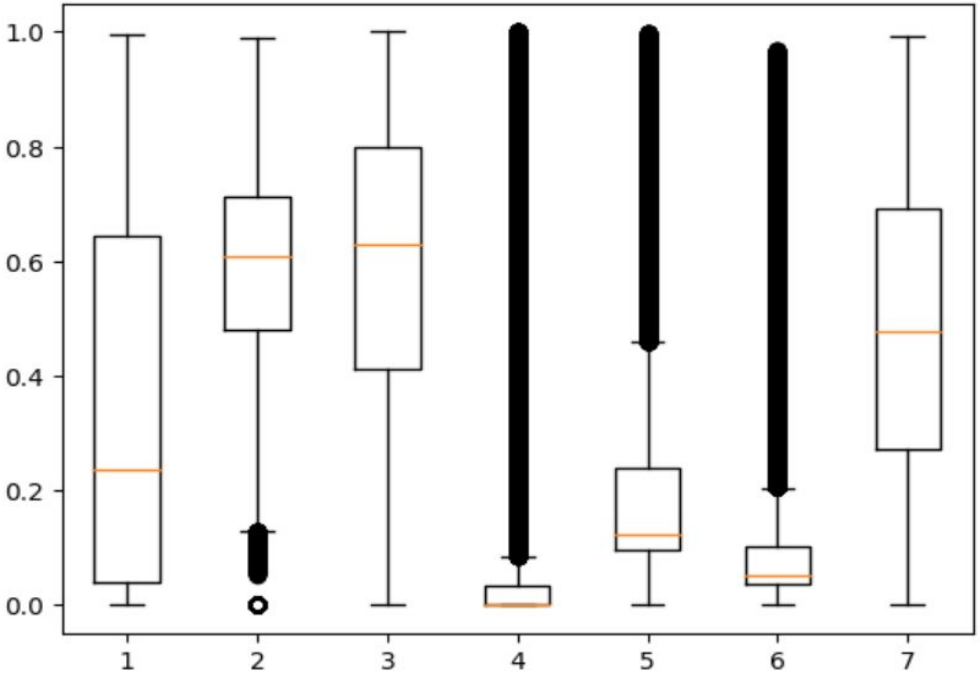
Spotify Song Recommendation

Akshu E, Ian W, Rinah Z, Parav Manney

Data Exploration: Box Plot

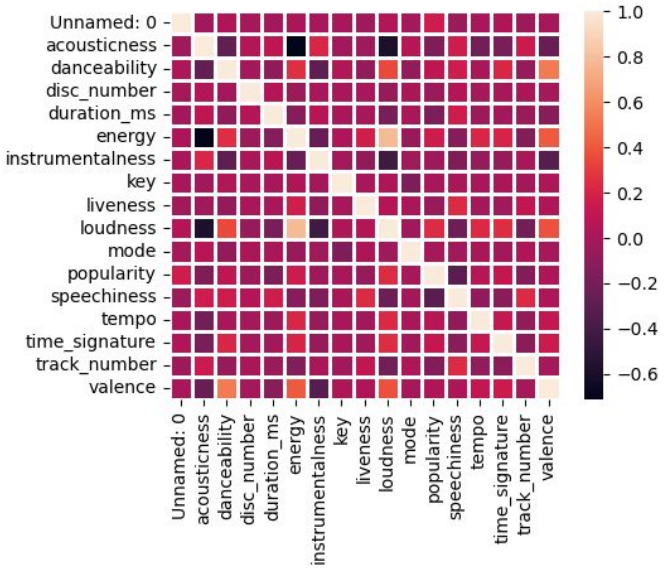


Data Exploration: Correlation between Variables

	acousticness	danceability	energy	instrumentalness	liveness	speechiness	valence
acousticness	1.000000	-0.140333	-0.680052	-0.017341	-0.100606	-0.141853	-0.139961
danceability	-0.140333	1.000000	0.108619	-0.048337	-0.094379	0.207628	0.441894
energy	-0.680052	0.108619	1.000000	-0.015609	0.172962	0.195137	0.331110
instrumentalness	-0.017341	-0.048337	-0.015609	1.000000	-0.027068	-0.058977	-0.112023
liveness	-0.100606	-0.094379	0.172962	-0.027068	1.000000	0.034053	0.026892
speechiness	-0.141853	0.207628	0.195137	-0.058977	0.034053	1.000000	0.090648
valence	-0.139961	0.441894	0.331110	-0.112023	0.026892	0.090648	1.000000

