data (1).csv

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

import plotly.graph_objs as go
import plotly.express as px

import warnings
warnings.simplefilter("ignore")
```

In [2]:
```
sn.set_style('darkgrid')
```

In [3]:
```
df=pd.read_csv('data (1).csv')

# Dropping the unnamed column
df.drop(columns='Unnamed: 0', axis=1, inplace=True)
```

In [4]:
```
df.head()
```

Out[4]:

	acousticness	danceability	duration_ms	energy	instrumentalness	key	liveness	loudness	r
0	0.0102	0.833	204600	0.434	0.021900	2	0.1650	-8.795	
1	0.1990	0.743	326933	0.359	0.006110	1	0.1370	-10.401	
2	0.0344	0.838	185707	0.412	0.000234	2	0.1590	-7.148	
3	0.6040	0.494	199413	0.338	0.510000	5	0.0922	-15.236	
4	0.1800	0.678	392893	0.561	0.512000	5	0.4390	-11.648	

In [5]:
```
df.shape
```

Out[5]:
```
(2017, 16)
```

In [6]:
```
df.columns
```

Out[6]:
```
Index(['acousticness', 'danceability', 'duration_ms', 'energy',
       'instrumentalness', 'key', 'liveness', 'loudness', 'mode',
       'speechiness', 'tempo', 'time_signature', 'valence', 'target',
       'song_title', 'artist'],
      dtype='object')
```

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

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2017 entries, 0 to 2016
Data columns (total 16 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   acousticness      2017 non-null   float64
 1   danceability      2017 non-null   float64
 2   duration_ms       2017 non-null   int64  
 3   energy            2017 non-null   float64
 4   instrumentalness  2017 non-null   float64
 5   key               2017 non-null   int64  
 6   liveness          2017 non-null   float64
 7   loudness          2017 non-null   float64
 8   mode              2017 non-null   int64  
 9   speechiness       2017 non-null   float64
 10  tempo             2017 non-null   float64
 11  time_signature    2017 non-null   float64
 12  valence           2017 non-null   float64
 13  target            2017 non-null   int64  
 14  song_title        2017 non-null   object 
 15  artist            2017 non-null   object 
dtypes: float64(10), int64(4), object(2)
memory usage: 252.3+ KB
```

In [8]:
```
df.isnull().sum()
```

Out[8]:
```
acousticness        0
danceability        0
duration_ms         0
energy              0
instrumentalness    0
key                 0
liveness            0
loudness            0
mode                0
speechiness         0
tempo               0
time_signature      0
valence             0
target              0
song_title          0
artist              0
dtype: int64
```

