mlprojectcohorotsofsongs

February 19, 2025

1 CREATING COHORTS OF SONGS

2 Importing the libraries

[1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

[2]:
```
# EDA- Exploratory Data Analysis
```

[3]:
```
df=pd.read_csv("rolling_stones_spotify.csv")
```

[4]:
```
df
```

[5]:
```
df = df.drop(['Unnamed: 0'], axis=1)
```

[6]:
```
df.shape
```

[7]:
```
df.info()
```

[8]:
```
df.describe()
```

[9]:
```
df.isnull().sum()
```

[10]:
```
df.columns
```

[11]:
```
df.dtypes
```

[12]:
```
df
```

[13]:
```
df.corr()
```

[14]:
```
sns.heatmap(df.corr())
```

[15]:
```
df['track_number'].value_counts().plot(kind='barh', figsize=(10,15))
plt.xlabel('counts')
plt.ylabel('Track Number')
plt.title('Number of Track Number')
plt.show()
```

[16]:
```
sns.scatterplot(x=df['acousticness'],y=df['energy'])
plt.title('Acoustickness vs Enenrgy')
plt.show()
```

[17]:
```
sns.scatterplot(x=df['track_number'],y=df['energy'])
plt.title('Track Number vs Enenrgy')
plt.show()
```

[18]:
```
sns.scatterplot(x=df['energy'],y=df['danceability'])
plt.title('Energy vs Dance ability')
plt.show()
```

[19]:
```
plt.figure(figsize=(8,8),dpi=100)
plt.scatter(x=df['liveness'],y=df['loudness'])
plt.xlabel('Liveness')
plt.ylabel('Loudness')
plt.title('Liveness vs Loudenss')
plt.show()
```

[20]:
```
sns.scatterplot(x=df['liveness'],y=df['popularity'])
plt.title('Liveness vs Popularity')
plt.show()
```

[21]:
```
sns.scatterplot(x=df['energy'],y=df['liveness'],hue=df['popularity'])
plt.title('Energy vs Liveness with respect to Popularity')
plt.show()
```

[22]:
```
sns.histplot(df['duration_ms'])
plt.title('MS Duration Count')
plt.show()
```

[23]:
```
sns.scatterplot(x=df['tempo'],y=df['valence'],hue=df['loudness'])
plt.title('Tempo vs Valence with respect with loudness')
plt.show()
```

[24]:
```
sns.jointplot(x=df['track_number'],y=df['popularity'])
plt.show()
```

[25]:
```
sns.scatterplot(x=df['danceability'],y=df['energy'],hue=df['liveness'])
plt.title('Dance Ability vs Energy with respect to Liveness')
plt.show()
```

[26]:
```
sns.jointplot(x=df['danceability'],y=df['energy'],kind='hex')
plt.show()
```

[27]:
```
sns.jointplot(x=df['instrumentalness'],y=df['tempo'])
plt.show()
```

[28]:
```
plt.figure(figsize=(10,20))
sns.countplot(y=df['album'])
plt.title('Album')
plt.show()
```

[29]:
```
plt.figure(figsize=(10,20))
sns.scatterplot(y=df['album'],x=df['popularity'])
plt.title('Album vs Popularity')
plt.show()
```

[30]:
```
plt.figure(figsize=(10,20))
sns.scatterplot(y=df['album'],x=df['popularity'],hue=df['track_number'])
plt.title('Album vs Popularity with respect to Track Number')
plt.show()
```

[31]:
```
 plt.figure(figsize=(8,12))
sns.countplot(y=df['release_date'])
plt.title('Number of Songs Released in a date')
plt.show()
```

[32]:
```
sns.boxplot(df['popularity'])
plt.show()
```

[33]:
```
plt.figure(figsize=(15,5))
sns.boxplot(x=df['popularity'],y=df['track_number'])
plt.show()
```

[34]:
```
plt.figure(figsize=(15,5))
sns.barplot(x=df['track_number'],y=df['energy'])
plt.show()
```