Correlation Matrix (w/Colors + Insights)

- Close to 1:
  - Energy and Loudness (>0.75)
- Mostly moderate/low for other variables
- Close to -1:
  - Popularity and Speechiness
  - Energy, Acousticness, and Loudness



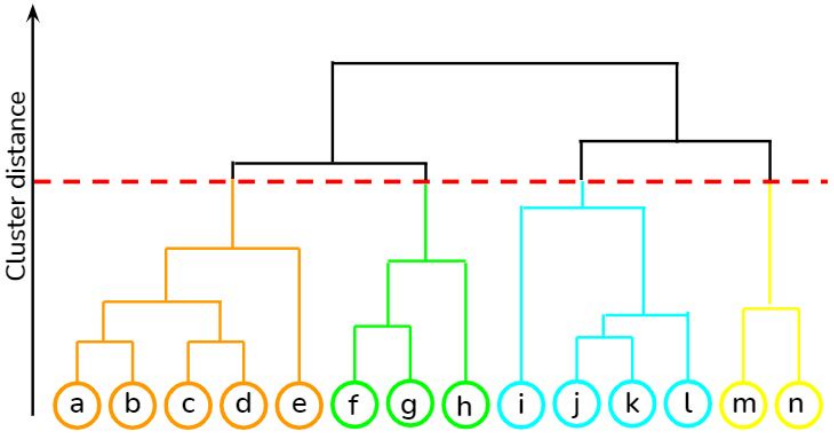
Handling Outliers

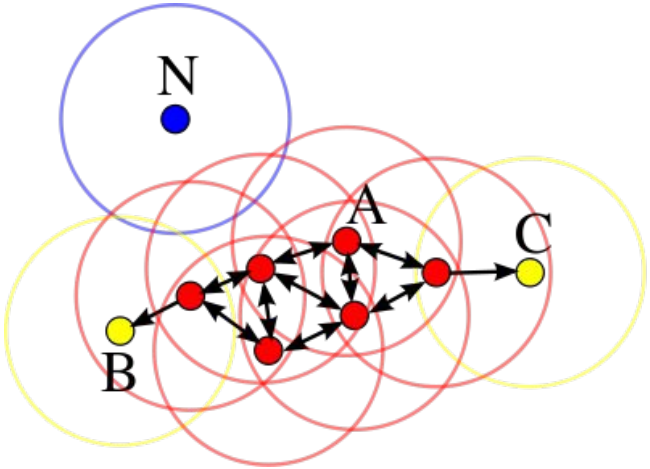
```
def remove_outliers(data, threshold=2):
    # Calculate the Z-scores for the data
    z_scores = stats.zscore(data, axis=0)

    # Find the indices of the data points that have a Z-score greater t
    outlier_indices = np.where(np.abs(z_scores) > threshold)


    # Remove outliers from the dataset
    data_filtered = data[(np.abs(z_scores) < threshold).all(axis=1)]

    return data_filtered

remove_outliers(tracks).describe()
```

- Using Z-scores to identify outliers and remove them.

