sum of squared distances was used to find optimal k
- KNN: accuracy was the main metric for clustering performance, but also included recall, precision, f1-score, and AUC-ROC to show that the model scored high in all areas.

At the end of the day we have to answer the question ourselves, did the clustered songs make sense and were they enjoyable? Could we use this to make a playlist we would listen to? This is why we tested ourselves by picking a few songs from a cluster made by each method to listen to and decide.

Thoughts: Both playlists from the GMM model sound pretty good, they don't fit into a genre or central idea but they encapsulate a good group of songs.

GMM playlists:
cluster 0
cluster 1

GMM Cluster 1



K-Means Cluster 1

K-Means Cluster 2

K-Means Cluster 3

Where do we go next?

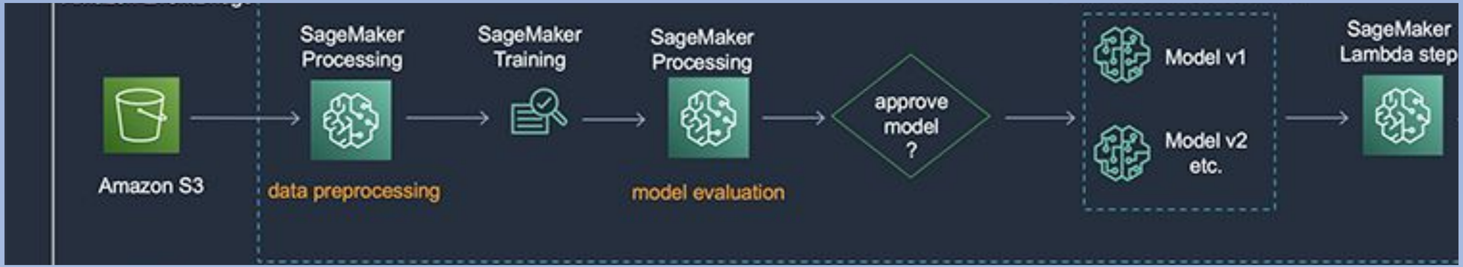
- Deploy it on the cloud
- Create a pipeline to feed newly charting songs

Repeat the cycle as time goes on.

What new questions can we ask?

Mainly it begs two questions:

- Would user interaction data make better clustering decisions?
- Could we make ground truth labels? Such as songs originally belonging to a playlist or based on genre.



Conclusion

It is hard to quantify the effectiveness of a model with no ground truth labels but it comes down to, do we like the results from the model?

Yes, the models are not horribly bad and the small playlists we created from them were enjoyable. There is much to be done to improve the cohesiveness of the song selection but for a first try it was a success.