In [1]:
```
import pandas as pd
```

In [2]:
```
data=pd.read_csv("C:\\Users\\Dell\\OneDrive\\Desktop\\excel books\\musicdata.csv")
```

In [3]:
```
print(data.head())
```

```
   Unnamed: 0                                 Track Name  \
0           0                             Bijlee Bijlee
1           1                               Expert Jatt
2           2  Kaun Nachdi (From "Sonu Ke Titu Ki Sweety")
3           3                               Na Na Na Na
4           4                               Patiala Peg

                    Artists                        Album Name  \
0              Harrdy Sandhu                      Bijlee Bijlee
1                      Nawab                        Expert Jatt
2  Guru Randhawa, Neeti Mohan  High Rated Gabru - Guru Randhawa
3                    J Star                        Na Na Na Na
4            Diljit Dosanjh  Do Gabru - Diljit Dosanjh & Akhil

                Album ID                Track ID  Popularity Release Date  \
0  3tG0IGB24sRhGFLs5F1Km8  1iZLpuGMr4tn1F5bZu32Kb          70   2021-10-30
1  2gibg5SCTep0wsIMefGzkd  7rr6n1NFIcQXCsi43P0YNl          65   2018-01-18
2  6EDbwGsQNQRLf73c7QwZ2f  3s7m0jmCXGcM8tmlvjCvAa          64   2019-03-02
3  4xBqgoiRSOMU1VlKuntVQW  5GjxbFTZAMhrVfVrNrrwrG          52         2015
4  1uxDllRe9CPhdr8rhz2QCZ  6TikcWOLRsPq66GBx2jk67          46   2018-07-10

   Duration (ms)  Explicit  ... Energy  Key  Loudness  Mode  Speechiness  \
0         168450     False  ...  0.670    1    -5.313     0       0.1430
1         199535     False  ...  0.948    6    -2.816     0       0.1990
2         183373     False  ...  0.830    4    -3.981     0       0.0455
3         209730     False  ...  0.863    3    -3.760     1       0.0413
4         188314     False  ...  0.811    5    -3.253     0       0.1840

   Acousticness  Instrumentalness  Liveness  Valence    Tempo
0        0.2690          0.000000    0.0733    0.643  100.004