In [9]:
```
len(df.columns)
```

Out[9]:
```
16
```

In [10]:
```
df['target'].value_counts()
```

Out[10]:
```
target
1    1020
0     997
Name: count, dtype: int64
```

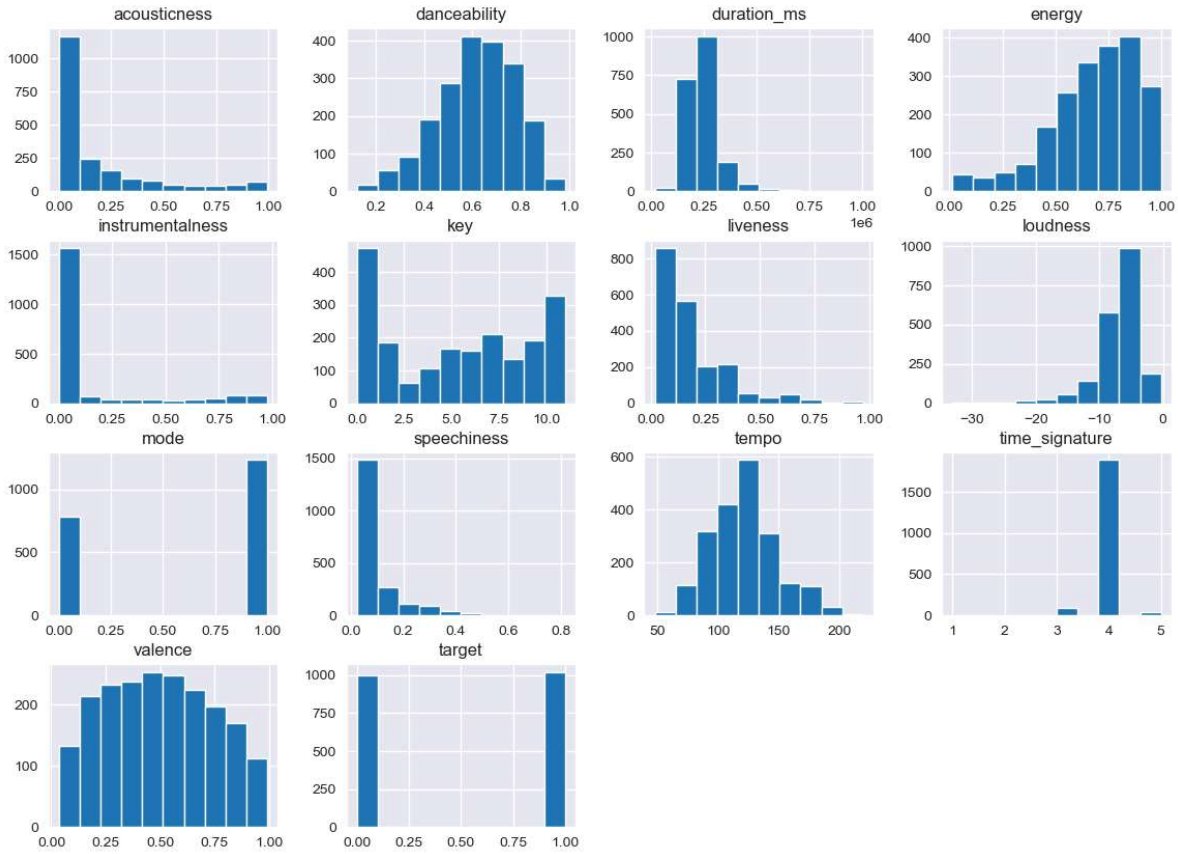
In [15]:
```
df.describe()
```

Out[15]:

	acousticness	danceability	duration_ms	energy	instrumentalness	key	
count	2017.000000	2017.000000	2.017000e+03	2017.000000	2017.000000	2017.000000	20
mean	0.187590	0.618422	2.463062e+05	0.681577	0.133286	5.342588	
std	0.259989	0.161029	8.198181e+04	0.210273	0.273162	3.648240	
min	0.000003	0.122000	1.604200e+04	0.014800	0.000000	0.000000	
25%	0.009630	0.514000	2.000150e+05	0.563000	0.000000	2.000000	
50%	0.063300	0.631000	2.292610e+05	0.715000	0.000076	6.000000	
75%	0.265000	0.738000	2.703330e+05	0.846000	0.054000	9.000000	
max	0.995000	0.984000	1.004627e+06	0.998000	0.976000	11.000000	

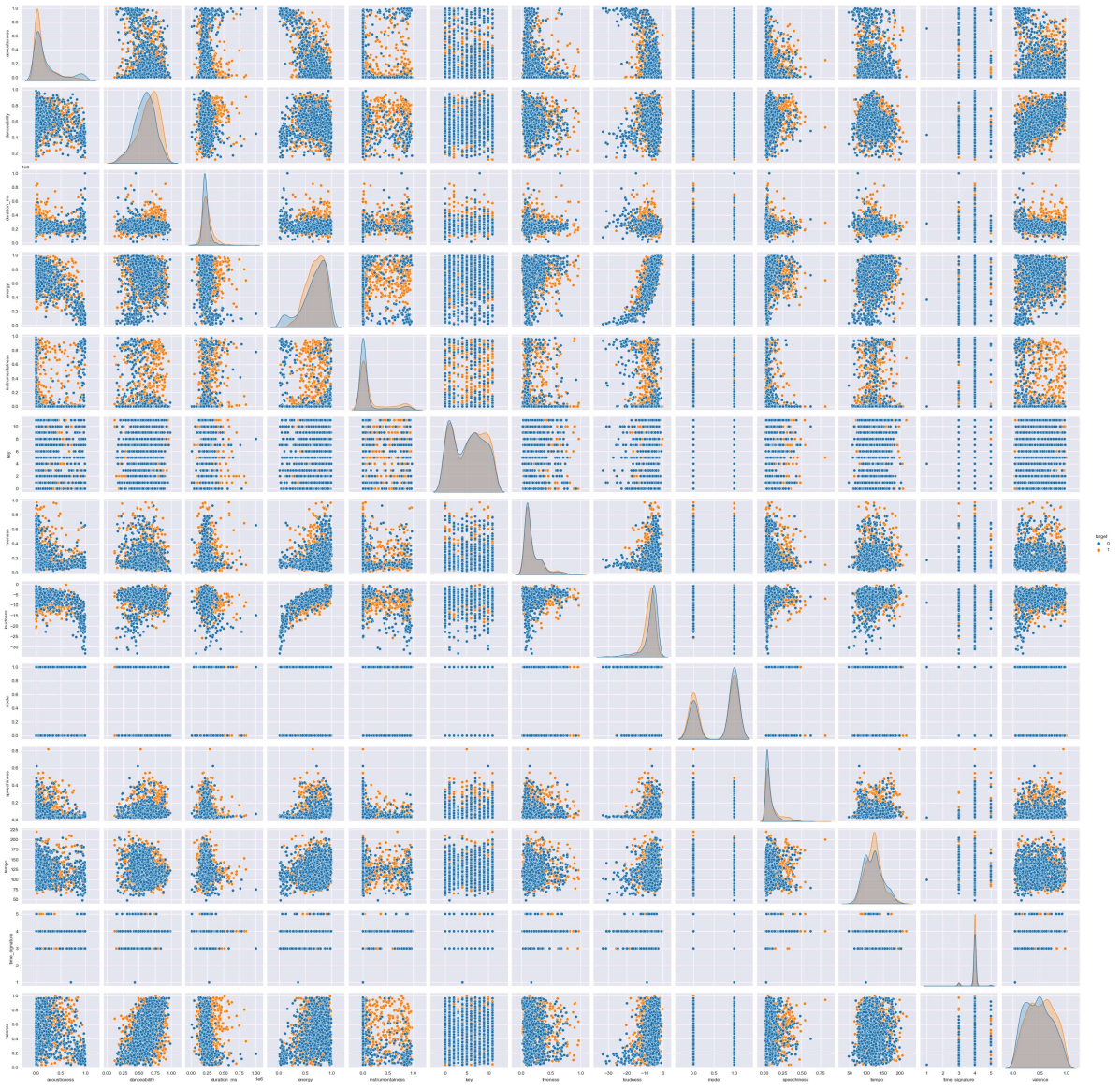
In [16]:
```
df.hist(figsize=(14,10))
```

