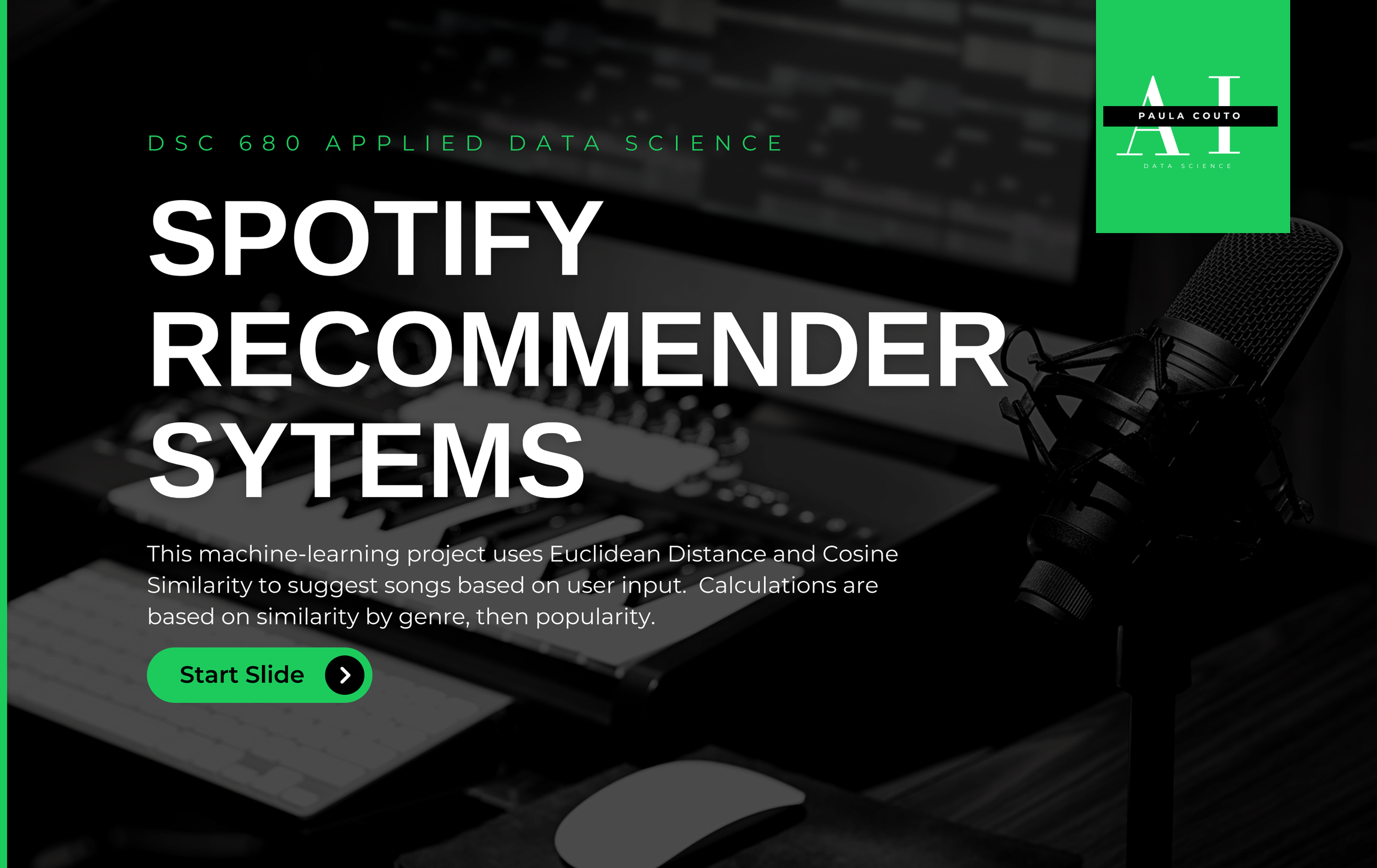


PAULA COUTO

DATA SCIENCE

DSC 680 APPLIED DATA SCIENCE

SPOTIFY RECOMMENDER SYTEMS

This machine-learning project uses Euclidean Distance and Cosine Similarity to suggest songs based on user input. Calculations are based on similarity by genre, then popularity.

Start Slide

Check out my DataSciencePortfolio

BUSINESS PROBLEM

Streaming services are most likely to retain their customers if they are able to showcase the extent of their catalog by providing selections of products that they already know their consumers like. By gathering information from user behavior such as search, plays, repeats, and skips, their machine learningmodel is able to identify their preferences.

Streaming services use machine learning algorithms to personalize their user's experience within the app and increase their retention.

BACKGROUND

When user data is not available such as when the listener is new to the subscription, it’s a critical time for a streaming company to capture their audience’s attention. In this scenario, the most ideal way to suggest songs to listen to,is for the streaming service to use the search function as a reference point to find other music similar to what the customer is looking for.