1        0.2980          0.000000    0.0784    0.647  172.038
2        0.0357          0.000000    0.0419    0.753  127.999
3        0.3760          0.000014    0.0916    0.807   95.000
4        0.0259          0.000000    0.3110    0.835  175.910

[5 rows x 22 columns]
```

In [4]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100 entries, 0 to 99
Data columns (total 22 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   Unnamed: 0        100 non-null    int64  
 1   Track Name        94 non-null     object 
 2   Artists           94 non-null     object 
 3   Album Name        94 non-null     object 
 4   Album ID          100 non-null    object 
 5   Track ID          100 non-null    object 
 6   Popularity        100 non-null    int64  
 7   Release Date      100 non-null    object 
 8   Duration (ms)     100 non-null    int64  
 9   Explicit          100 non-null    bool   
 10  External URLs     100 non-null    object 
 11  Danceability      100 non-null    float64
 12  Energy            100 non-null    float64
 13  Key               100 non-null    int64  
 14  Loudness          100 non-null    float64
 15  Mode              100 non-null    int64  
 16  Speechiness       100 non-null    float64
 17  Acousticness      100 non-null    float64
 18  Instrumentalness  100 non-null    float64
 19  Liveness          100 non-null    float64
 20  Valence           100 non-null    float64
 21  Tempo             100 non-null    float64
dtypes: bool(1), float64(9), int64(5), object(7)
memory usage: 16.6+ KB
```

In [5]:
```
data.index
```

Out[5]:
```
RangeIndex(start=0, stop=100, step=1)
```

In [6]:
```
data.isnull().sum()
```

Out[6]:
```
Unnamed: 0          0
Track Name          6
Artists             6
Album Name          6
Album ID            0
Track ID            0
Popularity          0
Release Date        0
Duration (ms)       0
Explicit            0
External URLs       0
Danceability        0
Energy              0
Key                 0
Loudness            0
Mode                0
Speechiness         0
Acousticness        0
Instrumentalness    0
Liveness            0
Valence             0
Tempo               0
dtype: int64
```

In [7]:
```
data.describe()
```

Out[7]:

| | Unnamed: 0 | Popularity | Duration (ms) | Danceability | Energy | Key | Loudness | |
|---|---|---|---|---|---|---|---|---|
| **count** | 100.000000 | 100.000000 | 100.000000 | 100.000000 | 100.00000 | 100.00000 | 100.000000 | 100. |
| **mean** | 49.500000 | 50.950000 | 210543.180000 | 0.767210 | 0.79763 | 4.54000 | -4.399930 | 0. |
| **std** | 29.011492 | 16.496326 | 37961.050214 | 0.085302 | 0.11572 | 3.64434 | 1.612703 | 0. |
| **min** | 0.000000 | 0.000000 | 141862.000000 | 0.501000 | 0.47700 | 0.00000 | -8.272000 | 0. |
| **25%** | 24.750000 | 46.000000 | 186098.500000 | 0.714750 | 0.71125 | 1.00000 | -5.465250 | 0. |
| **50%** | 49.500000 | 56.500000 | 205076.000000 | 0.772000 | 0.81700 | 4.00000 | -4.252500 | 0. |
| **75%** | 74.250000 | 62.000000 | 226079.000000 | 0.826500 | 0.88125 | 7.25000 | -3.163250 | 1. |
| **max** | 99.000000 | 72.000000 | 367818.000000 | 0.959000 | 0.98800 | 11.00000 | -0.223000 | 1. |

In [8]:
```
cleaned_data=data.drop(columns=['Unnamed: 0'])
cleaned_data
```

Out[8]:

| | Track Name | Artists | Album Name | Album ID | Track ID | Pc |
|---|---|---|---|---|---|---|
| **0** | Bijlee Bijlee | Harrdy Sandhu | Bijlee Bijlee | 3tG0lGB24sRhGFLs5F1Km8 | 1iZLpuGMr4tn1F5bZu32Kb | |
| **1** | Expert Jatt | Nawab | Expert Jatt | 2gibg5SCTep0wsIMefGzkd | 7rr6n1NFIcQXCsi43P0YNl | |
| **2** | Kaun Nachdi (From "Sonu Ke Titu Ki Sweety") | Guru Randhawa, Neeti Mohan | High Rated Gabru - Guru Randhawa | 6EDbwGsQNQRLf73c7QwZ2f | 3s7m0jmCXGcM8tmlvjCvAa | |
| **3** | Na Na Na Na | J Star | Na Na Na Na | 4xBqgoiRSOMU1VlKuntVQW | 5GjxbFTZAMhrVfVrNrrwrG | |
| **4** | Patiala Peg | Diljit Dosanjh | Do Gabru - Diljit Dosanjh & Akhil | 1uxDllRe9CPhdr8rhz2QCZ | 6TikcWOLRsPq66GBx2jk67 | |
| **...** | ... | ... | ... | ... | ... | |
| **95** | NaN | NaN | NaN | 2jw92hf4mnlSbYywvU3Anj | 3OZr3vo7SmYpn5XqeQEAOM | |
| **96** | Move Your Lakk | Diljit Dosanjh, Badshah, Sonakshi Sinha | Move Your Lakk | 0V06TMGQQQkvKxNmFlKyEj | 3aYMKdSitJeHUCZO8Wt6fw | |
| **97** | Patola (From "Blackmail") | Guru Randhawa, Preet Hundal | Patola (From "Blackmail") | 2XAAIDEpPb57NsKgAHLGVQ | 17LZzRCY0iFWlDDuAG7BIM | |
| **98** | Ban Ja Rani (From "Tumhari Sulu") | Guru Randhawa | High Rated Gabru - Guru Randhawa | 6EDbwGsQNQRLf73c7QwZ2f | 7cQtGVoPCK9DlspeYjdHOA | |
| **99** | Hauli Hauli (From "De De Pyaar De") | Garry Sandhu, Neha Kakkar, Mellow D | Dance Syndrome | 6e1XB070vlPaxGDAsi8AF6 | 4XyKoSEWrkQjI4AekJYWNx | |

100 rows × 21 columns

In [9]:
```
columns_with_na = ['Track Name','Artists','Album Name']
cleaned_data[columns_with_na]=cleaned_data[columns_with_na].fillna('unkown')
```

In [10]:
```
cleaned_data
```

Out[10]:

| | Track Name | Artists | Album Name | Album ID | Track ID | Pc |
|---|---|---|---|---|---|---|
| **0** | Bijlee Bijlee | Harrdy Sandhu | Bijlee Bijlee | 3tG0lGB24sRhGFLs5F1Km8 | 1iZLpuGMr4tn1F5bZu32Kb | |
| **1** | Expert Jatt | Nawab | Expert Jatt | 2gibg5SCTep0wsIMefGzkd | 7rr6n1NFIcQXCsi43P0YNl | |
| **2** | Kaun Nachdi (From "Sonu Ke Titu Ki Sweety") | Guru Randhawa, Neeti Mohan | High Rated Gabru - Guru Randhawa | 6EDbwGsQNQRLf73c7QwZ2f | 3s7m0jmCXGcM8tmlvjCvAa | |
| **3** | Na Na Na Na | J Star | Na Na Na Na | 4xBqgoiRSOMU1VlKuntVQW | 5GjxbFTZAMhrVfVrNrrwrG | |
| **4** | Patiala Peg | Diljit Dosanjh | Do Gabru - Diljit Dosanjh & Akhil | 1uxDllRe9CPhdr8rhz2QCZ | 6TikcWOLRsPq66GBx2jk67 | |
| **...** | ... | ... | ... | ... | ... | |
| **95** | unkown | unkown | unkown | 2jw92hf4mnISbYywvU3Anj | 3OZr3vo7SmYpn5XqeQEAOM | |
| **96** | Move Your Lakk | Diljit Dosanjh, Badshah, Sonakshi Sinha | Move Your Lakk | 0V06TMGQQQkvKxNmFlKyEj | 3aYMKdSitJeHUCZO8Wt6fw | |
| **97** | Patola (From "Blackmail") | Guru Randhawa, Preet Hundal | Patola (From "Blackmail") | 2XAAIDEpPb57NsKgAHLGVQ | 17LZzRCY0iFWlDDuAG7BIM | |
| **98** | Ban Ja Rani (From "Tumhari Sulu") | Guru Randhawa | High Rated Gabru - Guru Randhawa | 6EDbwGsQNQRLf73c7QwZ2f | 7cQtGVoPCK9DlspeYjdHOA | |
| **99** | Hauli Hauli (From "De De Pyaar De") | Garry Sandhu, Neha Kakkar, Mellow D | Dance Syndrome | 6e1XB070vlPaxGDAsi8AF6 | 4XyKoSEWrkQjI4AekJYWNx | |

100 rows × 21 columns

In [11]:
```
cleaned_data.isnull().sum()
```

Out[11]:
```
Track Name          0
Artists             0
Album Name          0
Album ID            0
Track ID            0
Popularity          0
Release Date        0
Duration (ms)       0
Explicit            0
External URLs       0
Danceability        0
Energy              0
Key                 0
Loudness            0
Mode                0
Speechiness         0
Acousticness        0
Instrumentalness    0
Liveness            0
Valence             0
Tempo               0
dtype: int64
```

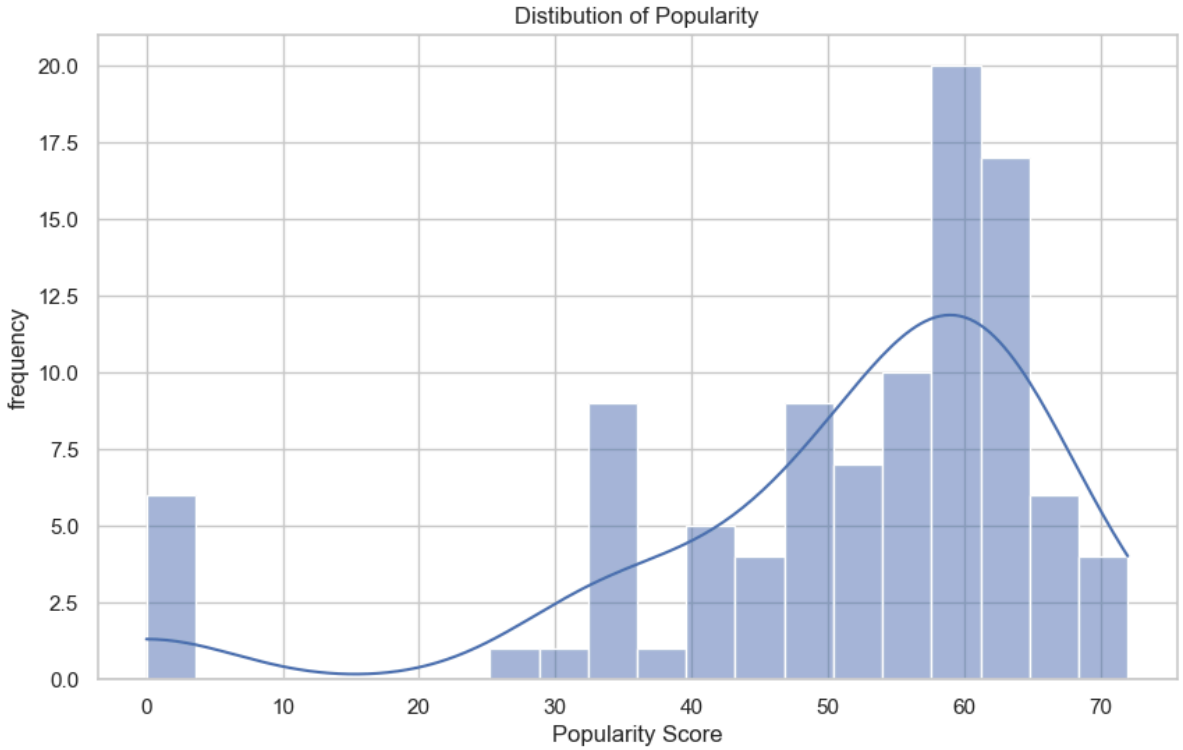
In [34]:
```
cleaned_data.describe()
```

Out[34]:

| | Popularity | Duration (ms) | Explicit | Danceability | Energy | Key | Loudness | Mo |
|---|---|---|---|---|---|---|---|---|
| **count** | 100.000000 | 100.000000 | 100.00 | 100.000000 | 100.00000 | 100.00000 | 100.000000 | 100.000 |
| **mean** | 50.950000 | 210543.180000 | 0.01 | 0.767210 | 0.79763 | 4.54000 | -4.399930 | 0.430 |
| **std** | 16.496326 | 37961.050214 | 0.10 | 0.085302 | 0.11572 | 3.64434 | 1.612703 | 0.497 |
| **min** | 0.000000 | 141862.000000 | 0.00 | 0.501000 | 0.47700 | 0.00000 | -8.272000 | 0.000 |
| **25%** | 46.000000 | 186098.500000 | 0.00 | 0.714750 | 0.71125 | 1.00000 | -5.465250 | 0.000 |
| **50%** | 56.500000 | 205076.000000 | 0.00 | 0.772000 | 0.81700 | 4.00000 | -4.252500 | 0.000 |
| **75%** | 62.000000 | 226079.000000 | 0.00 | 0.826500 | 0.88125 | 7.25000 | -3.163250 | 1.000 |
| **max** | 72.000000 | 367818.000000 | 1.00 | 0.959000 | 0.98800 | 11.00000 | -0.223000 | 1.000 |

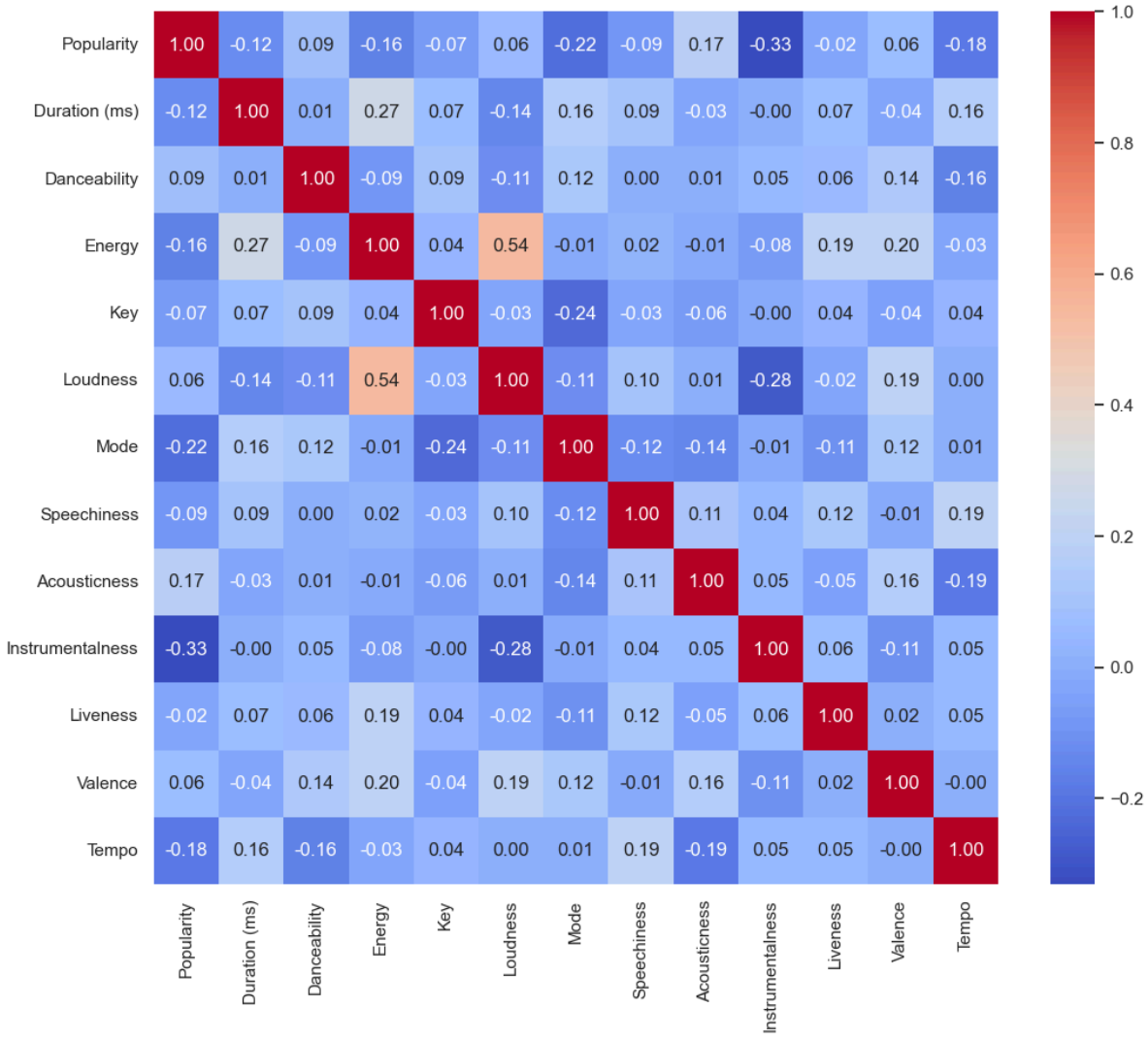
In [12]:
```
import seaborn as sns
import matplotlib.pyplot as plt
```

In [13]:
```
sns.set(style='whitegrid')
plt.figure(figsize=(10,6))
sns.histplot(cleaned_data['Popularity'],bins=20,kde=True)
plt.title('Distibution of Popularity')
plt.xlabel('Popularity Score')
plt.ylabel('frequency')
plt.show()
```

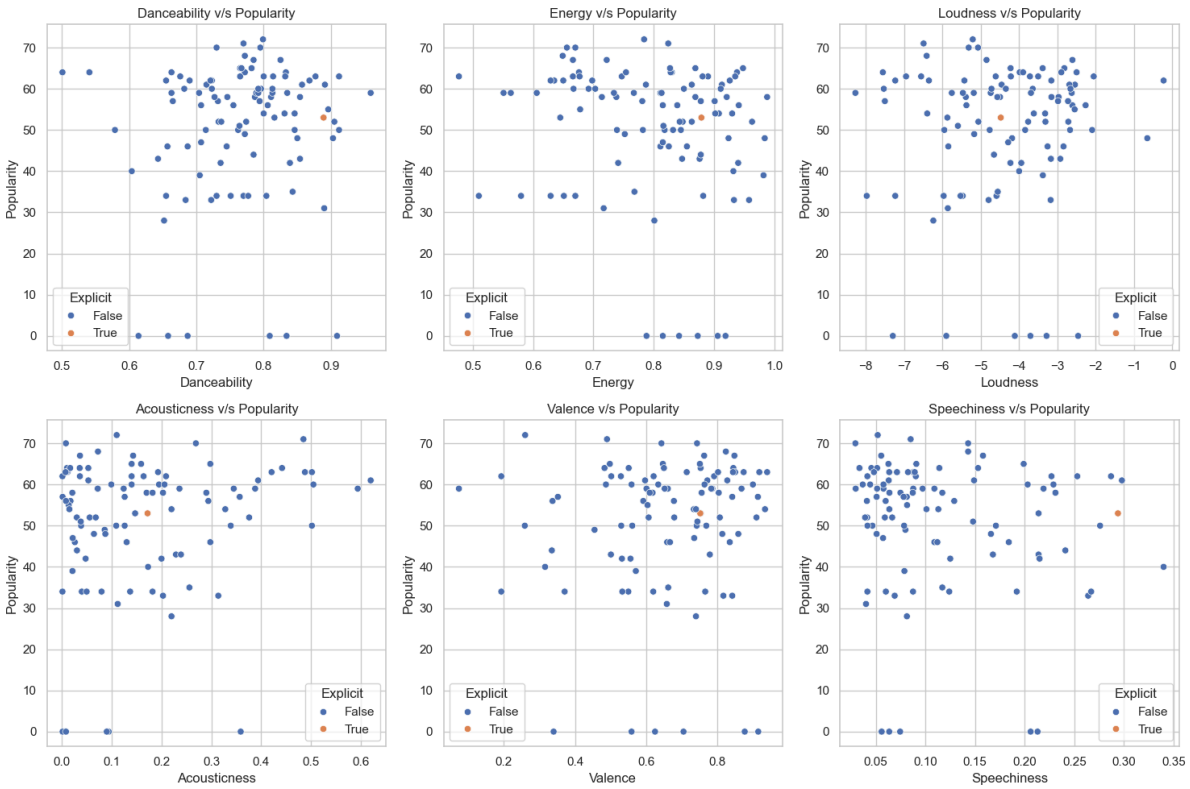



In [14]:
```
plt.figure(figsize=(12,10))
corr_matrix=cleaned_data.select_dtypes(include=['float64', 'int64']).corr()
sns.heatmap(corr_matrix,annot=True,cmap='coolwarm',fmt=".2f")
```

Out[14]: <AxesSubplot:>



In [15]:
```
features=['Danceability', 'Energy', 'Loudness', 'Acousticness', 'Valence','Speechir
plt.figure(figsize=(15,10))
for i , feature in enumerate(features,1):
    plt.subplot(2,3,i)
    sns.scatterplot(x=cleaned_data[feature],y=cleaned_data['Popularity'],hue=cleane
    plt.title(f'{feature} v/s Popularity')
    plt.xlabel(feature)
    plt.ylabel('Popularity')
plt.tight_layout()
plt.show()
```




In [16]:
```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression  #y=mx+c
from sklearn.metrics import mean_squared_error, r2_score
from sklearn.preprocessing import StandardScaler #z=x-u/sigma
```

In [17]:
```
cleaned_data.dtypes
```

Out[17]:
```
Track Name           object
Artists              object
Album Name           object
Album ID             object
Track ID             object
Popularity            int64
Release Date         object
Duration (ms)         int64
Explicit               bool
External URLs        object
Danceability        float64
Energy              float64
Key                   int64
Loudness            float64
Mode                  int64
Speechiness         float64
Acousticness        float64
Instrumentalness    float64
Liveness            float64
Valence             float64
Tempo               float64
dtype: object
```

In [18]:
```
cleaned_data['Explicit']=cleaned_data['Explicit'].astype('int')
```

In [19]:
```
cleaned_data['Explicit']
```

Out[19]:
```
0     0
1     0
2     0
3     0
4     0
     ..
95    0
96    0
97    0
98    0
99    0
Name: Explicit, Length: 100, dtype: int32
```

In [20]:
```
features=['Danceability', 'Valence']
x=cleaned_data[features]
y=cleaned_data['Popularity']

scaler=StandardScaler()
x_scaled=scaler.fit_transform(x)
```

In [21]:
```
X_train,X_test,Y_train,Y_test=train_test_split(x_scaled,y,test_size=0.2,random_stat
```

In [22]:
```
model=LinearRegression()
```

In [23]:
```
model.fit(X_train,Y_train)
```

Out[23]:
```
LinearRegression()
```

In [24]:
```
y_pred=model.predict(X_test)
```

In [25]:
```
y_pred
```

Out[25]:
```
array([49.35952842, 46.8821292 , 48.73216991, 51.90165825, 52.37423224,
       50.3400056 , 47.81632694, 49.97176225, 46.27102562, 49.66480711,
       53.49744559, 49.71086162, 50.79208592, 50.74596714, 53.31894992,
       50.98846133, 49.00102128, 43.19213957, 53.51038667, 51.32723902])
```

In [26]:
```
Y_test
```

Out[26]:
```
42    43
94    50
37    50
6     59
79    48
32    61
24    59
0     70
33    72
93    60
68    54
60    47
57    35
2     64
54    63
72    54
78    58
16    59
25    59
27    53
Name: Popularity, dtype: int64
```

In [27]:
```
mse= mean_squared_error(Y_test,y_pred)
```

In [28]:
```
mse
```

Out[28]:
```
121.45929939785506
```

In [29]:
```
accuracy =r2_score(Y_test,y_pred)
```

In [ ]:
```

```

In [31]:
```
coefficients = pd.Series(model.coef_, index=features)
coefficients
```

Out[31]:
```
Danceability    1.121259
Valence         1.793277
dtype: float64
```

In [33]:
```
#y=m1x+m2x+c
```

In [ ]:
```

```