Out[16]:
```
array([[<Axes: title={'center': 'acousticness'}>,
        <Axes: title={'center': 'danceability'}>,
        <Axes: title={'center': 'duration_ms'}>,
        <Axes: title={'center': 'energy'}>],
       [<Axes: title={'center': 'instrumentalness'}>,
        <Axes: title={'center': 'key'}>,
        <Axes: title={'center': 'liveness'}>,
        <Axes: title={'center': 'loudness'}>],
       [<Axes: title={'center': 'mode'}>,
        <Axes: title={'center': 'speechiness'}>,
        <Axes: title={'center': 'tempo'}>,
        <Axes: title={'center': 'time_signature'}>],
       [<Axes: title={'center': 'valence'}>,
        <Axes: title={'center': 'target'}>, <Axes: >, <Axes: >]],
      dtype=object)
```



In [19]:
```
plt.figure(figsize=(7,6))

sn.pairplot(data=df, hue='target')
```

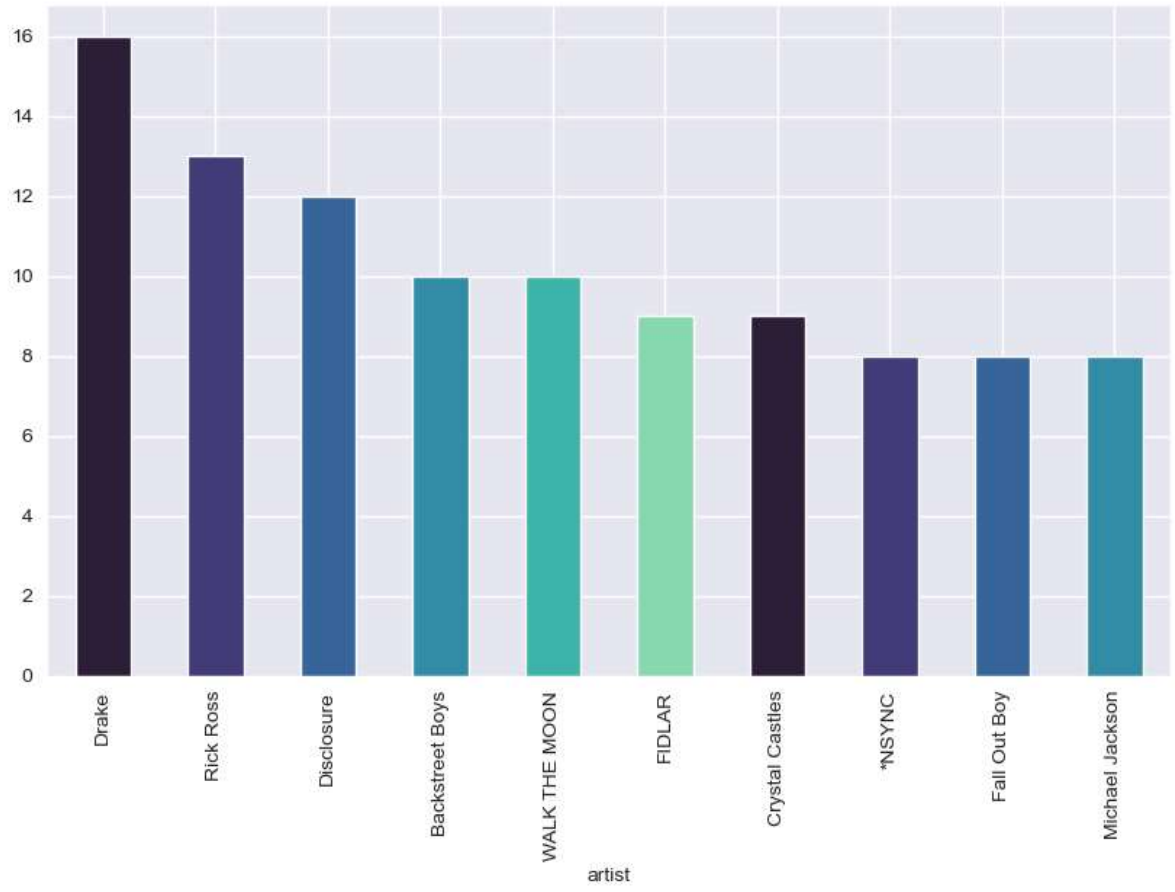
Out[19]:
```
<seaborn.axisgrid.PairGrid at 0x224443c2a10>

<Figure size 700x600 with 0 Axes>
```



In [20]:
```
top_10_artist=df.groupby('artist').count().sort_values(by='song_title', ascend
top_10_artist
```

Out[20]:
```
artist
Drake              16
Rick Ross          13
Disclosure         12
Backstreet Boys    10
WALK THE MOON      10
FIDLAR              9
Crystal Castles     9
*NSYNC              8
Fall Out Boy        8
Michael Jackson     8
Name: song_title, dtype: int64
```

In [23]:
```
top_10_artist.plot.bar(figsize=(10,6), color=sn.color_palette('mako'))
plt.show()
```