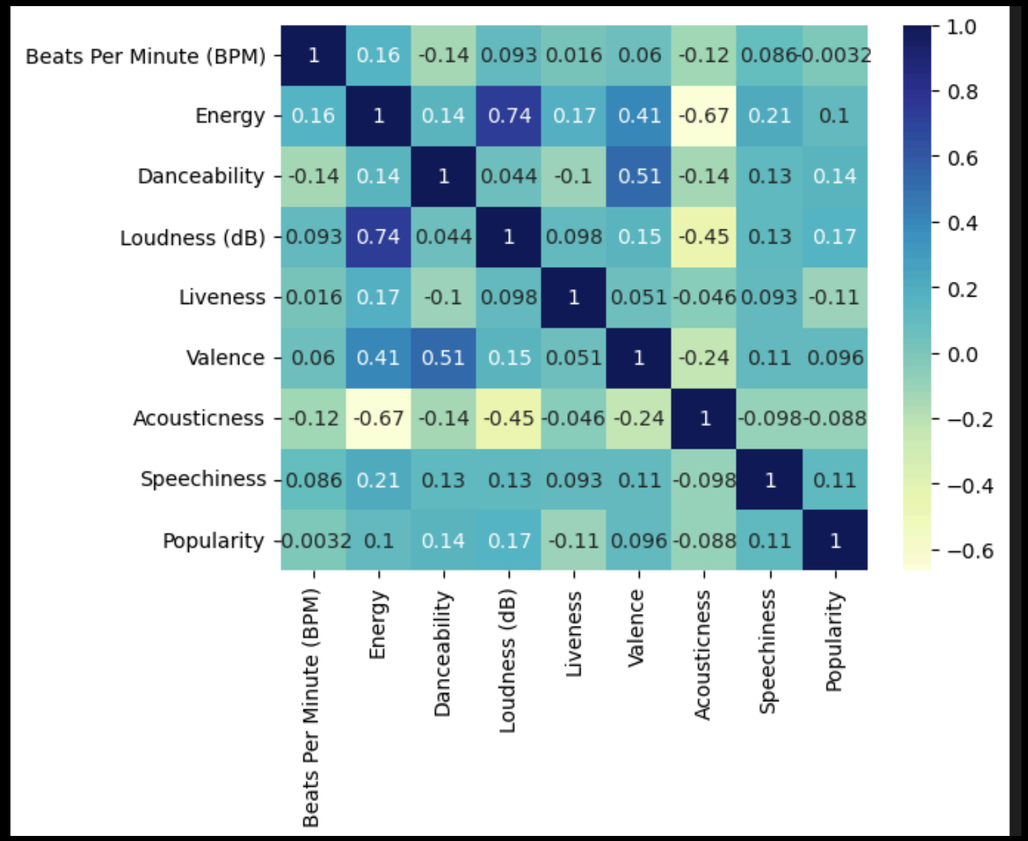
DATA SOURCE

The dataset used in this project, Spotify_Top_2000.csv is a preprocessed dataset from Kaggle. It provides information about the most popular songs in Spotify from 1956 to 2019.

Unique Songs
1,994

Features
13

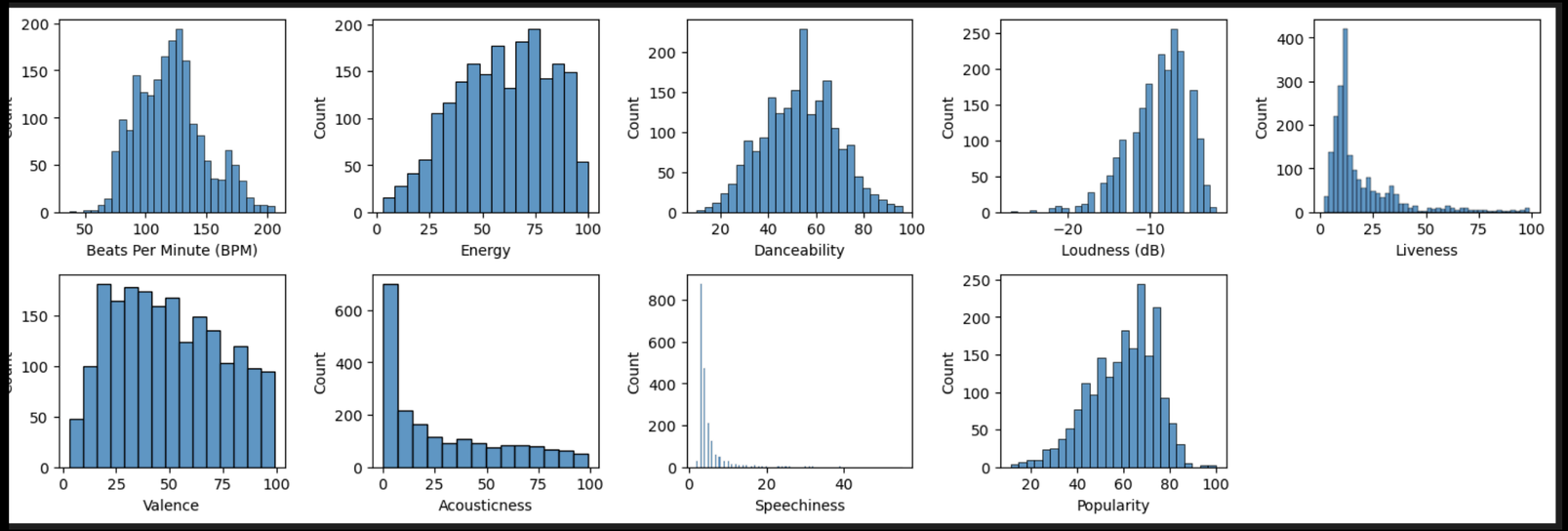
In this project, we are using features that make up each song to calculate the best musical offerings from the top 1994 songs in the database that are similar to their song input. To further analyze the dataset for feature reduction, I used the correlation matrix to see if any of the columns are related enough that one could represent the other. In reducing dimension, setting a threshold for the correlation score allows us to keep important data without losing insights even if we had dropped some of our variables.



