USING CLUSTERING TO PREDICT THE MUSIC GENRE OF SONGS WITH THE GIVEN FEATURES OF THE DATASET

INTRODUCTION TO THE PROBLEM

The problem is to find if it is possible to accurately cluster groups of songs together based on their genre without knowing what there genre is. The question I am trying to solve is will the using clustering be an accurate way to predict the genre of a song? Is clustering a useful way to conduct unsupervised learning?

WHAT IS CLUSTERING AND HOW DOES IT WORK?

"Clustering is an unsupervised machine learning algorithm that organizes and classifies different objects, data points, or observations into groups or clusters based on similarities or patterns. There are a variety of ways to use clustering in machine learning from initial explorations of a dataset to monitoring ongoing processes."Source: : https://www.ibm.com/topics/clustering

INTRODUCTION OF THE DATASET

The data set is taken from kaggle from user 'vicsuperman'. The data set include columns: instance_id, artist_name, track_name, popularity, acousticness, danceability, duration_ms, energy, instrumentalness, key, liveness, loudness, mode, speechiness, tempo, obtained_date, valence, music_genre. Columns deemed neccesarry to predict what genre of music a song is were: 'acousticness', 'danceability', 'energy', 'instrumentalness', 'liveness', 'loudness', 'speechiness', 'tempo', 'valence'. These characteristics were labeled as 'features' in the code. The 'music_genre' column was used to test the accuracy of using the clustering method for guessing the music genre of different songs.

DATA UNDERSTANDING/VISUALIZATION

The data that is used to conduct the analysis is made up of different parts. These parts are features that can be used to help identify the genre of a song, charateristics that don't help identify the genre of a song, and the genre of the song. Using the features that can help identify the genre of a song a model will be created to predict the genre. Then the result of that model will be tested against the real result, the genre. More information about the creation of the model will be in 'Modeling' and more information about the results will be in 'Storytelling'. The resulting visualization of the graph shows that it is quite hard to predict the genre based soley off of something like how loud a song is and how much energy there is to it. However, Some genres were predicted to a higher success than expected this graph doesn't show that. What shows the accuracy is the classification report.

PRE-PROCESSING THE DATA

To preprocess the data all null values were dropped from the dataset as any rows containing with null would mess with the output. Some columns had data as a '?' instead of a number which would cause issues with inputing the data into the model so that data was converted

to N/A before dropping of N/A values took place so all of these unusable rows could be removed in one go.

MODELING (CLUSTERING)
To create the model used to predict the genre of different songs K-means clustering was used. This makes sense since the exact number of genres a song can be is known. There are 10 genres in the dataset to choose from so we have predetermined 10 clusters for the K-means clustering to be conducted on. The model is fitted on a list of scaled features which are the features deemed neccessary to predict the genre. Scaled here means that the features are transformed so they are readable and uniform when input into the model. The results of the model are grouped by a new column 'predicted_genre' and are displayed in the calssification report.

STORYTELLING (CLUSTERING ANALYSIS)
I feel that the results of this experiment came out okay. I was not expecting to guess every genre perfectly and based on the information I had I am surprised by the results in a good way. Accuracy for picking the right genre of a song was a little under 30% and for having 10 different genres a song could be I felt this was fairly respectable. Some key things to note here is the model never predicted that a song would be part of the 'rock' genre accuractely once and the model was surprisingly accuracte at determining if a song was a 'classical' song.

IMPACT SECTION
Clustering is not commonly used for creating models for unsupervised learning. Music recommendation apps like Spotify use many different types of AI and ML instead of just one technique to find which song to recommend to a user. If recommendation systems can use another technique to help create more accurate recommendations for users than they should do so to give users the best experience. Completeing a project like this with a less common technique for recommendations could create discussion around new techniques to help create better recommendation systems in the future.

REFERENCES
https://www.kaggle.com/datasets/vicsuperman/prediction-of-music-genre/data
https://www.ibm.com/topics/clustering

CODE

import pandas as pd import numpy as np from sklearn.preprocessing import StandardScaler from sklearn.cluster import KMeans from sklearn.metrics import confusion_matrix, classification_report from sklearn.decomposition import PCA import matplotlib.pyplot as plt

In [87]:
```
df = pd.read_csv("music_genre.csv")
```

In [88]:
```
df.replace('?', np.nan, inplace=True)
```

```
df = df.dropna()

features = df[['acousticness', 'danceability', 'energy', 'instrumentalness', 'liven

scaler = StandardScaler()
scaled_features = scaler.fit_transform(features)

df_features_scaled = pd.DataFrame(scaled_features, columns=features.columns)
```

In [89]:
```
kmeans = KMeans(n_clusters=10, n_init=10, random_state=42)
df['predicted_genre'] = kmeans.fit_predict(scaled_features)
clusters = kmeans.fit_predict(scaled_features)
```

In [90]:
```
cluster_to_genre = df.groupby('predicted_genre')['music_genre'].agg(lambda x: x.val

df['predicted_genre'] = df['predicted_genre'].map(cluster_to_genre)

print(classification_report(df['music_genre'], df['predicted_genre'], zero_division
```

```
              precision    recall  f1-score   support

 Alternative       0.20      0.24      0.22      4495
       Anime       0.19      0.27      0.22      4497
       Blues       0.22      0.12      0.15      4470
   Classical       0.64      0.81      0.71      4500
     Country       0.17      0.28      0.21      4486
  Electronic       0.38      0.29      0.33      4466
     Hip-Hop       0.40      0.41      0.40      4520
        Jazz       0.23      0.26      0.24      4521
         Rap       0.25      0.27      0.26      4504
        Rock       0.00      0.00      0.00      4561

    accuracy                           0.29     45020
   macro avg       0.27      0.29      0.28     45020
weighted avg       0.27      0.29      0.28     45020
```

In [91]:
```
features = df[['acousticness', 'danceability', 'energy', 'instrumentalness', 'liven

scaler = StandardScaler()
scaled_features = scaler.fit_transform(features)
```

In [92]:
```
n_clusters = 10  # adjust based on your dataset and domain knowledge

kmeans = KMeans(n_clusters=n_clusters, n_init=10, random_state=42)
df['cluster'] = kmeans.fit_predict(scaled_features)
```

In [93]:
```
pca = PCA(n_components=2)
principal_components = pca.fit_transform(scaled_features)

pca_df = pd.DataFrame(data=principal_components, columns=['PC1', 'PC2'])
pca_df['cluster'] = df['cluster']
```

In [94]:
```
plt.figure(figsize=(10, 8))
colors = plt.cm.get_cmap('viridis', n_clusters)
```

```
scatter = plt.scatter(pca_df['PC1'], pca_df['PC2'], c=pca_df['cluster'], cmap=color
plt.title('Cluster Visualization using PCA')
plt.xlabel('Principal Component 1')
plt.ylabel('Principal Component 2')
plt.colorbar(scatter, ticks=range(n_clusters))
plt.show()
```

```
C:\Users\Holden\AppData\Local\Temp\ipykernel_2236\2663688093.py:2: MatplotlibDepreca
tionWarning: The get_cmap function was deprecated in Matplotlib 3.7 and will be remo
ved two minor releases later. Use ``matplotlib.colormaps[name]`` or ``matplotlib.col
ormaps.get_cmap(obj)`` instead.
  colors = plt.cm.get_cmap('viridis', n_clusters)
```