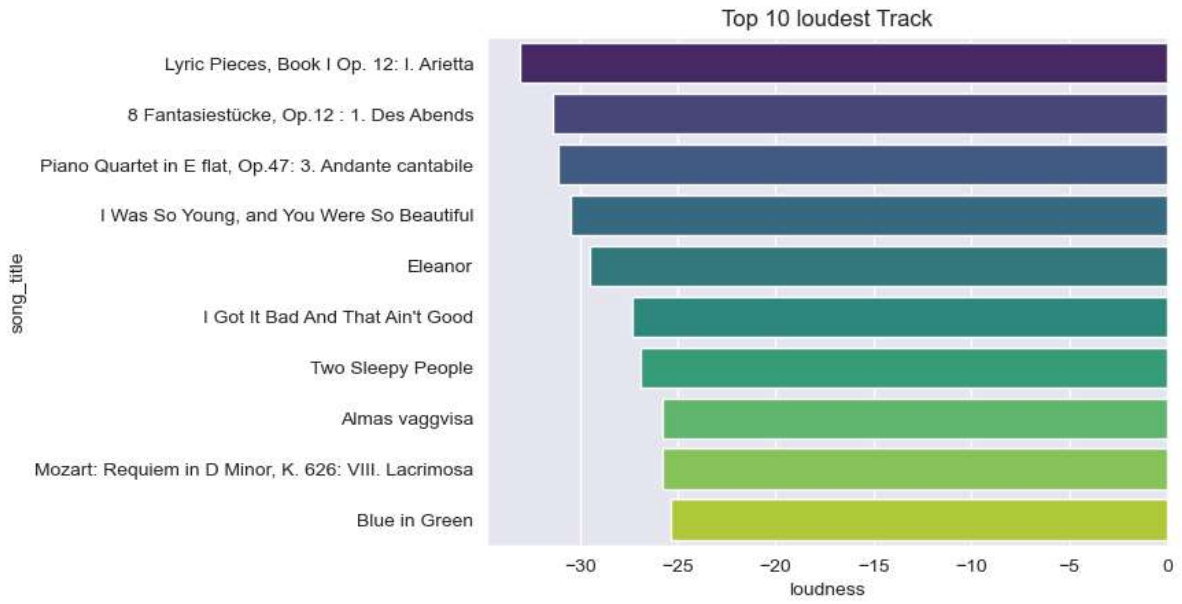
In [24]:
```
loud_10_track=df[['loudness', 'song_title']].sort_values(by='loudness', ascend
loud_10_track
```

Out[24]:

	loudness	song_title
1594	-33.097	Lyric Pieces, Book I Op. 12: I. Arietta
1596	-31.367	8 Fantasiestücke, Op.12 : 1. Des Abends
1598	-31.082	Piano Quartet in E flat, Op.47: 3. Andante can...
1531	-30.447	I Was So Young, and You Were So Beautiful
1549	-29.460	Eleanor
1571	-27.350	I Got It Bad And That Ain't Good
1541	-26.924	Two Sleepy People
1546	-25.766	Almas vaggvisa
817	-25.756	Mozart: Requiem in D Minor, K. 626: VIII. Lacr...
1537	-25.358	Blue in Green

In [25]:
```
sn.barplot(x='loudness', y='song_title', data=loud_10_track, palette='viridis'

plt.title('Top 10 loudest Track')
```

Out[25]:
```
Text(0.5, 1.0, 'Top 10 loudest Track')
```



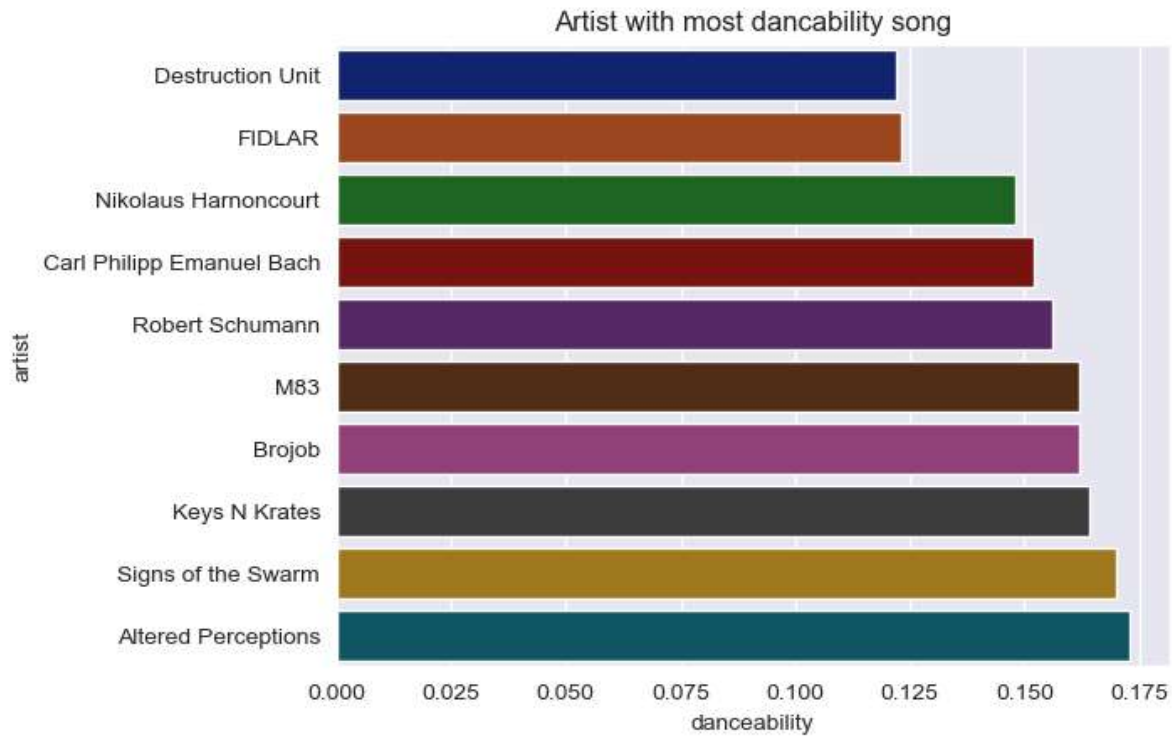
In [26]:
```
dance_song=df[['danceability','artist']].sort_values(by='danceability', ascend
dance_song
```