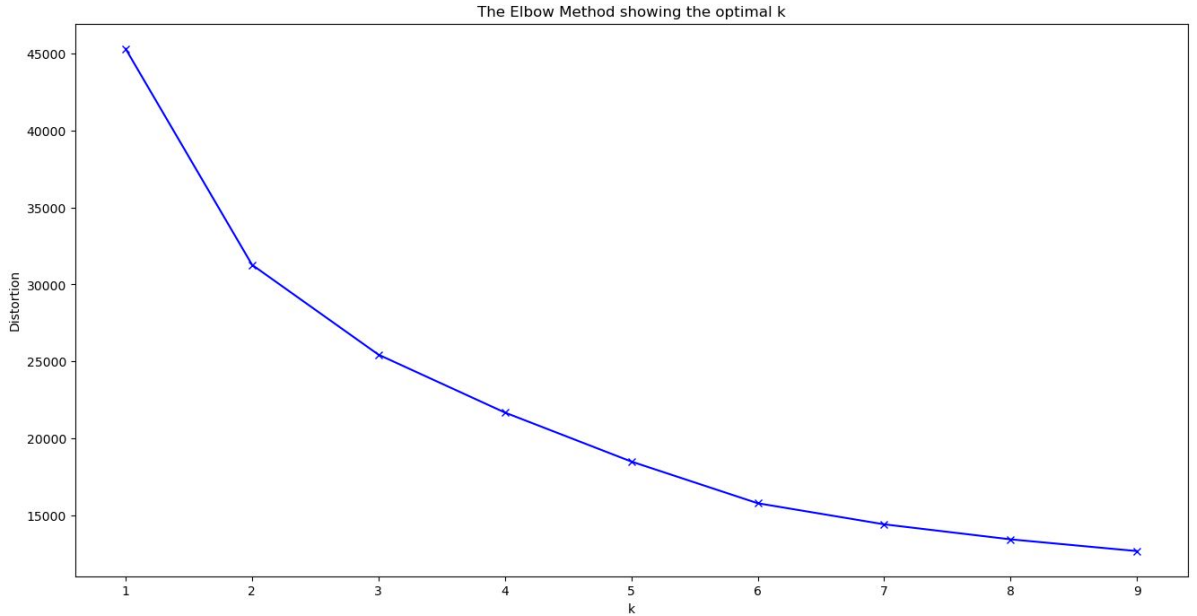
Comparing against other algorithms:

- Hierarchical Clustering
- DBSCAN





Optimal # of Clusters



User Interaction: Multiple Recommendations

```
How many song suggestions would you like to see?
> 4
   Unnamed: 0  acousticness               album_id  \
1           1         0.863  1bcqsH5UyTBzmh9YizdsBE
2           2         0.750  4tKijjmxGClg4JOLAyo2qE
3           3         0.763  6FeJF5r8roonnKraJxr4oB
4           4         0.770  4tKijjmxGClg4JOLAyo2qE

                                        analysis_url  \
1  https://api.spotify.com/v1/audio-analysis/3VAX...
2  https://api.spotify.com/v1/audio-analysis/1L3Y...
3  https://api.spotify.com/v1/audio-analysis/6aCe...
4  https://api.spotify.com/v1/audio-analysis/1Vo8...
```

User Interaction: Feedback

```
Would you like to provide some feedback on the features of the reccomended songs we provided?(yes/no)
yes
for ['4xWMewm6CYMstu0sPgd9jJ'] how would you rate these features? On a scale from 0-1
acousticness:0.3
0.3
dancebility0.6
0.6
energy:0.1
0.1
instrumentalness:0.04
0.04
liveness:0.7
0.7
speechiness:0.3
0.3
valence:0.5
0.5
For the song by ['4xWMewm6CYMstu0sPgd9jJ'] would you like to provide any feedback, so we can improve future suggestions?
Feedback (optional):
>
Feedback for ['4xWMewm6CYMstu0sPgd9jJ']: Feedback -

Thank you for providing feedback!
Would you like to provide some feedback on the features of the reccomended songs we provided?(yes/no)
no
```

Future Enhancements

- Playing around with number of components
  - PCA to reduce dimensions: Increased efficiency (runtime) while maintaining reliability
  - Exploring other components (language, time period, etc.)

- Improving user interaction
  - Creation of a playlist directly on Spotify with top recommended songs
  - Contextual recommendations (based on time of day, mood, location)

Impact and Applications of Findings

- Benefits of algorithm:
  - Personalized experiences, more customers, business growth
  - More user engagement and new music discovery

- Applying recommendations to other fields:
  - Netflix (recommending shows based on ratings, stats, and more)
  - Amazon (recommending products in an industry based on recent purchases)
  - Google (recommending activities/places based on past searches)