Next Slide

MUSIC PROFILE

In an effort to understand the features of the dataset, I used a histogram to understand their distribution.



MODEL IMPLEMENTATION

Music Company

EUCLIDEAN DISTANCE

Euclidean Distance is defined as the distance between two points in Euclidean space. To find the distance between two points, the length of the line segment that connects the two points should be measured.

SOURCE: GeeksforGeeks

OUTPUT

```
[19]  input_song= "The Pretender"
      recommended = recommender_feat_dist(input_song, matrix_df)
      print(recommended)

...   ['The Pretender', 'Everlong', 'Epic', "Don't Look Back In Anger - Remastered", 'Sabotage']
```

MODEL IMPLEMENTATION

COSINE SIMILARITY

The cosine similarity is beneficial because even if the two similar data objects are far apart by the Euclidean Distance because of the size, they could still have a smaller angle between them. The smaller the angle, higher the similarity.

SOURCE: GeeksforGeeks

```python
def spotify_recommender_genre(input_song, reference):
    input_transform = genre_vector.transform(reference[reference['Title']==input_song]['Top Genre']).toarray()
    input_array = reference[reference['Title']==input_song].select_dtypes(include=np.number).to_numpy()

    sim = []
    for idx, row in reference.iterrows():
        name = row['Title']

        rec_transform = genre_vector.transform(reference[reference['Title']==name]['Top Genre']).toarray()
        rec_array = reference[reference['Title']==name].select_dtypes(include=np.number).to_numpy()

        text_sim = cosine_similarity(input_transform, rec_transform)[0][0]
        feature_sim = cosine_similarity(input_array, rec_array)[0][0]
        sim.append(text_sim + feature_sim)

    return sim
```

```python
def find_similar_songs(input_song, reference=rec_df1):
    if rec_df1[rec_df1['Title'] == input_song].shape[0] == 0:
        print('This song is not on our database\
        Check out these other recommendations:\n')

        for song in reference.sample(n=7)['Title'].values:
            print(song)
        return

    reference['similarity_score'] = spotify_recommender_genre(input_song, reference)

    reference.sort_values(by=['similarity_score', 'Popularity'],
                          ascending = [False, False],
                          inplace=True)

    print(reference[['Title', 'Artist','Top Genre']][:7])
```

Next Slide

MODEL IMPLEMENTATION OUTPUT

COSINE SIMILARITY

The cosine similarity is beneficial because even if the two similar data objects are far apart by the Euclidean Distance because of the size, they could still have a smaller angle between them. The smaller the angle, higher the similarity.

SOURCE: GeeksforGeeks

```
find_similar_songs(input_song)
```

	Title	Artist	Top Genre
3	The Pretender	Foo Fighters	alternative metal
1723	Everlong	Foo Fighters	alternative metal
1484	Epic	Faith No More	alternative metal
357	How You Remind Me	Nickelback	alternative metal
1583	Plush - 2017 Remaster	Stone Temple Pilots	alternative metal
1814	The Dolphin's Cry	Live	alternative metal
89	Times Like These	Foo Fighters	alternative metal

Next Slide

ETHICAL CONSIDERATIONS

An important ethical concern that needs to be addressed is the use of consumer data for retention. While keeping customer subscription by increasing usage is a critical part of business strategy, we have to be aware of how information gathered from consumers is used manipulate their listening habits. Even though Spotify's rates do not change, regardless of how many times a customer logs in or how many songs they listened to-this type of information used on sales or pay-per-use categories has the potential to be predatory. Users must not only be made aware of this but there should be systems set in place to prevent the abuse of such systems

Music
Company

DSC680 Applied Data Science

END

End Slide