Out[26]:

	danceability	artist
729	0.122	Destruction Unit
532	0.123	FIDLAR
817	0.148	Nikolaus Harnoncourt
1600	0.152	Carl Philipp Emanuel Bach
1598	0.156	Robert Schumann
522	0.162	M83
1348	0.162	Brojob
1464	0.164	Keys N Krates
1313	0.170	Signs of the Swarm
1306	0.173	Altered Perceptions

In [27]:
```
sn.barplot(x='danceability',y='artist', data=dance_song, palette='dark')

plt.title('Artist with most dancability song')
```

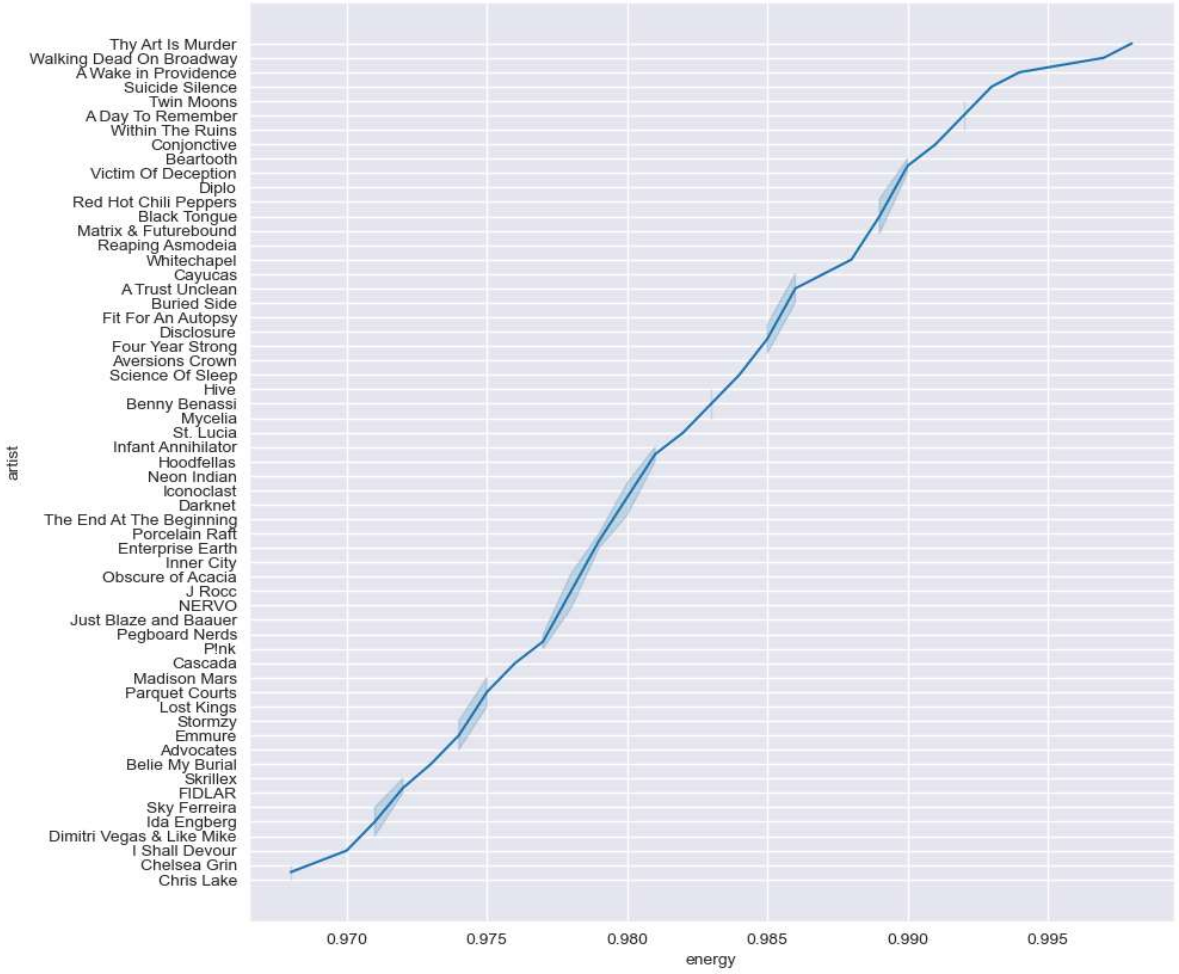
Out[27]:
```
Text(0.5, 1.0, 'Artist with most dancability song')
```



In [28]:
```
df5=df[['artist','energy']].sort_values(by='energy', ascending=False)[:60]
```

In [29]:
```
plt.figure(figsize=(10,10))
sn.lineplot(x='energy', y='artist', data=df5)
```

Out[29]: <Axes: xlabel='energy', ylabel='artist'>



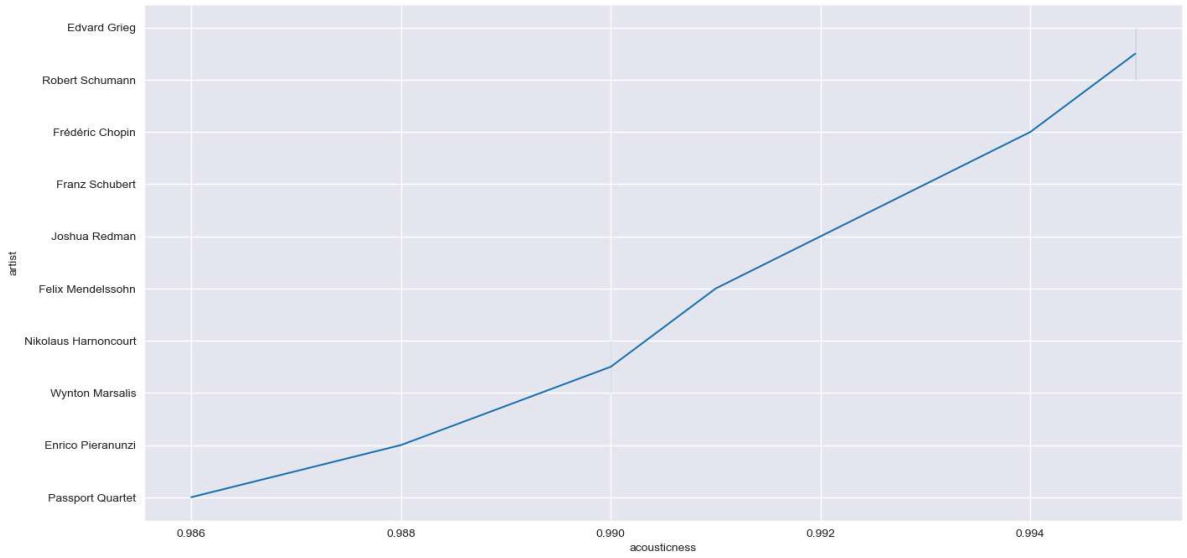
In [30]:
```
df6=df[['artist','acousticness', 'song_title']].sort_values(by='acousticness',
df6
```

Out[30]:

	artist	acousticness	song_title
1594	Edvard Grieg	0.995	Lyric Pieces, Book I Op. 12: I. Arietta
1596	Robert Schumann	0.995	8 Fantasiestücke, Op.12 : 1. Des Abends
1876	Frédéric Chopin	0.994	Nocturne No.1 In B Flat Minor, Op.9 No.1
1599	Franz Schubert	0.993	Piano Sonata No. 18 in G Major, Op. 78, D. 894...
1535	Joshua Redman	0.992	The Nearness of You
1603	Felix Mendelssohn	0.991	Viola Sonata in C Minor: III. Andante con vari...
817	Nikolaus Harnoncourt	0.990	Mozart: Requiem in D Minor, K. 626: VIII. Lacr...
1538	Wynton Marsalis	0.990	Where or When
1528	Enrico Pieranunzi	0.988	I Got It Bad
1530	Passport Quartet	0.986	I'm a Fool to Want You

In [31]:
```
plt.figure(figsize=(16,8))
sn.lineplot(x='acousticness',y='artist', data=df6)
```

Out[31]: <Axes: xlabel='acousticness', ylabel='artist'>



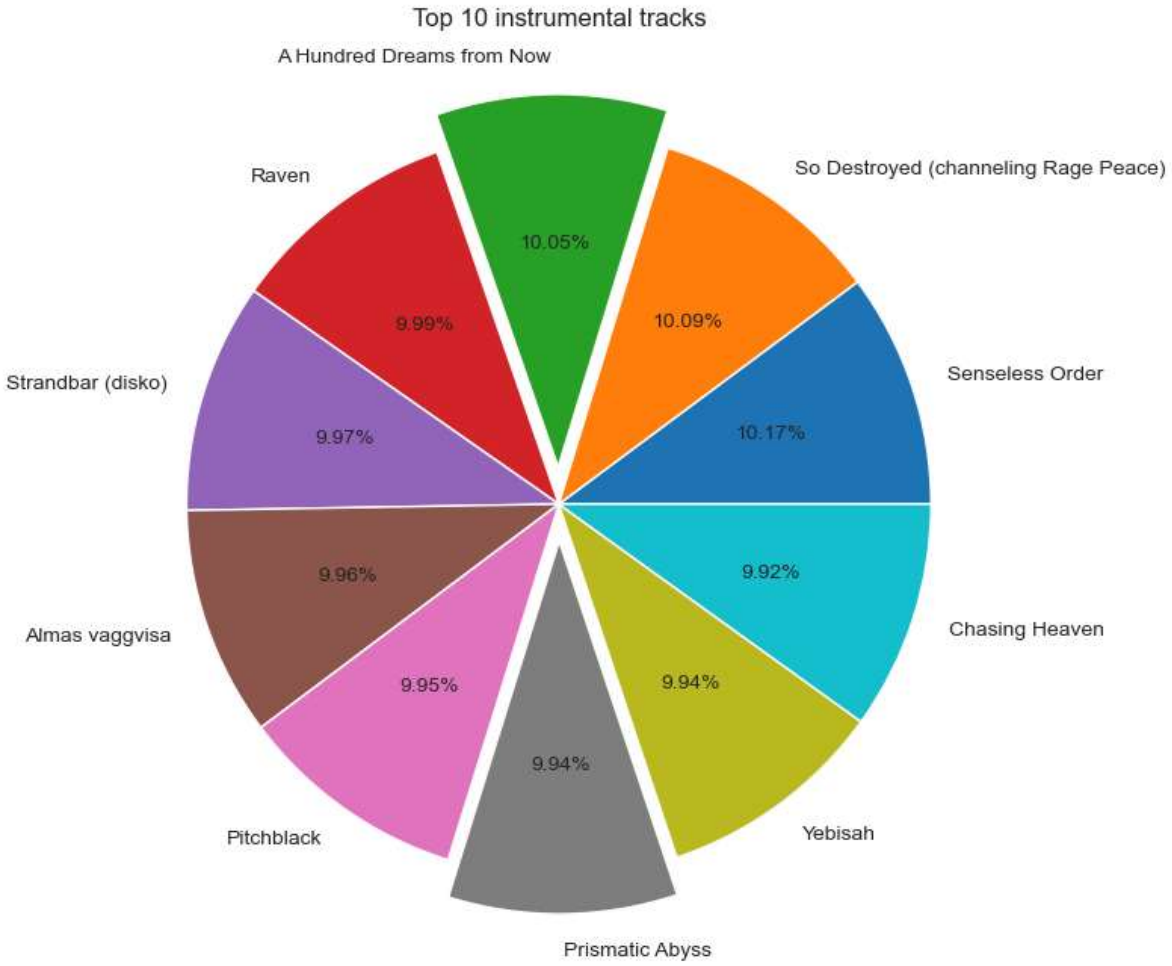
In [32]:
```
ins=df[['instrumentalness','artist','song_title']].sort_values(by='instrumenta
ins
```

Out[32]:

	instrumentalness	artist	song_title
1313	0.976	Signs of the Swarm	Senseless Order
271	0.968	Prince Rama	So Destroyed (channeling Rage Peace)
1575	0.964	Ray Bryant	A Hundred Dreams from Now
1619	0.958	John Dahlbäck	Raven
725	0.957	Todd Terje	Strandbar (disko)
1546	0.956	Jan Lundgren Trio	Almas vaggvisa
1322	0.955	Walking Dead On Broadway	Pitchblack
1349	0.954	Aversions Crown	Prismatic Abyss
1661	0.954	Mark Knight	Yebisah
121	0.952	Bassnectar	Chasing Heaven

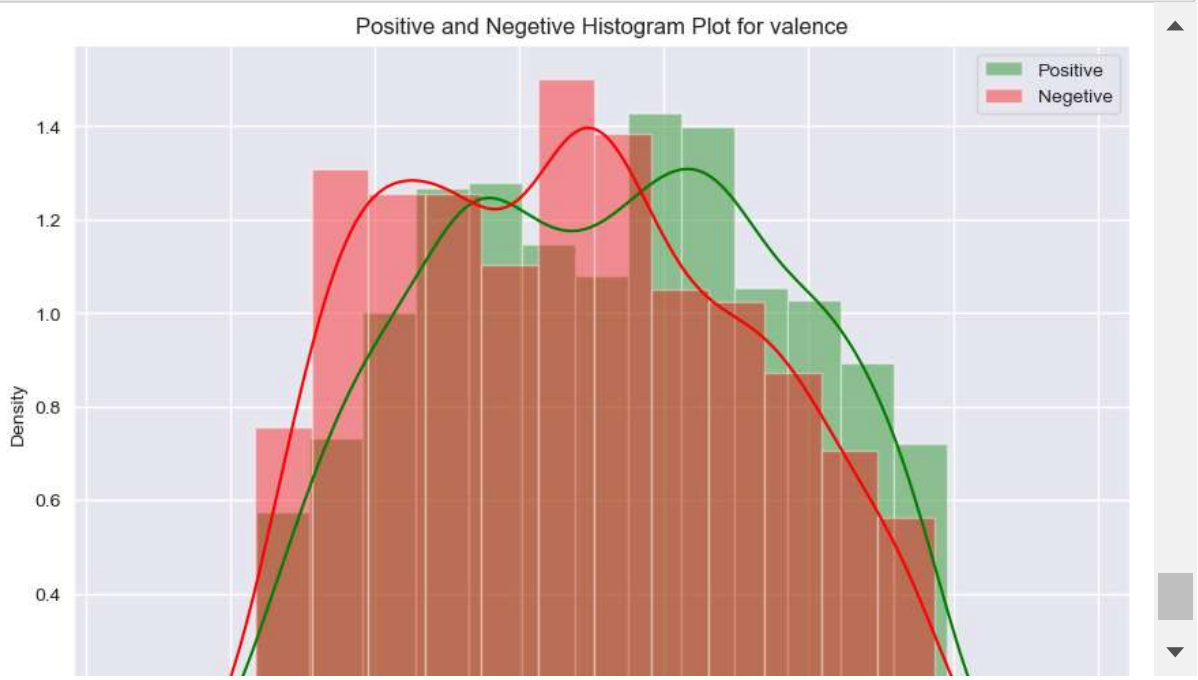
In [33]:
```
plt.figure(figsize=(10,8))
plt.pie(x='instrumentalness', data=ins,labels=ins.song_title, autopct="%.2f%%"

plt.title('Top 10 instrumental tracks')
```