[35]:
```
sns.boxplot(df['duration_ms'])
```

[36]:
```
cols = ['track_number','energy','popularity','liveness']
sns.pairplot(df,vars=cols)
```

[37]:
```
plt.show()
```

[38]:
```
cols = ['track_number','acousticness','danceability','liveness']
sns.pairplot(df,vars=cols,hue='popularity')
plt.show()
```

[39]:
```
 plt.figure(figsize=(8,15))
plt.subplots_adjust(hspace=0.5,wspace=0.5)
plt.subplot(5,2,1)
plt.hist(df['energy'])
plt.title('Energy')
plt.subplot(5,2,2)
plt.hist(df['track_number'])
plt.title('Track Number')
plt.subplot(5,2,4)
plt.hist(df['popularity'])
plt.title('Popularity')
plt.subplot(5,2,4)
plt.hist(df['liveness'])
plt.title('Liveness')
plt.subplot(5,2,5)
plt.hist(df['tempo'])
plt.title('Tempo')
plt.subplot(5,5,6)
plt.hist(df['duration_ms'])
plt.title('Duration MS')
plt.subplot(5,2,7)
plt.hist(df['instrumentalness'])
plt.title('Instrumentalness')
plt.subplot(5,2,8)
plt.hist(df['danceability'])
plt.title('Danceability')
plt.subplot(5,2,9)
plt.hist(df['loudness'])
plt.title('Loudness')
plt.subplot(5,2,10)
plt.hist(df['speechiness'])
plt.title('Speechiness')
plt.show()
```

cluster analysis

[40]:
```
df
```

[41]:
```
df.dtypes
```

[42]:
```
X = df.drop(['name','release_date','id','uri'],axis=1)
```

[43]:
```
X
```

[44]:
```
y = df['popularity']
```

[45]:
```
y
```

[46]:
```
from sklearn.preprocessing import LabelEncoder
```

[47]:
```
le = LabelEncoder()
```

[48]:
```
X['album'] = le.fit_transform(X['album'])
```

[49]:
```
X.head()
```

[50]:
```
from sklearn.preprocessing import MinMaxScaler
```

[51]:
```
ms = MinMaxScaler()
```

[52]:
```
cols = X.columns
```

[53]:
```
X = ms.fit_transform(X)
```

[54]:
```
X
```

[55]:
```
X = pd.DataFrame(X,columns=cols)
```

[56]:
```
X
```

[57]:
```
from sklearn.cluster import KMeans
```

[63]:
```
# Ensure X is already defined as your dataset
cs = []  # Initialize the list to store inertia values

# Loop through cluster numbers from 1 to 10
for i in range(1, 11):  # Ensuring the loop includes 10
    kmeans = KMeans(n_clusters=i, init='k-means++', max_iter=300, n_init=10,␣
 ↪random_state=0)
    kmeans.fit(X)  # Fit the model to the data
    cs.append(kmeans.inertia_)  # Append the inertia to the list

# Plot the results to visualize the elbow method (move outside loop)
plt.figure(figsize=(8, 5))
plt.plot(range(1, 11), cs, marker='o', linestyle='--', color='b')  # Correct␣
 ↪range
plt.title('Elbow Method')
plt.xlabel('Number of Clusters')
plt.ylabel('Inertia')
plt.show()
```

[59]:
```
# Assuming X is already defined as your dataset
# Initialize list to store inertia values
cs = []
```

```
# Loop to compute inertia for different cluster numbers
for i in range(1, 11):  # Ensuring the loop includes 10
    kmeans = KMeans(n_clusters=i, init='k-means++', max_iter=300, n_init=10,␣
 ↪random_state=0)
    kmeans.fit(X)  # Fit the model to the data
    cs.append(kmeans.inertia_)  # Append the inertia to the list

# Plot the results to visualize the elbow method
plt.figure(figsize=(8, 5))
plt.plot(range(1, 11), cs, marker='o', linestyle='--', color='b')  # Update␣
 ↪range to 1 to 10
plt.title('Elbow Method')
plt.xlabel('Number of Clusters')
plt.ylabel('Inertia')
plt.show()
```

[ ]: