BillBoard Hits Prediction-Project_Workflow_Details_EDA

Libraries Import

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

from IPython.display import IFrame

import warnings
warnings.filterwarnings('ignore')

from sklearn.preprocessing import StandardScaler


pd.options.plotting.backend = 'matplotlib'

#pd.options.plotting.backend = "plotly"


import scipy.stats as stats

from statsmodels.stats import weightstats as stests
```

Initial Data Analysis and Visualization

Bill Board Data

In [2]:
```python
df_bb = pd.read_excel('C:/Users/Anjum/Downloads/Technocolabs Materials/Predicting-Billboard-Hits-Using-Spotify-
```

In [3]:
```python
df_bb.head(2)
```

Out[3]:

	Track	Artist	SpotifyID	danceability	energy	key	mode	speechiness	acousticness	instrumentalness	liveness	valence
0	Lucid Dreams	Juice WRLD	285pBltuF7vW8TeWk8hdRR	0.511	0.566	6	0	0.20	0.349	0.0	0.340	0.218
1	Better Now	Post Malone	7dt6x5M1jzdTEt8oCbisTK	0.680	0.578	10	1	0.04	0.331	0.0	0.135	0.341

In [4]:
```python
df_bb.shape
```

Out[4]: (9329, 15)

In [5]:
```python
df_bb.shape
```

Out[5]: (9329, 15)

Dropping duplicate data based on SpotifyID

In [6]:
```python
df_bb.drop_duplicates(subset='SpotifyID',keep=False,inplace=True)
```

In [7]:
```python
df_bb.shape
```

Out[7]: (9258, 15)

MSD Data

In [8]:
```python
df_msd = pd.read_excel('C:/Users/Anjum/Downloads/Technocolabs Materials/Predicting-Billboard-Hits-Using-Spotify
```

In [9]:
```python
df_msd.head(2)
```

Out[9]:

	0	1	2	3	4	5	danceability	energy	key	loudness	mode
0	Sonora Santanera	Las Numero 1 De La Sonora Santanera	Amor De Cabaret	2016	TRAAADZ128F9348C2E	5JeyXbf7CrVQ8DqwFikN0P	0.699	0.519	8	-6.422	1
1	JennyAnyKind	I Need You	Young Boy Blues	2000	TRAAAVO128F93133D4	5uN6oY2NGLC6GKoywHKkZA	0.458	0.570	5	-9.159	0

In [10]:
```python
df_msd.shape
```

Out[10]: (5603, 18)

Renaming columns of MSD Dataframe

In [11]:
```python
df_msd.rename(columns= {0:'Artist',1:'Album',2:'Track',3:'Year',4:'TrackID',5:'SpotifyID'},inplace=True)
```

In [12]:
```python
df_msd.head()
```

Out[12]:

	Artist	Album	Track	Year	TrackID	SpotifyID	danceability	energy	key	loudness	mode
0	Sonora Santanera	Las Numero 1 De La Sonora Santanera	Amor De Cabaret	2016	TRAAADZ128F9348C2E	5JeyXbf7CrVQ8DqwFikN0P	0.699	0.519	8	-6.422	1
1	JennyAnyKind	I Need You	Young Boy Blues	2000	TRAAAVO128F93133D4	5uN6oY2NGLC6GKoywHKkZA	0.458	0.570	5	-9.159	0
2	Casual	Fear Itself	I Didn't Mean To	1994	TRAAAAW128F429D538	01TR6aAKrA2cI3Z0gnCOsu	0.751	0.549	6	-10.508	0
3	Jeff And Sheri Easter	Ordinary Day	The Moon And I (Ordinary Day Album Version)	2006	TRAAAMO128F1481E7F	7ERvcsBBYzSHqQQqIl9xAN	0.456	0.472	5	-8.328	1
4	Tweeterfriendly Music	Gin & Phonic	Drop of Rain	2003	TRAAAPK128E0786D96	6dfppQDZQ4x3fCs6N89kV3	0.498	0.300	7	-10.716	1

Dropping duplicate data based on SpotifyID

In [13]:
```python
df_msd.drop(columns=['Album','Year','TrackID',],axis=1,inplace=True)
```

In [14]:
```python
df_msd.shape
```

Out[14]: (5603, 15)

In [15]:
```python
df_msd.drop_duplicates(subset='SpotifyID',keep=False,inplace=True)
```

In [16]:
```python
df_msd.shape
```

Out[16]: (5522, 15)

In [17]:
```python
df_bb.shape ,df_msd.shape
```

Out[17]: ((9258, 15), (5522, 15))

In [18]:
```python
df_bb.head(2)
```

Out[18]:

	Track	Artist	SpotifyID	danceability	energy	key	mode	speechiness	acousticness	instrumentalness	liveness	valence
0	Lucid Dreams	Juice WRLD	285pBltuF7vW8TeWk8hdRR	0.511	0.566	6	0	0.20	0.349	0.0	0.340	0.218
1	Better Now	Post Malone	7dt6x5M1jzdTEt8oCbisTK	0.680	0.578	10	1	0.04	0.331	0.0	0.135	0.341

In [19]:
```python
df_msd.head(2)
```

Out[19]:

	Artist	Track	SpotifyID	danceability	energy	key	loudness	mode	speechiness	acousticness	instrumentalness
0	Sonora Santanera	Amor De Cabaret	5JeyXbf7CrVQ8DqwFikN0P	0.699	0.519	8	-6.422	1	0.0290	0.689000	0.000471
1	JennyAnyKind	Young Boy Blues	5uN6oY2NGLC6GKoywHKkZA	0.458	0.570	5	-9.159	0	0.0495	0.000055	0.000768

List of unique spotifyID that are present in bill board data and are hit.

In [20]:
```python
spotify_id_hit = list(df_bb['SpotifyID'].unique())

spotify_id_hit[0:5]
```

Out[20]:
```
['285pBltuF7vW8TeWk8hdRR',
 '7dt6x5M1jzdTEt8oCbisTK',
 '78QR3Wp35dqAhFEc2qAGjE',
 '2xLMifQCjDGFmkHkpNLD9h',
 '2iUXsYOEPhVqEBwsqP70rE']
```

Total number of hit tracks

Below are the ids present in only billboard and refers to hit songs

In [21]:
```python
len(spotify_id_hit)
```

Out[21]: 9258

In [22]:
```python
df_hit = df_bb[df_bb['SpotifyID'].isin(df_bb['SpotifyID'])]
df_hit.shape
```

Out[22]: (9258, 15)

Below are the ids not present in billboard and refers to non-hit songs

In [23]:
```python
df_non_hit = df_msd[~df_msd['SpotifyID'].isin(df_bb['SpotifyID'])]
df_non_hit.shape
```

Out[23]: (5424, 15)

In [24]:
```python
df_hit.head(2)
```

Out[24]:

	Track	Artist	SpotifyID	danceability	energy	key	mode	speechiness	acousticness	instrumentalness	liveness	valence
0	Lucid Dreams	Juice WRLD	285pBltuF7vW8TeWk8hdRR	0.511	0.566	6	0	0.20	0.349	0.0	0.340	0.218
1	Better Now	Post Malone	7dt6x5M1jzdTEt8oCbisTK	0.680	0.578	10	1	0.04	0.331	0.0	0.135	0.341

In [25]:
```python
df_non_hit.head(2)
```

Out[25]:

	Artist	Track	SpotifyID	danceability	energy	key	loudness	mode	speechiness	acousticness	instrumentalness
0	Sonora Santanera	Amor De Cabaret	5JeyXbf7CrVQ8DqwFikN0P	0.699	0.519	8	-6.422	1	0.0290	0.689000	0.000471
1	JennyAnyKind	Young Boy Blues	5uN6oY2NGLC6GKoywHKkZA	0.458	0.570	5	-9.159	0	0.0495	0.000055	0.000768

We have to set the sequence of the dataframe same for both.

In [26]:
```python
df_hit.columns
```

Out[26]:
```
Index(['Track', 'Artist', 'SpotifyID', 'danceability', 'energy', 'key', 'mode',
       'speechiness', 'acousticness', 'instrumentalness', 'liveness',
       'valence', 'tempo', 'duration_ms', 'loudness'],
      dtype='object')
```

Rearrange columns of df_non_hit as per df_hit dataframe

In [27]:
```python
df_non_hit = df_non_hit[['Track', 'Artist', 'SpotifyID', 'danceability', 'energy', 'key', 'mode',
       'speechiness', 'acousticness', 'instrumentalness', 'liveness',
       'valence', 'tempo', 'duration_ms', 'loudness']]
```

The sequence of the columns are now same for both.

In [28]:
```python
df_hit.head(2)
```

Out[28]:

	Track	Artist	SpotifyID	danceability	energy	key	mode	speechiness	acousticness	instrumentalness	liveness	valence
0	Lucid Dreams	Juice WRLD	285pBltuF7vW8TeWk8hdRR	0.511	0.566	6	0	0.20	0.349	0.0	0.340	0.218
1	Better Now	Post Malone	7dt6x5M1jzdTEt8oCbisTK	0.680	0.578	10	1	0.04	0.331	0.0	0.135	0.341

In [29]:
```python
df_non_hit.head(2)
```

Out[29]:

	Track	Artist	SpotifyID	danceability	energy	key	mode	speechiness	acousticness	instrumentalness	liveness
0	Amor De Cabaret	Sonora Santanera	5JeyXbf7CrVQ8DqwFikN0P	0.699	0.519	8	1	0.0290	0.689000	0.000471	0.9350
1	Young Boy Blues	JennyAnyKind	5uN6oY2NGLC6GKoywHKkZA	0.458	0.570	5	0	0.0495	0.000055	0.000768	0.0988

Appending column 'is_hit' indicator as 0 for df_non_hit and 1 for df_hit

In [30]:
```python
df_hit['is_hit']= 1
df_non_hit['is_hit'] = 0
```

New column is_hit added successfully for both the dataset and ready to be merged together.

In [31]:
```python
df_hit.head(2)
```

Out[31]:

	Track	Artist	SpotifyID	danceability	energy	key	mode	speechiness	acousticness	instrumentalness	liveness	valence
0	Lucid Dreams	Juice WRLD	285pBltuF7vW8TeWk8hdRR	0.511	0.566	6	0	0.20	0.349	0.0	0.340	0.218
1	Better Now	Post Malone	7dt6x5M1jzdTEt8oCbisTK	0.680	0.578	10	1	0.04	0.331	0.0	0.135	0.341

In [32]:
```python
df_non_hit.head(2)
```

Out[32]:

	Track	Artist	SpotifyID	danceability	energy	key	mode	speechiness	acousticness	instrumentalness	liveness
0	Amor De Cabaret	Sonora Santanera	5JeyXbf7CrVQ8DqwFikN0P	0.699	0.519	8	1	0.0290	0.689000	0.000471	0.9350
1	Young Boy Blues	JennyAnyKind	5uN6oY2NGLC6GKoywHKkZA	0.458	0.570	5	0	0.0495	0.000055	0.000768	0.0988

In [ ]:

In [33]:
```python
df_merge = pd.concat([df_hit,df_non_hit])
```

Dropping duplicate records based on spotifyID

In [34]:
```python
df_merge.drop_duplicates(subset='SpotifyID',keep=False,inplace=True)
```

In [35]:
```python
df_merge.shape
```

Out[35]: (14682, 16)

Shuffling the dataframe and exporting the data for pre-processing.

In [36]:
```python
df_merge = df_merge.sample(frac=1).reset_index(drop=True)
```

So, this is the data we can use for further pre-processing.

In [37]:
```python
df_merge.head()
```

Out[37]:

	Track	Artist	SpotifyID	danceability	energy	key	mode	speechiness	acousticness	instrumentalness	liveness
0	Firedance	Mystic Rhythms Band	2yScBrg0zVe6WnafJLfCIL	0.637	0.876	8	1	0.0415	0.1060	0.313000	0.136
1	Stop The World	Extreme	4Vghbo90YLD9BudOpMKW0K	0.452	0.724	8	1	0.0435	0.0038	0.001120	0.687
2	Nights	Frank Ocean	7eqoqGkKwgOaWNNHx90uEZ	0.466	0.548	5	0	0.1180	0.4200	0.000001	0.113
3	Suds In The Bucket	Sara Evans	6NhpIdjYoufuNNIBsgOztc	0.511	0.903	10	1	0.0397	0.0520	0.000529	0.304
4	Shrivasta	William Coulter	0LkvMWl5u8zu1vqsGCH6jG	0.358	0.285	9	0	0.0381	0.9410	0.893000	0.121

EDA and Visualization

In [38]:
```python
df_merge.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 14682 entries, 0 to 14681
Data columns (total 16 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   Track             14681 non-null  object
 1   Artist            14680 non-null  object
 2   SpotifyID         14682 non-null  object
 3   danceability      14682 non-null  float64
 4   energy            14682 non-null  float64
 5   key               14682 non-null  int64
 6   mode              14682 non-null  int64
 7   speechiness       14682 non-null  float64
 8   acousticness      14682 non-null  float64
 9   instrumentalness  14682 non-null  float64
 10  liveness          14682 non-null  float64
 11  valence           14682 non-null  float64
 12  tempo             14682 non-null  float64
 13  duration_ms       14682 non-null  int64
 14  loudness          14682 non-null  float64
 15  is_hit            14682 non-null  int64
dtypes: float64(9), int64(4), object(3)
memory usage: 1.8+ MB
```

Setting spotifyID as index as it is unique for all records.

In [39]:
```python
df_merge.set_index('SpotifyID',inplace=True)
```

In [40]:
```python
df_merge.head()
```

Out[40]:

SpotifyID	Track	Artist	danceability	energy	key	mode	speechiness	acousticness	instrumentalness	liveness
2yScBrg0zVe6WnafJLfCIL	Firedance	Mystic Rhythms Band	0.637	0.876	8	1	0.0415	0.1060	0.313000	0.136
4Vghbo90YLD9BudOpMKW0K	Stop The World	Extreme	0.452	0.724	8	1	0.0435	0.0038	0.001120	0.687
7eqoqGkKwgOaWNNHx90uEZ	Nights	Frank Ocean	0.466	0.548	5	0	0.1180	0.4200	0.000001	0.113
6NhpIdjYoufuNNIBsgOztc	Suds In The Bucket	Sara Evans	0.511	0.903	10	1	0.0397	0.0520	0.000529	0.304
0LkvMWl5u8zu1vqsGCH6jG	Shrivasta	William Coulter	0.358	0.285	9	0	0.0381	0.9410	0.893000	0.121

Missing values Treatment

In [41]:
```python
df_merge.isna().sum()
```

Out[41]:
```
Track               1
Artist              2
danceability        0
energy              0
key                 0
mode                0
speechiness         0
acousticness        0
instrumentalness    0
liveness            0
valence             0
tempo               0
duration_ms         0
loudness            0
is_hit              0
dtype: int64
```

Dropping the records where Track and Artist are null.

In [42]:
```python
df_merge = df_merge.dropna(subset=['Track','Artist'],axis=0)
```

Now, No missing values left to be treated.

In [43]:
```python
df_merge.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 14680 entries, 2yScBrg0zVe6WnafJLfCIL to 17VqQQo5RkdA7iukLr0okv
Data columns (total 15 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   Track             14680 non-null  object
 1   Artist            14680 non-null  object
 2   danceability      14680 non-null  float64
 3   energy            14680 non-null  float64
 4   key               14680 non-null  int64
 5   mode              14680 non-null  int64
 6   speechiness       14680 non-null  float64
 7   acousticness      14680 non-null  float64
 8   instrumentalness  14680 non-null  float64
 9   liveness          14680 non-null  float64
 10  valence           14680 non-null  float64
 11  tempo             14680 non-null  float64
 12  duration_ms       14680 non-null  int64
 13  loudness          14680 non-null  float64
 14  is_hit            14680 non-null  int64
dtypes: float64(9), int64(4), object(2)
memory usage: 1.8+ MB
```

In [44]:
```python
df_numeric = df_merge.select_dtypes(include=np.number)
```

In [45]:
```python
df_numeric.columns
```

Out[45]:
```
Index(['danceability', 'energy', 'key', 'mode', 'speechiness', 'acousticness',
       'instrumentalness', 'liveness', 'valence', 'tempo', 'duration_ms',
       'loudness', 'is_hit'],
      dtype='object')
```

In [46]:
```python
ss = StandardScaler()
```

In [47]:
```python
df_numeric_ss = ss.fit_transform(df_numeric)
```

In [48]:
```python
df_numeric_ss = pd.DataFrame(df_numeric_ss,columns=df_numeric.columns)
```

Standard Scaled data

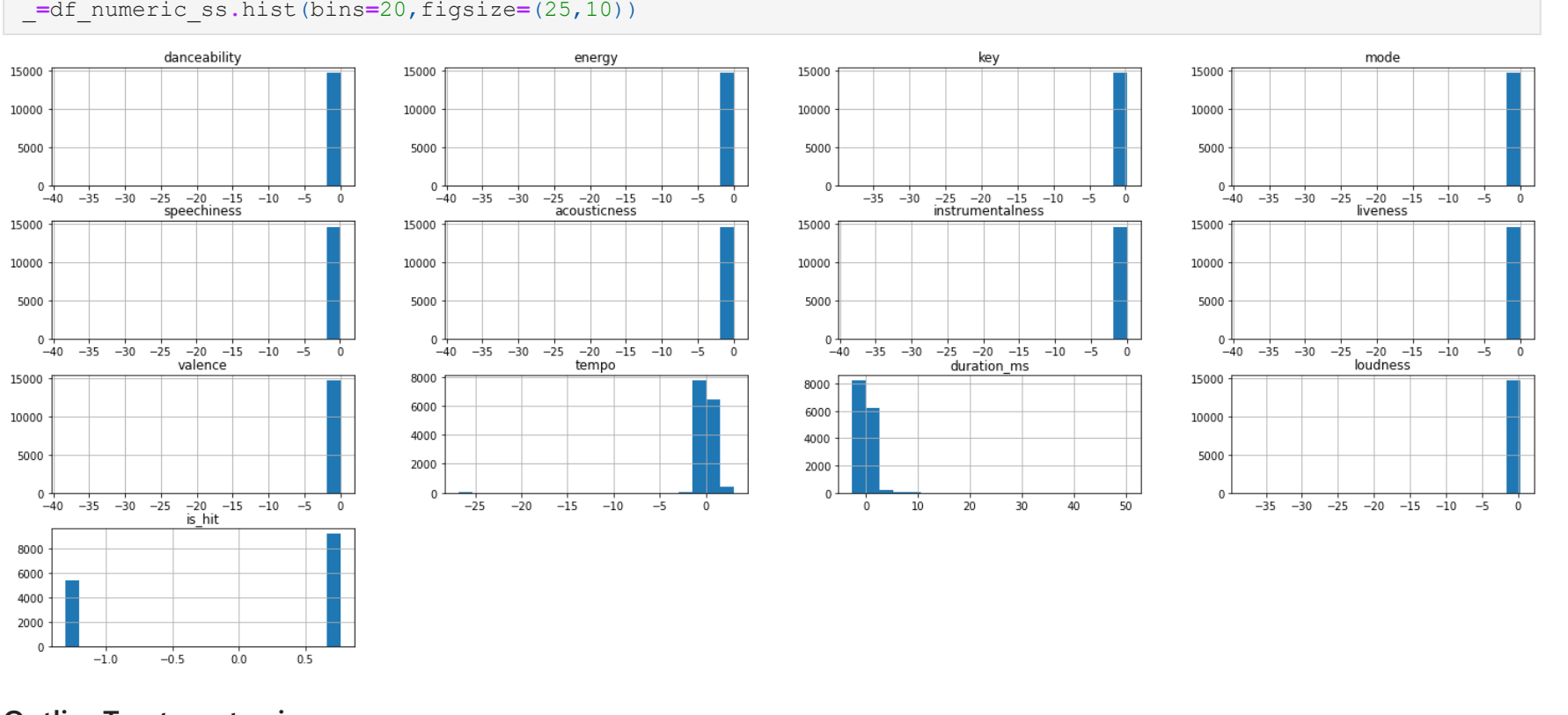
In [49]:
```python
df_numeric_ss.head()
```

Out[49]:

	danceability	energy	key	mode	speechiness	acousticness	instrumentalness	liveness	valence	tempo	duration_ms
0	0.027493	0.034721	0.128226	0.038577	0.023965	0.021275	0.034854	0.023722	0.031735	0.500962	-0.376258
1	0.020399	0.028894	0.128226	0.038577	0.024042	0.017355	0.022890	0.044857	0.018429	1.310263	1.294579
2	0.020936	0.022146	0.014781	0.000243	0.026900	0.033318	0.022847	0.022840	0.022187	-0.719670	0.735136
3	0.022662	0.035757	0.203857	0.038577	0.023896	0.019203	0.022867	0.030166	0.040400	1.008523	-0.144910
4	0.016795	0.012063	0.166041	0.000243	0.023835	0.053301	0.057104	0.023147	0.018276	0.694093	-0.020203

Plotting histogram of scaled numeric data

In [50]:
```python
_=df_numeric_ss.hist(bins=20,figsize=(25,10))
```



Outlier Treatment using z-score

In [51]:
```python
df_merge.shape
```

Out[51]: (14680, 15)

In [52]:
```python
df_merge_num = df_merge.select_dtypes(include=np.number)
```

In [53]:
```python
#find absolute value of z-score for each observation
z = np.abs(stats.zscore(df_merge_num))

#only keep rows in dataframe with all z-scores less than absolute value of 3
df_cleaned = df_merge_num[(z<3).all(axis=1)]

#find how many rows are left in the dataframe
df_cleaned.shape
```

Out[53]: (14530, 13)

Number of rows dropped due to outlier treatment = 150

In [54]:
```python
df_merge.shape[0] - df_cleaned.shape[0]
```

Out[54]: 150

Construction of data free of outliers and missing values

In [55]:
```python
df_merge = df_merge[df_merge.index.isin(df_cleaned.index)]

df_merge.shape
```

Out[55]: (14530, 15)

In [56]:
```python
df_merge.isna().sum()
```

Out[56]:
```
Track               0
Artist              0
danceability        0
energy              0
key                 0
mode                0
speechiness         0
acousticness        0
instrumentalness    0
liveness            0
valence             0
tempo               0
duration_ms         0
loudness            0
is_hit              0
dtype: int64
```

Export data for modeling

In [57]:
```python
df_merge.to_csv('bb_msd_merged.csv')

print('Merged dataframe exported........')
```

Merged dataframe exported........

Checking if the target data is Imbalanced or not. The ratio is around 63:37

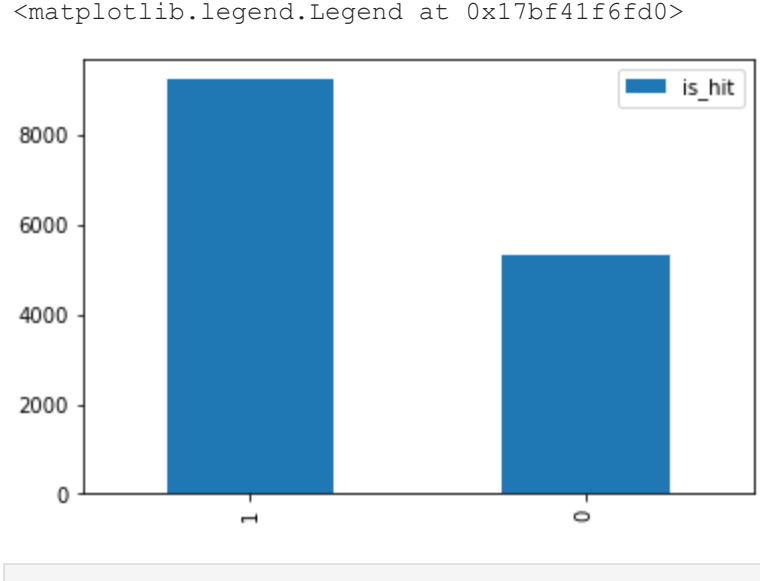
In [58]:
```python
df_merge['is_hit'].value_counts(normalize=True).to_frame()
```

Out[58]:

	is_hit
1	0.634893
0	0.365107

In [59]:
```python
df_merge['is_hit'].value_counts().plot(kind='bar')
plt.legend()
```

Out[59]: <matplotlib.legend.Legend at 0x17bf41f6fd0>



In [60]:
```python
df_merge.head()
```

Out[60]:

SpotifyID	Track	Artist	danceability	energy	key	mode	speechiness	acousticness	instrumentalness	liveness
2yScBrg0zVe6WnafJLfCIL	Firedance	Mystic Rhythms Band	0.637	0.876	8	1	0.0415	0.1060	0.313000	0.136
4Vghbo90YLD9BudOpMKW0K	Stop The World	Extreme	0.452	0.724	8	1	0.0435	0.0038	0.001120	0.687
7eqoqGkKwgOaWNNHx90uEZ	Nights	Frank Ocean	0.466	0.548	5	0	0.1180	0.4200	0.000001	0.113
6NhpIdjYoufuNNIBsgOztc	Suds In The Bucket	Sara Evans	0.511	0.903	10	1	0.0397	0.0520	0.000529	0.304
0LkvMWl5u8zu1vqsGCH6jG	Shrivasta	William Coulter	0.358	0.285	9	0	0.0381	0.9410	0.893000	0.121

Top 10 energetic songs

In [61]:
```python
top_10_energy = df_merge.sort_values(by='energy',ascending=False).head(10)
```

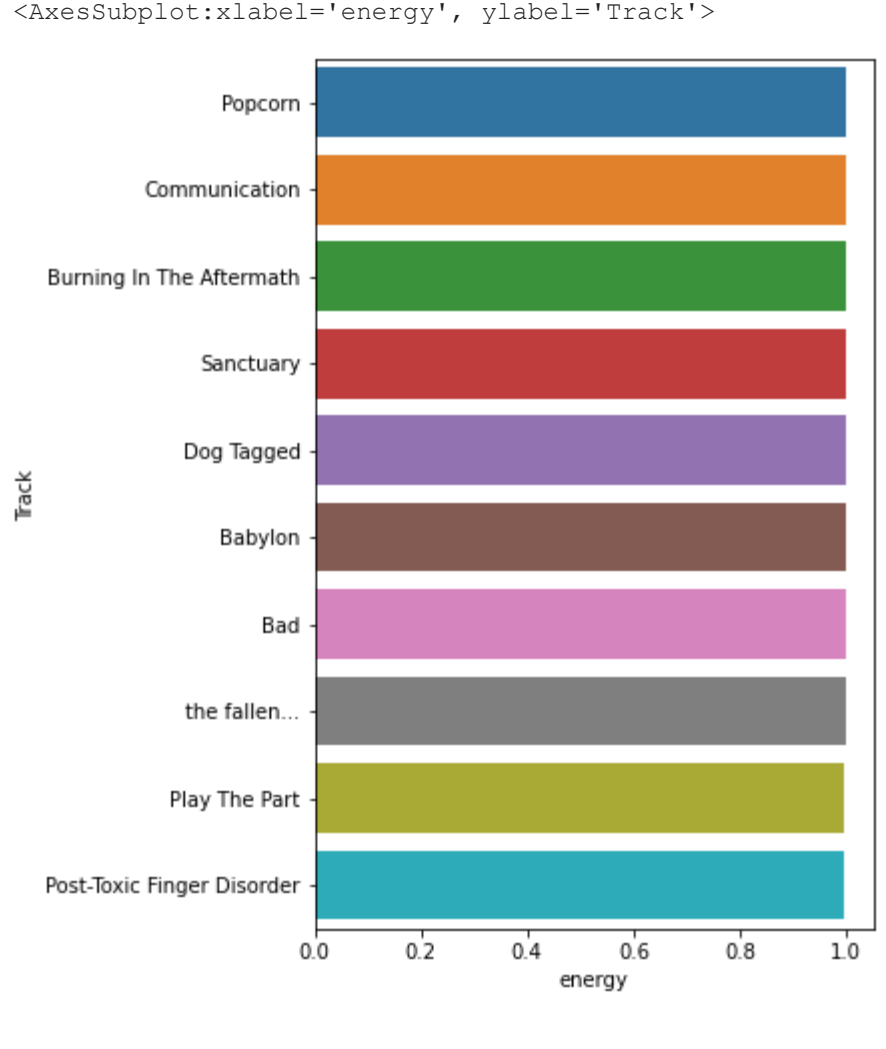
In [62]:
```python
top_10_energy.head()
```

Out[62]:

SpotifyID	Track	Artist	danceability	energy	key	mode	speechiness	acousticness	instrumentalness	liveness
1iT9oqeZGYOnspkV4AiloJ	Popcorn	Richi M.	0.816	1.000	7	0	0.0644	0.030000	0.852000	0.3600
608wNyUrCxorcPh4miyKAq	Communication	Mario Piu	0.662	0.999	4	0	0.0982	0.044700	0.812000	0.0949
3HEJpfNiNs7wsUxpMh9n0X	Burning In The Aftermath	The Suicide Machines	0.464	0.999	2	1	0.1560	0.001380	0.000000	0.2850
3TPcpHQ0LWUdP7x2ZIQCzZ	Sanctuary	Sick Of It All	0.337	0.999	10	1	0.1910	0.000004	0.000247	0.1150
56poKFI5j8kzPDO2M4BQHR	Dog Tagged	Kill The Client	0.157	0.998	7	1	0.2230	0.000002	0.949000	0.1860

In [63]:
```python
plt.figure(figsize=(5,8))
sns.barplot(y='Track',x='energy',data=top_10_energy)
```

Out[63]: <AxesSubplot:xlabel='energy', ylabel='Track'>



Top 10 danceable songs

In [64]:
```python
top_10_dance = df_merge.sort_values(by='danceability',ascending=False).head(10)
```

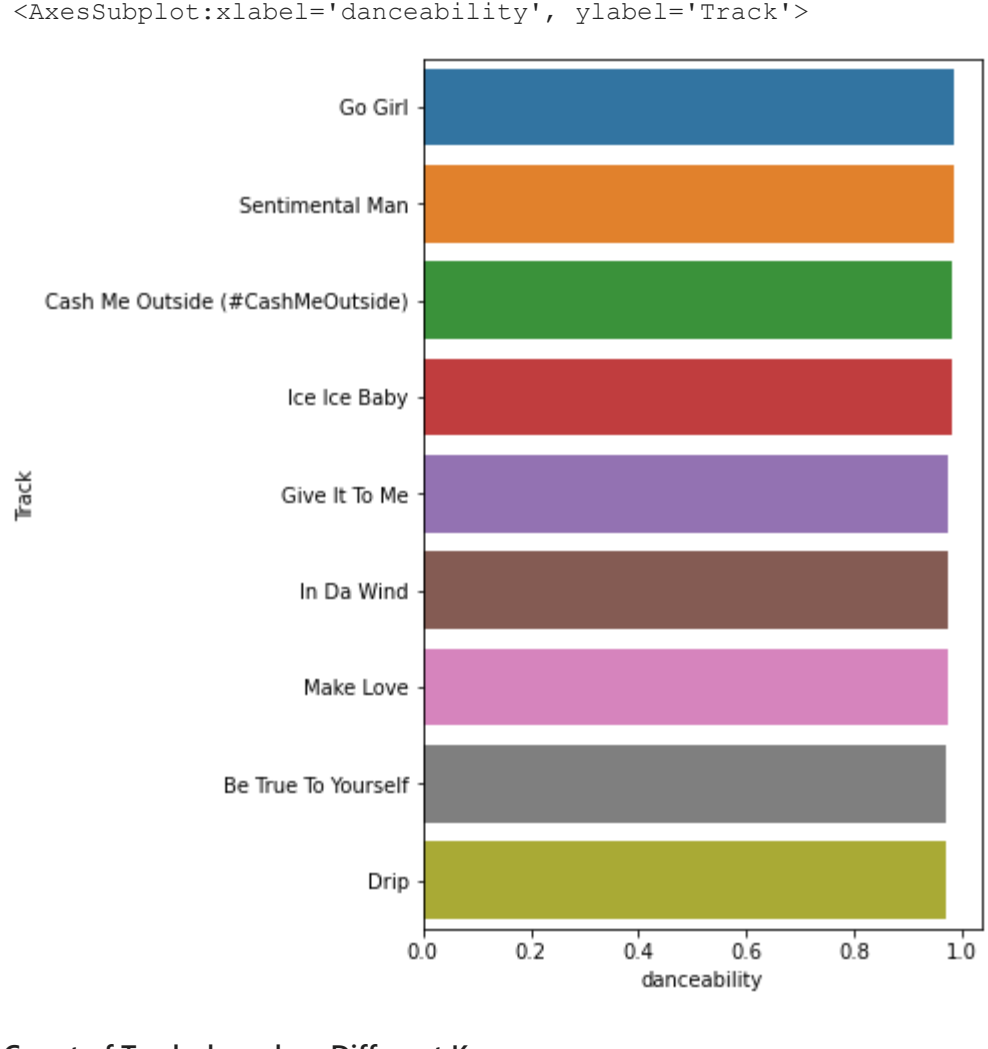
In [65]:
```python
top_10_dance.head()
```

Out[65]:

SpotifyID	Track	Artist	danceability	energy	key	mode	speechiness	acousticness	instrumentalness
1MgM0WDaoQ2A3bnIQTR6ty	Go Girl	Pitbull Featuring Trina & Young Bo$$	0.986	0.597	8	1	0.3080	0.1450	0.000002
7n0ShZeoewT4ZOJCmwHdnQ	Sentimental Man	Winston Reedy	0.984	0.395	11	1	0.0653	0.0236	0.000084
1r6cyaW7yMykwSyxTuwcuM	Cash Me Outside (#CashMeOutside)	DJ Suede The Remix God	0.981	0.417	11	0	0.1340	0.0471	0.359000
3kUkjtNjWG7jFEMIEPnVJq	Ice Ice Baby	Glee Cast	0.980	0.816	7	1	0.0831	0.0192	0.002980
3XVozq1aeqsJwpXrEZrDJ9	Ice Ice Baby	Vanilla Ice	0.979	0.792	2	1	0.0388	0.0025	0.000008

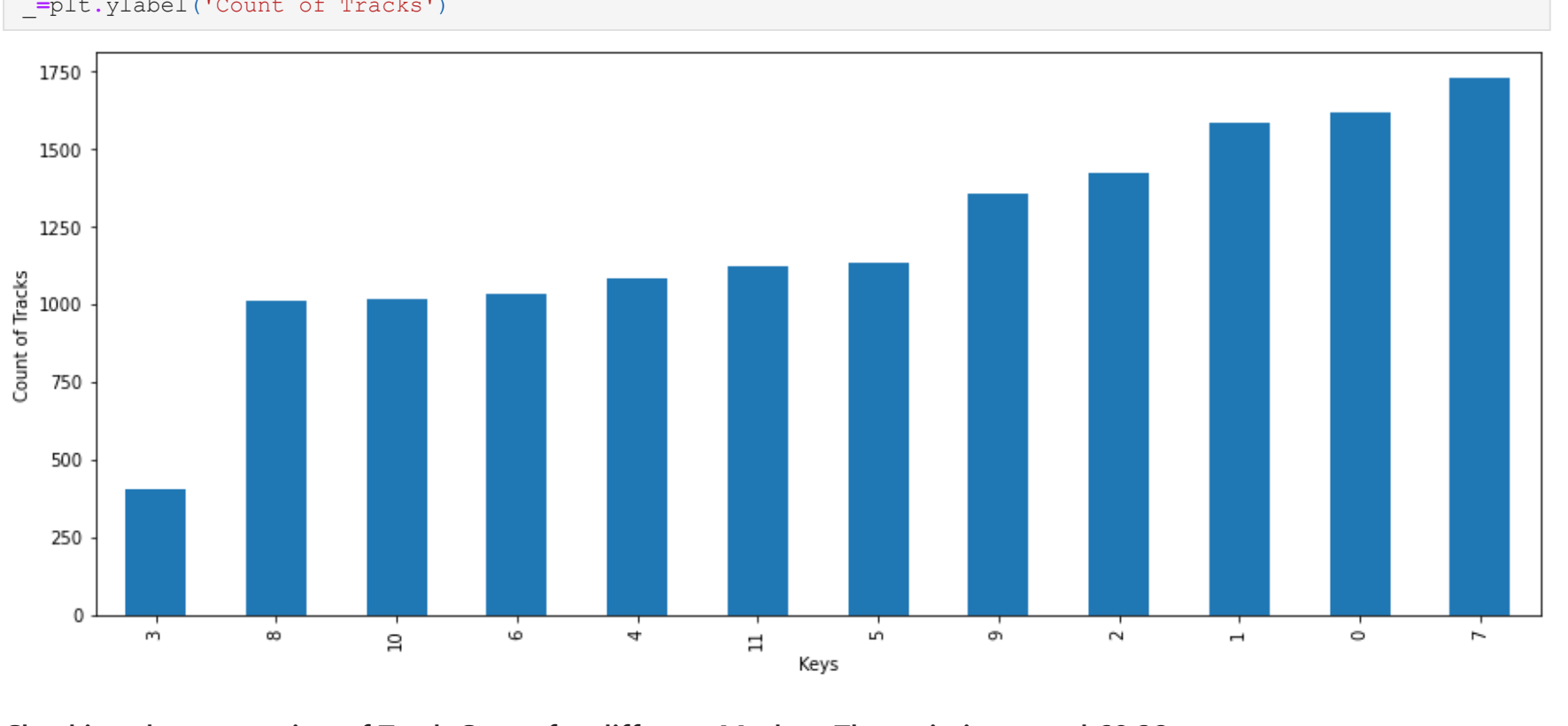
In [66]:
```python
plt.figure(figsize=(5,8))
sns.barplot(y='Track',x='danceability',data=top_10_dance)
```

Out[66]: <AxesSubplot:xlabel='danceability', ylabel='Track'>



Count of Tracks based on Different Keys

In [67]:
```python
df_merge['key'].value_counts().sort_values().plot(kind='bar',figsize=(15,6))
plt.xlabel('Keys')
_=plt.ylabel('Count of Tracks')
```



Checking the proportion of Track Count for different Modes . The ratio is around 68:32

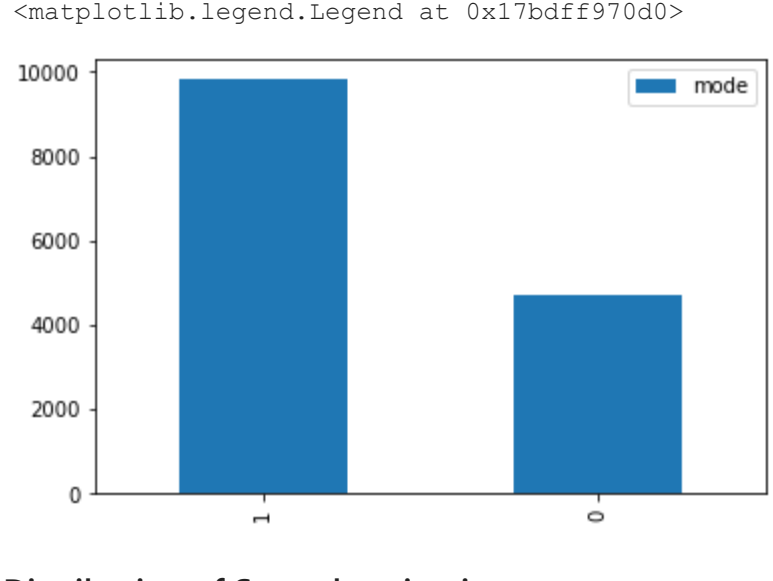
In [68]:
```python
df_merge['mode'].value_counts(normalize=True).to_frame()
```

Out[68]:

	mode
1	0.675155
0	0.324845

In [69]:
```python
df_merge['mode'].value_counts().plot(kind='bar')
plt.legend()
```

Out[69]: <matplotlib.legend.Legend at 0x17bdff970d0>

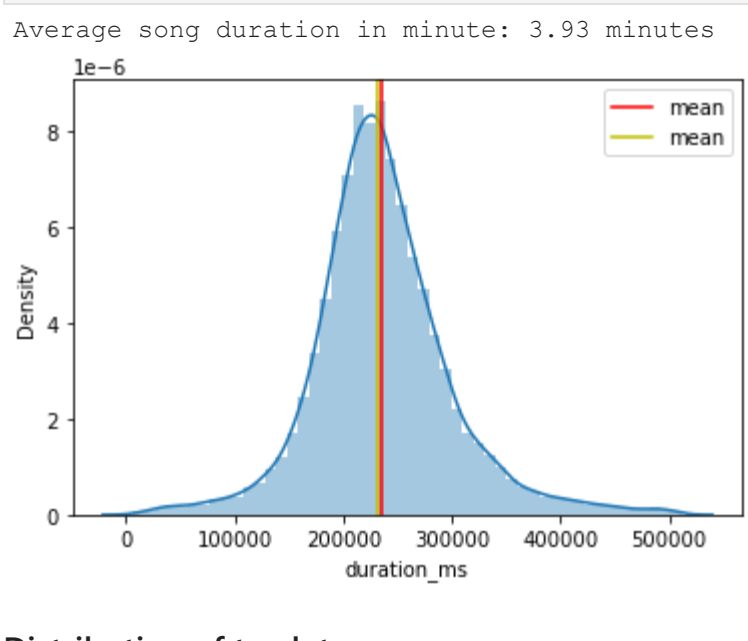


Distribution of Song duration in ms.

In [70]:
```python
avg_duration = df_merge['duration_ms'].mean()
median_duration = df_merge['duration_ms'].median()
```

In [71]:
```python
sns.distplot(df_merge['duration_ms'])
plt.axvline(avg_duration,label='mean',color='r')
plt.axvline(median_duration,label='mean',color='y')
plt.legend()

print('Average song duration in minute:',round(avg_duration/60000,2),'minutes')
```

Average song duration in minute: 3.93 minutes

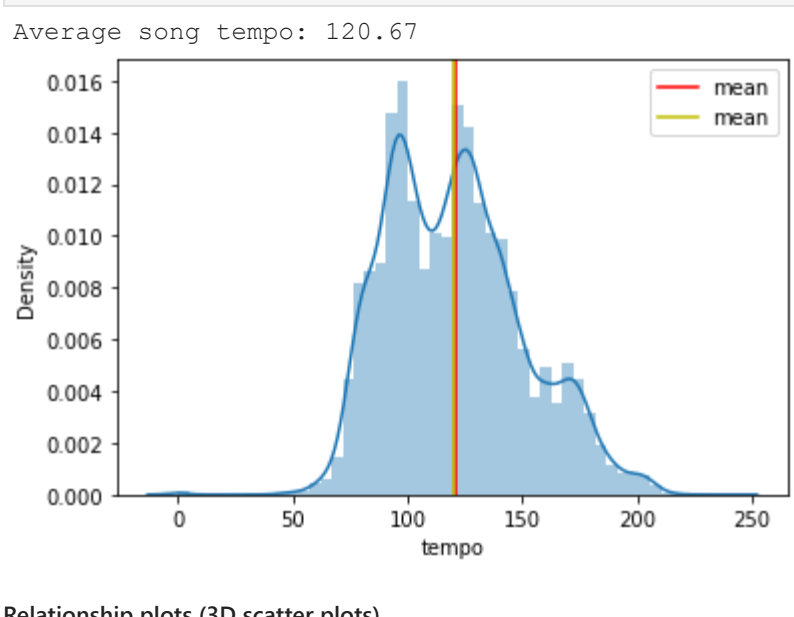


Distribution of track tempo.

In [72]:
```python
avg_tempo = df_merge['tempo'].mean()
median_tempo = df_merge['tempo'].median()
```

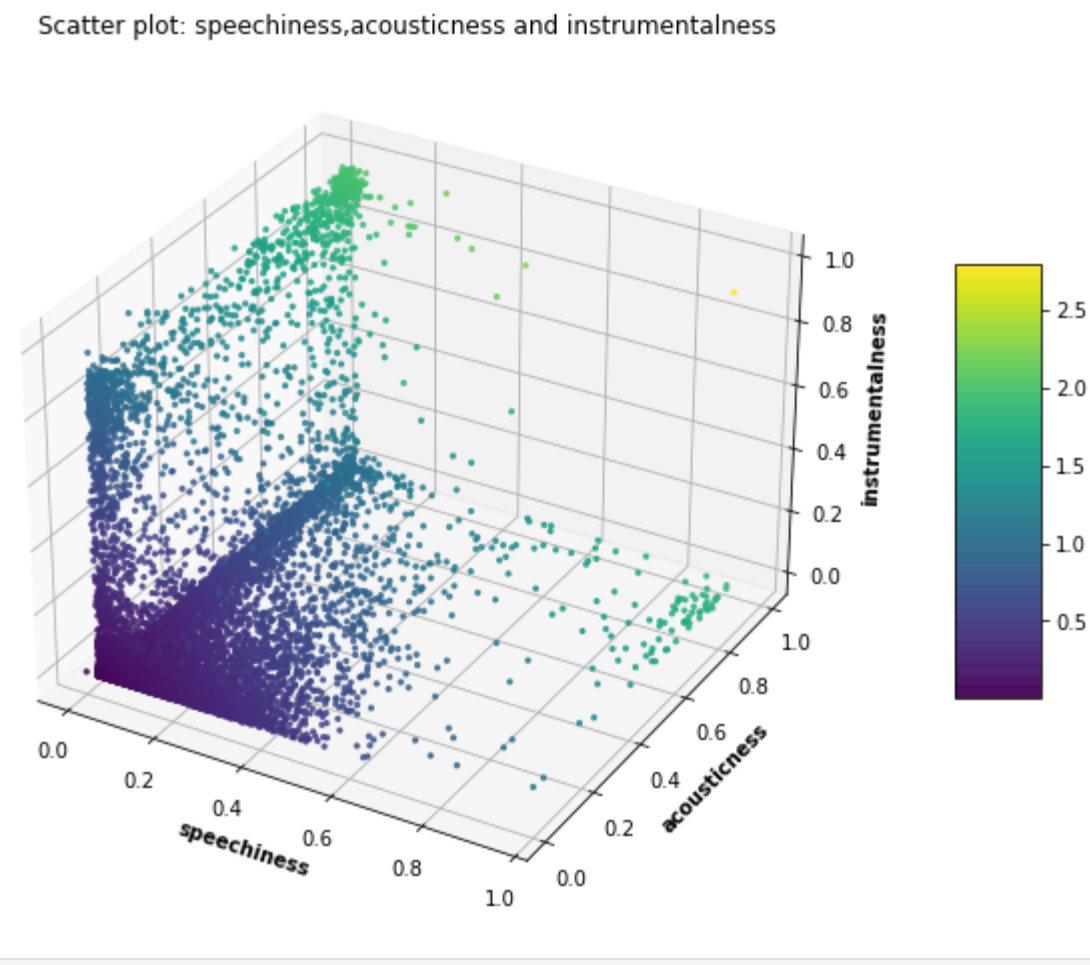
In [73]:
```python
sns.distplot(df_merge['tempo'])
plt.axvline(avg_tempo,label='mean',color='r')
plt.axvline(median_tempo,label='mean',color='y')
plt.legend()

print('Average song tempo:',round(avg_tempo,2))
```

Average song tempo: 120.67

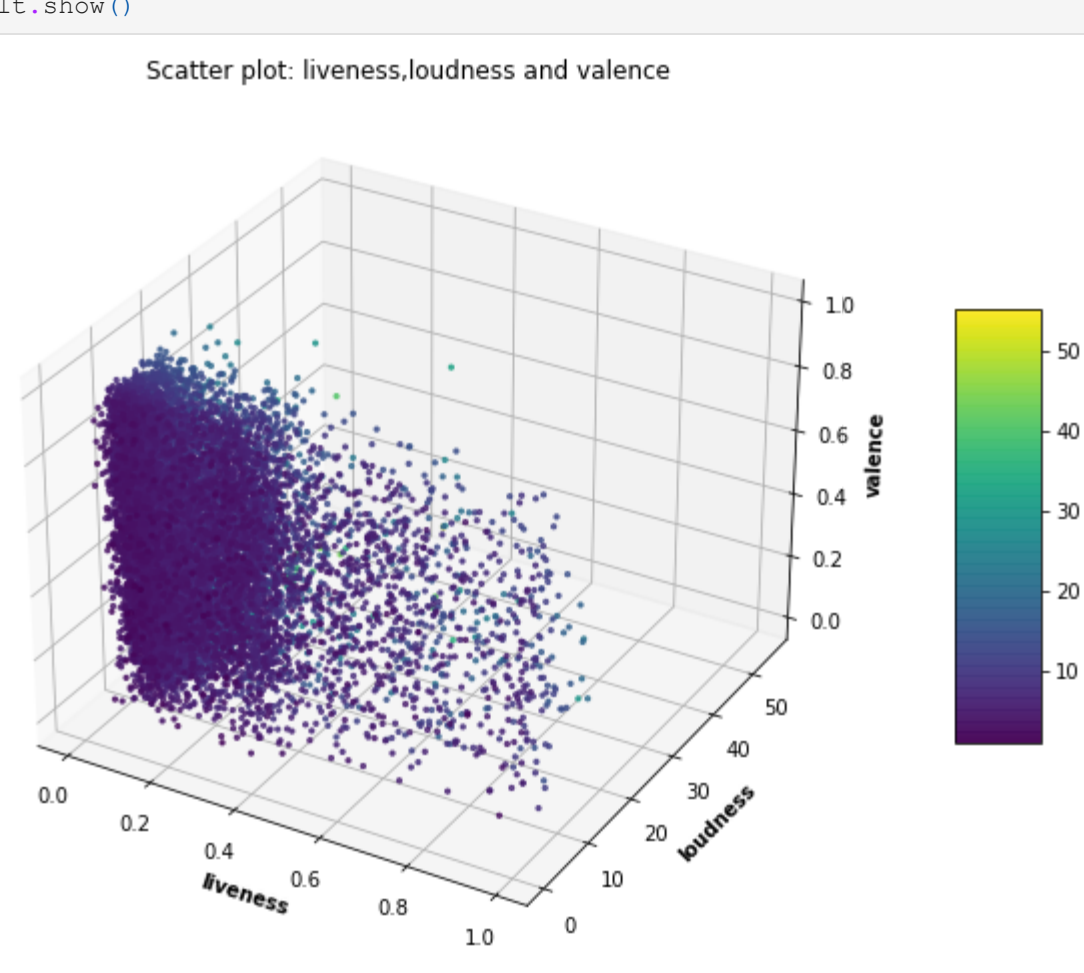


Relationship plots (3D scatter plots)

In [74]:
```python
x = df_merge['speechiness']

y = df_merge['acousticness']

z = df_merge['instrumentalness']


fig = plt.figure(figsize = (20, 8))
ax = plt.axes(projection ="3d")


sctt = ax.scatter3D(x, y, z,
                    alpha = 0.8,
                    c = (x + y + z),
                    marker ='.')

plt.title("Scatter plot: speechiness,acousticness and instrumentalness")

ax.set_xlabel('speechiness', fontweight ='bold')
ax.set_ylabel('acousticness', fontweight ='bold')
ax.set_zlabel('instrumentalness', fontweight ='bold')
fig.colorbar(sctt, ax = ax, shrink = 0.5, aspect = 5)

# show plot
plt.show()
```




```
In [75]: x = df_merge['liveness']

y = df_merge['loudness'].abs()

z = df_merge['valence']


fig = plt.figure(figsize = (20, 8))
ax = plt.axes(projection ="3d")


sctt = ax.scatter3D(x, y, z,
                    alpha = 0.8,
                    c = (x + y + z),
                    marker ='.')

plt.title("Scatter plot: liveness,loudness and valence")

ax.set_xlabel('liveness', fontweight ='bold')
ax.set_ylabel('loudness', fontweight ='bold')
ax.set_zlabel('valence', fontweight ='bold')
fig.colorbar(sctt, ax = ax, shrink = 0.5, aspect = 5)

# show plot
plt.show()
```



Relationship plots (2D scatter plots)

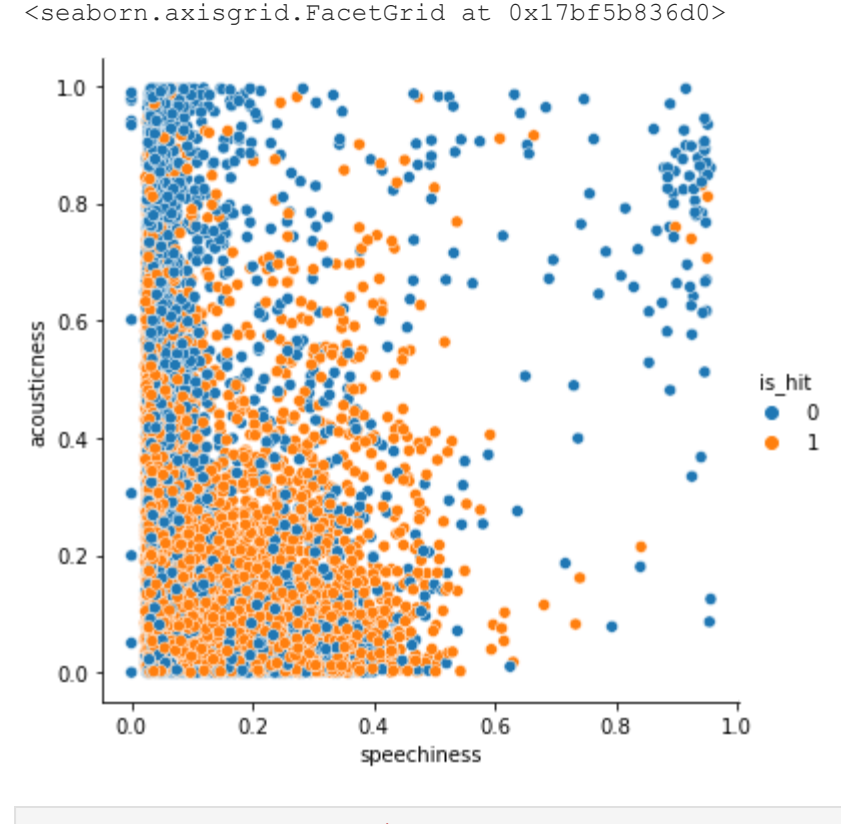
```
In [76]: df_merge.head()
```

Out[76]:

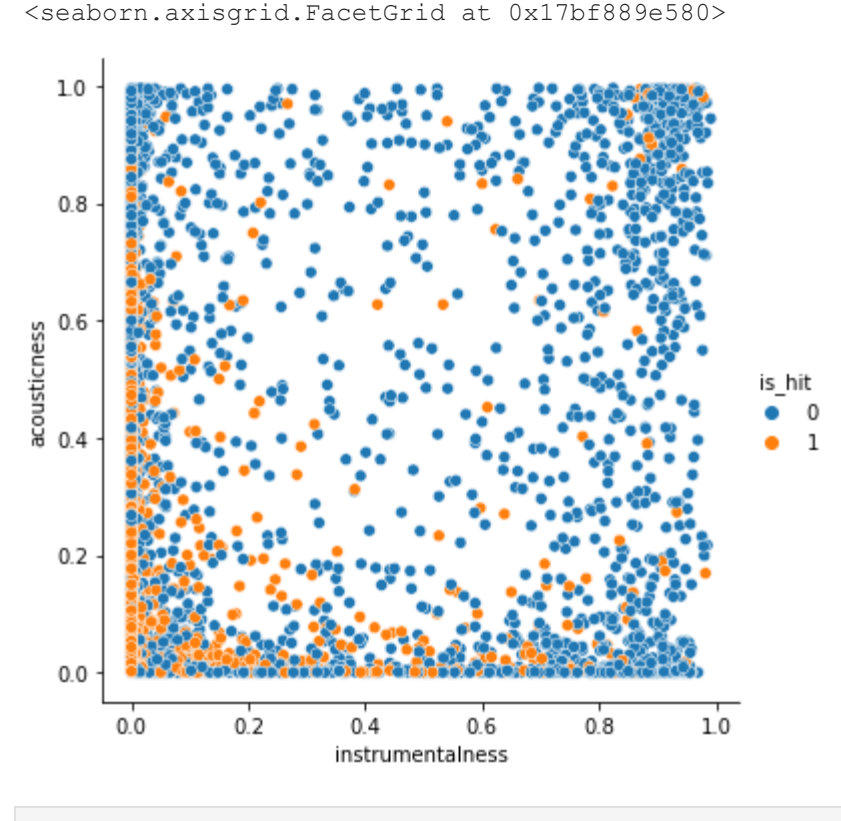
SpotifyID	Track	Artist	danceability	energy	key	mode	speechiness	acousticness	instrumentalness	liveness	vale
2yScBrg0zVe6WnafJLfCIL	Firedance	Mystic Rhythms Band	0.637	0.876	8	1	0.0415	0.1060	0.313000	0.136	0.
4Vghbo90YLD9BudOpMKW0K	Stop The World	Extreme	0.452	0.724	8	1	0.0435	0.0038	0.001120	0.687	0.
7eqoqGkKwgOaWNNHx90uEZ	Nights	Frank Ocean	0.466	0.548	5	0	0.1180	0.4200	0.000001	0.113	0.
6NhpIdjYoufuNNlBsgOztc	Suds In The Bucket	Sara Evans	0.511	0.903	10	1	0.0397	0.0520	0.000529	0.304	0.
0LkvMWl5u8zu1vqsGCH6jG	Shrivasta	William Coulter	0.358	0.285	9	0	0.0381	0.9410	0.893000	0.121	0.

```
In [77]: sns.relplot(df_merge['danceability'],df_merge['energy'],hue=df_merge['is_hit'])
Out[77]: <seaborn.axisgrid.FacetGrid at 0x17bf8722460>
```

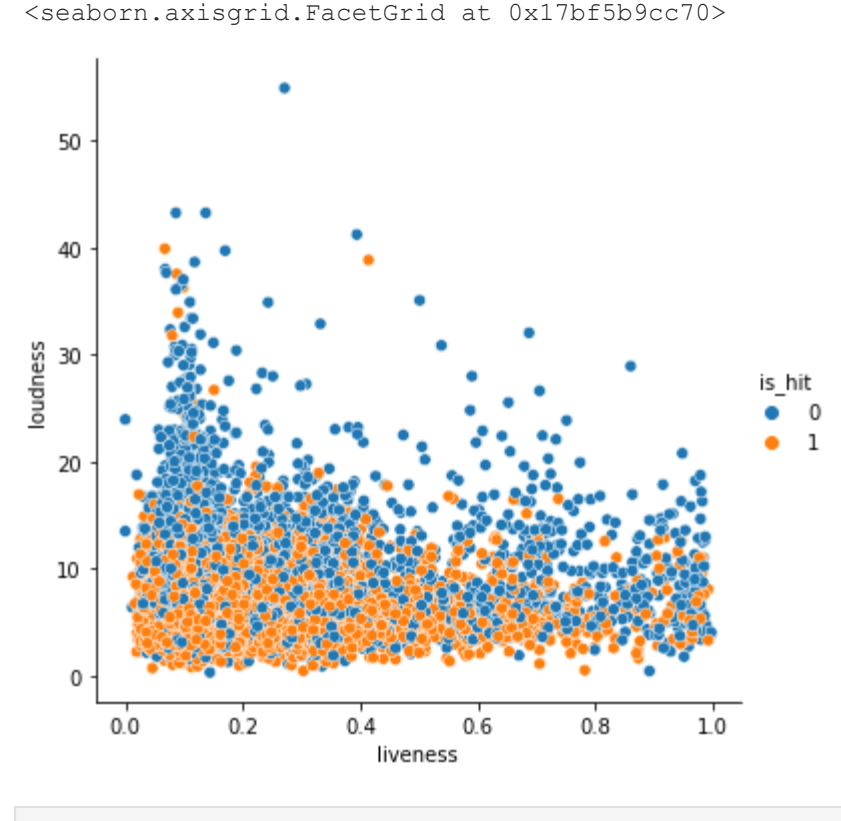

```
In [78]: sns.relplot(df_merge['speechiness'],df_merge['acousticness'],hue=df_merge['is_hit'])
Out[78]: <seaborn.axisgrid.FacetGrid at 0x17bf5b836d0>
```



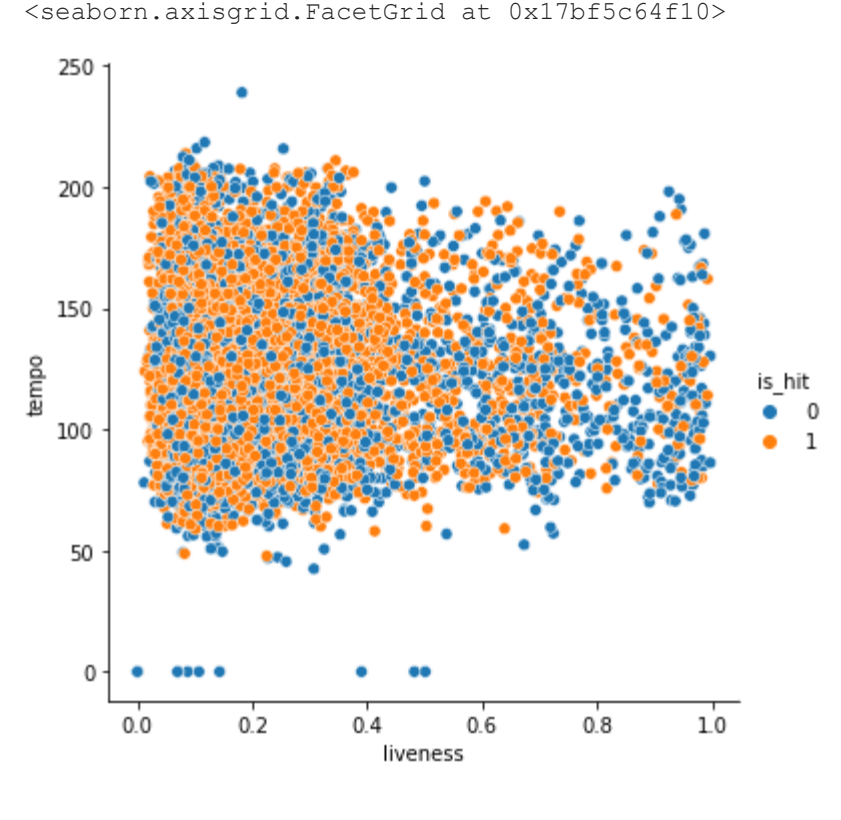

```
In [79]: sns.relplot(df_merge['instrumentalness'],df_merge['acousticness'],hue=df_merge['is_hit'])
Out[79]: <seaborn.axisgrid.FacetGrid at 0x17bf889e580>
```




```
In [80]: sns.relplot(df_merge['liveness'],df_merge['loudness'].abs(),hue=df_merge['is_hit'])
Out[80]: <seaborn.axisgrid.FacetGrid at 0x17bf5b9cc70>
```

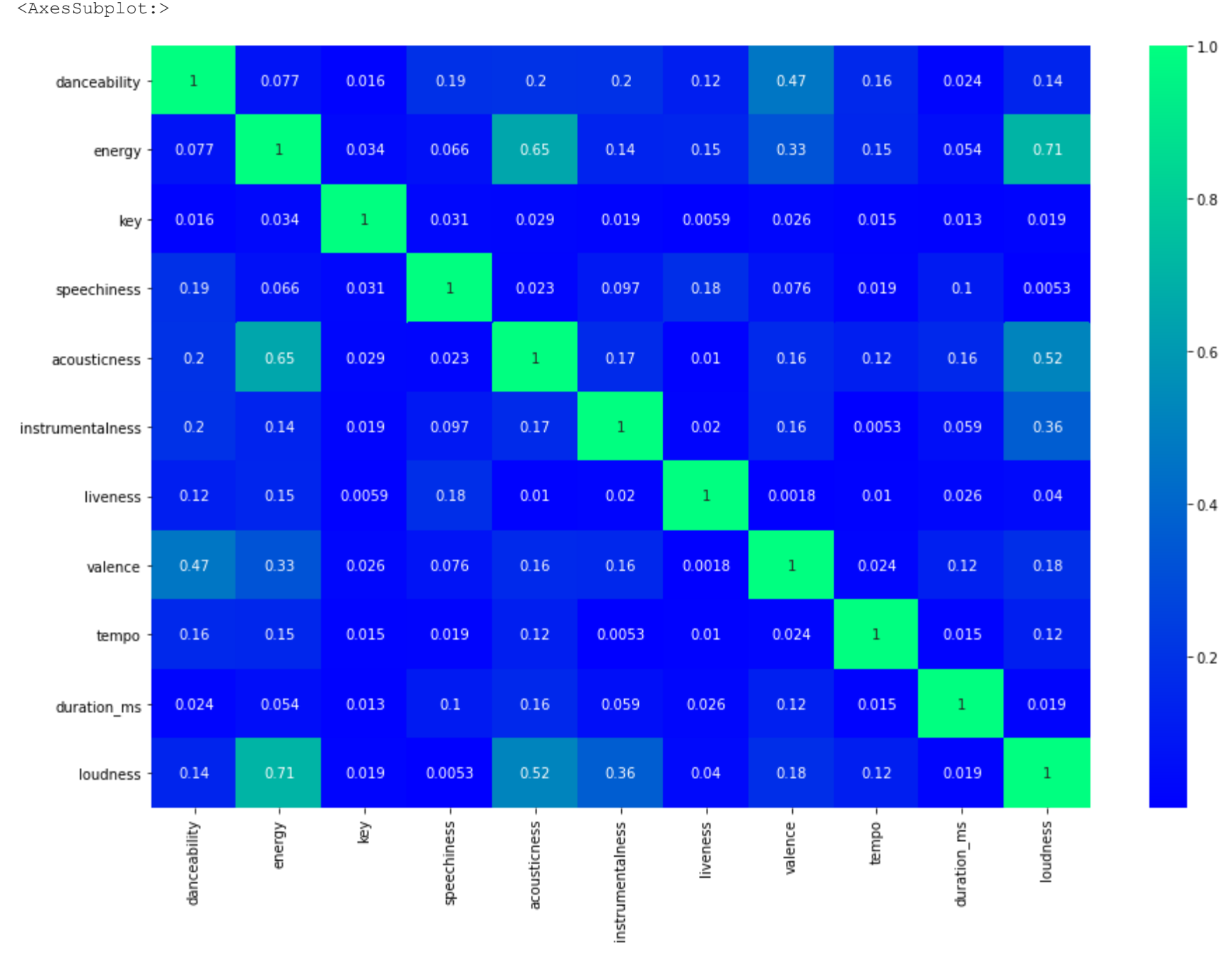



```
In [81]: sns.relplot(df_merge['liveness'],df_merge['tempo'].abs(),hue=df_merge['is_hit'])
Out[81]: <seaborn.axisgrid.FacetGrid at 0x17bf5c64f10>
```



Overall Relationship plot : correlation heatmap.

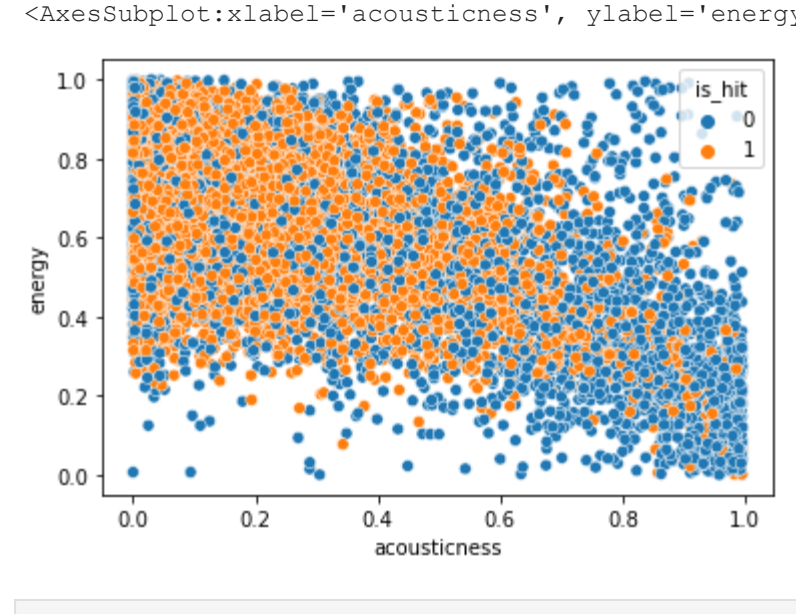
```
In [82]: df_num = df_merge.select_dtypes(include=np.number).drop(['mode','is_hit'],axis=1)
plt.figure(figsize=(15,10))
sns.heatmap(df_num.corr().abs(),annot=True,cmap='winter')
Out[82]: <AxesSubplot:>
```



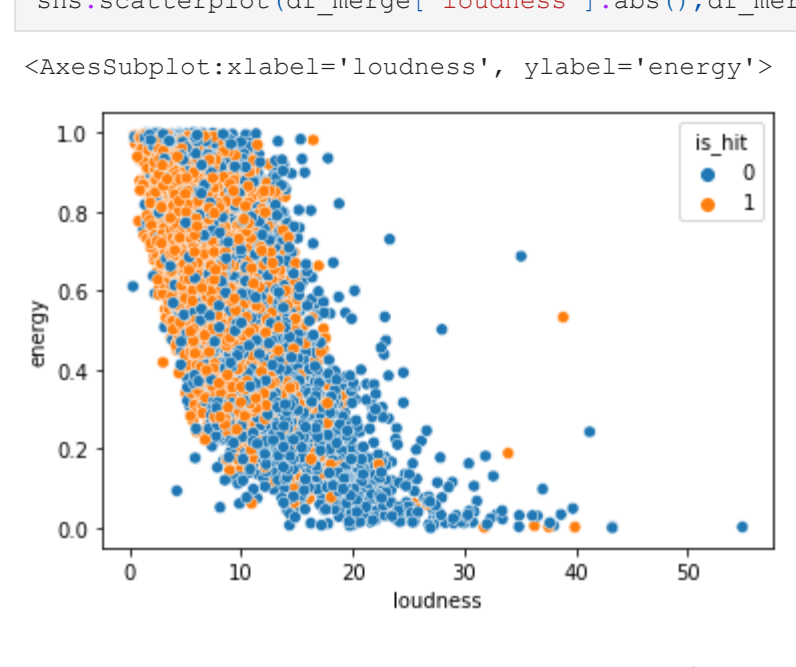
acousticness and energy , loudness and energy are the only pair of features which are having significant correlation among them.

let's revisit these relations.

```
In [83]: sns.scatterplot(df_merge['acousticness'],df_merge['energy'],hue=df_merge['is_hit'])
Out[83]: <AxesSubplot:xlabel='acousticness', ylabel='energy'>
```




```
In [84]: sns.scatterplot(df_merge['loudness'].abs(),df_merge['energy'],hue=df_merge['is_hit'])
Out[84]: <AxesSubplot:xlabel='loudness', ylabel='energy'>
```



Hypothesis tests to check the dependency of numerical features and categorical features on target variable.

- H0: features are not-dependent with target variable.
- Ha: features are dependent with target variable.
- Consideration:
  - Confidence Interval: 95% (significance level : 5%)
  - Numerical features : 2-sample z-test
  - categorical features : chi-square test of independence

```
In [85]: df_merge.head(2)
```

Out[85]:

SpotifyID	Track	Artist	danceability	energy	key	mode	speechiness	acousticness	instrumentalness	liveness	vale
2yScBrg0zVe6WnafJLfCIL	Firedance	Mystic Rhythms Band	0.637	0.876	8	1	0.0415	0.1060	0.31300	0.136	0.
4Vghbo90YLD9BudOpMKW0K	Stop The World	Extreme	0.452	0.724	8	1	0.0435	0.0038	0.00112	0.687	0.

```
In [86]: g1 = df_merge[df_merge['is_hit']==0]['danceability']
g2 = df_merge[df_merge['is_hit']==1]['danceability']

In [87]: stests.ztest(g1,g2)[1]
Out[87]: 1.2427110394554739e-202

In [88]: df_merge_num = df_merge_num.drop(['key','mode','is_hit'],axis=1)
```

List of all numerical features

```
In [89]: df_merge_num.columns
Out[89]: Index(['danceability', 'energy', 'speechiness', 'acousticness',
       'instrumentalness', 'liveness', 'valence', 'tempo', 'duration_ms',
       'loudness'],
      dtype='object')

In [90]: p_values_list=[]
for col in df_merge_num.columns:
    g1 = df_merge[df_merge['is_hit']==0][col]
    g2 = df_merge[df_merge['is_hit']==1][col]
    p_value = stests.ztest(g1,g2)[1]
    p_values_list.append(p_value)
dependency_list=['dependent'  if p<0.05 else 'not-dependent' for p in p_values_list ]

pd.DataFrame({'Feature':df_merge_num.columns, 'P-values':p_values_list,'Dependency with target':dependency_list
```

Out[90]:

	Feature	P-values	Dependency with target
0	danceability	1.242711e-202	dependent
1	energy	1.002509e-102	dependent
2	speechiness	8.908876e-01	not-dependent
3	acousticness	3.256461e-304	dependent
4	instrumentalness	0.000000e+00	dependent
5	liveness	1.469094e-31	dependent
6	valence	4.847784e-02	dependent
7	tempo	3.855218e-02	dependent
8	duration_ms	1.100337e-17	dependent
9	loudness	0.000000e+00	dependent

List of categorical features

```
In [91]: df_merge.head(1)
```

Out[91]:

SpotifyID	Track	Artist	danceability	energy	key	mode	speechiness	acousticness	instrumentalness	liveness	valence
2yScBrg0zVe6WnafJLfCIL	Firedance	Mystic Rhythms Band	0.637	0.876	8	1	0.0415	0.106	0.313	0.136	0.672

```
In [92]: df_merge_cat = df_merge[['key','mode']]

df_merge_cat.columns
Out[92]: Index(['key', 'mode'], dtype='object')

In [93]: p_vals=[]
for col in df_merge_cat.columns:
    df1 = pd.crosstab(df_merge['is_hit'],df_merge[col])
    p_val = stats.chi2_contingency(df1)[1]
    p_vals.append(p_val)
dependency_list_cat=['dependent'  if p<0.05 else 'not-dependent' for p in p_vals ]

pd.DataFrame({'Feature':df_merge_cat.columns, 'P-values':p_vals,'Dependency with target':dependency_list_cat})
```

Out[93]:

	Feature	P-values	Dependency with target
0	key	3.708598e-33	dependent
1	mode	1.135764e-02	dependent

From the above hypothesis test with 95% Confidence Interval we can clearly see that apart from speechiness all are significant predictors of target variable.

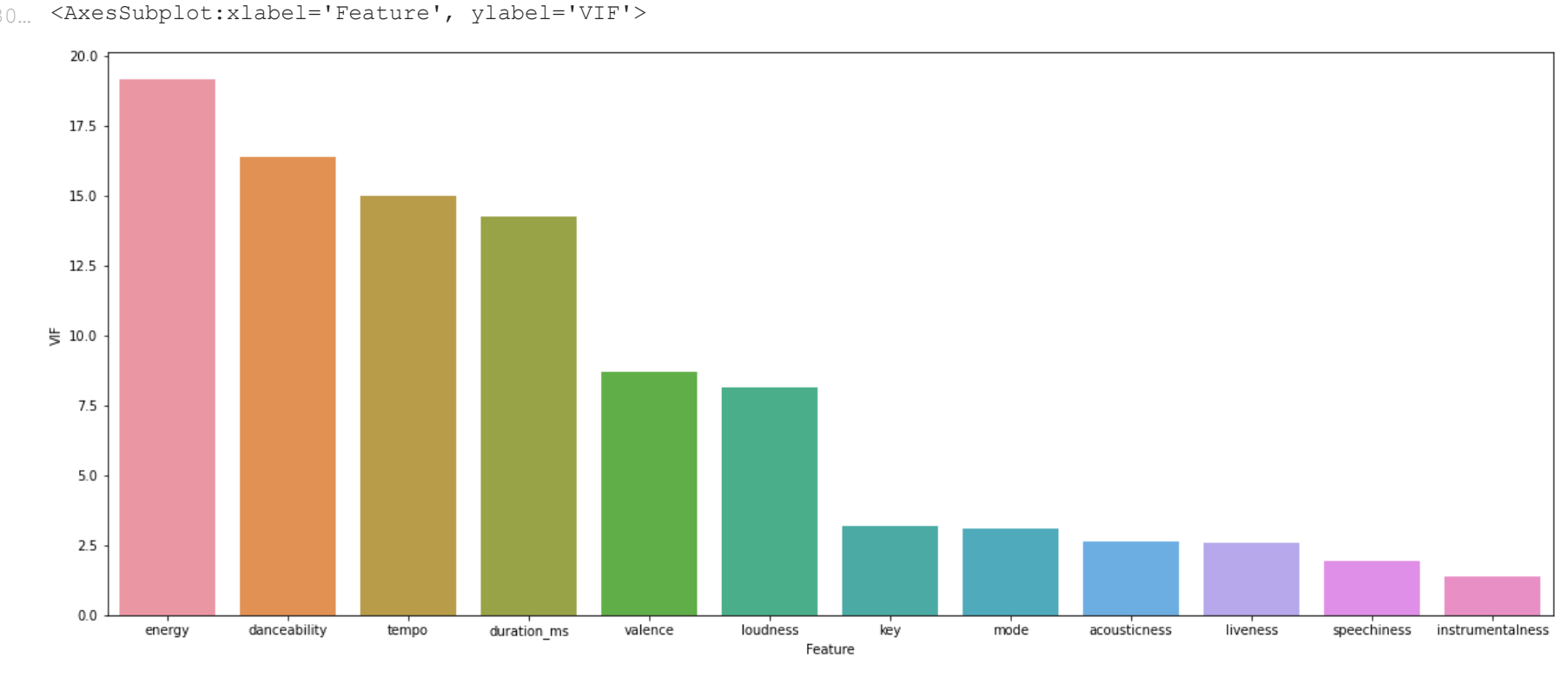
Multicollinearity check : energy column is having highest VIF factor but not significantly large to be dropped.

```
In [124…: from statsmodels.stats.outliers_influence import variance_inflation_factor

X = df_merge.drop(['Track','Artist','is_hit'],axis=1)

# VIF dataframe
vif_data = pd.DataFrame()
vif_data["Feature"] = X.columns

# calculating VIF for each feature
vif_data["VIF"] = [variance_inflation_factor(X.values, i)
                          for i in range(len(X.columns))]

vif_data = vif_data.sort_values(by='VIF',ascending=False)
vif_data
```

Out[124…:

	Feature	VIF
1	energy	19.171371
0	danceability	16.374929
9	tempo	15.011448
10	duration_ms	14.263656
8	valence	8.691957
11	loudness	8.142193
2	key	3.209195
3	mode	3.111841
5	acousticness	2.622257
7	liveness	2.586114
4	speechiness	1.939668
6	instrumentalness	1.393568

```
In [130…: plt.figure(figsize=(20,8))
sns.barplot(x='Feature',y='VIF',data=vif_data)
Out[130…: <AxesSubplot:xlabel='Feature', ylabel='VIF'>
```



Conclusion :

- Columns Track and Artist needs to be dropped .
- As per statistical significance 'speechiness' column is not a good predictor and can be dropped.
- As per vif factor column 'energy' is involved in multicollinearity and can be dropped.
- Data is moderately balanced , so oversampling not required.
- There is no clear difference of target variable among numerical features visually. no inference can be drawn prior to model building.