Out[33]:
```
Text(0.5, 1.0, 'Top 10 instrumental tracks')
```



In [34]:
```
int_feature_cols=['tempo','loudness','acousticness','danceability','duration_m
                  'instrumentalness','liveness','speechiness','valence']
```

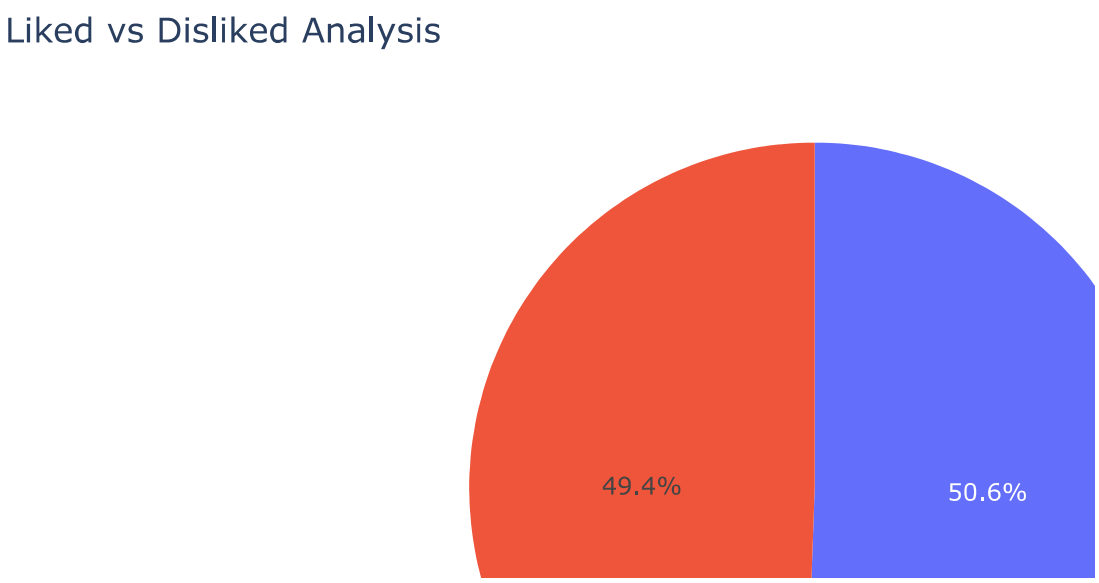
In [35]:
```
for feature_col in int_feature_cols:
  pos_data=df[df['target']==1][feature_col]
  neg_data=df[df['target']==0][feature_col]

  plt.figure(figsize=(10,7))
  sn.distplot(pos_data, label='Positive', color='green')
  sn.distplot(neg_data, label='Negetive', color='red')

  plt.legend(loc='upper right')
  plt.title(f"Positive and Negetive Histogram Plot for {feature_col}")
  plt.show()
```



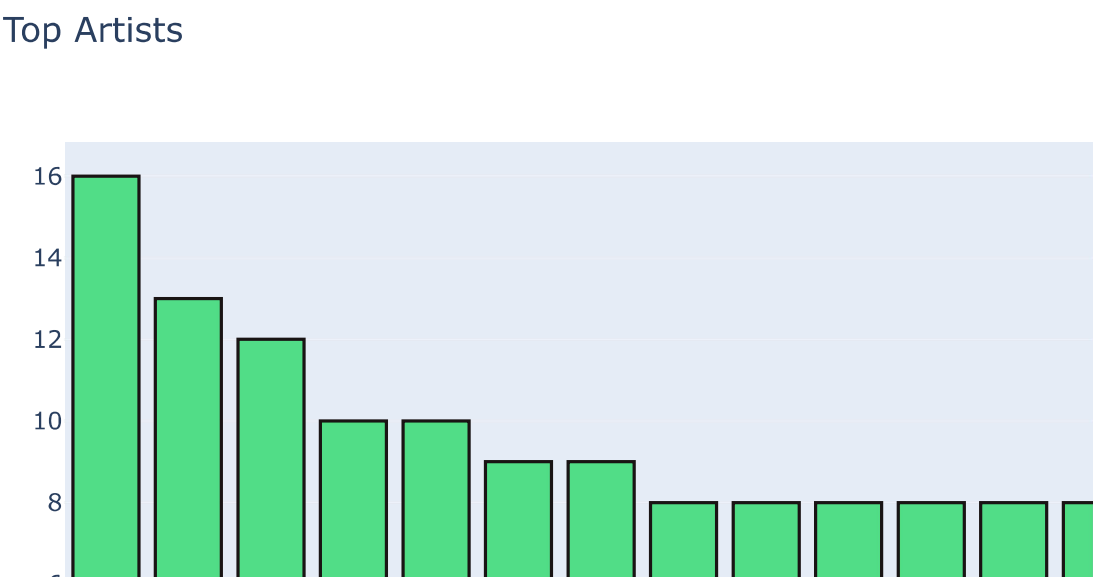
In [36]:
```
df['target'].value_counts()
```

Out[36]:
```
target
1    1020
0     997
Name: count, dtype: int64
```

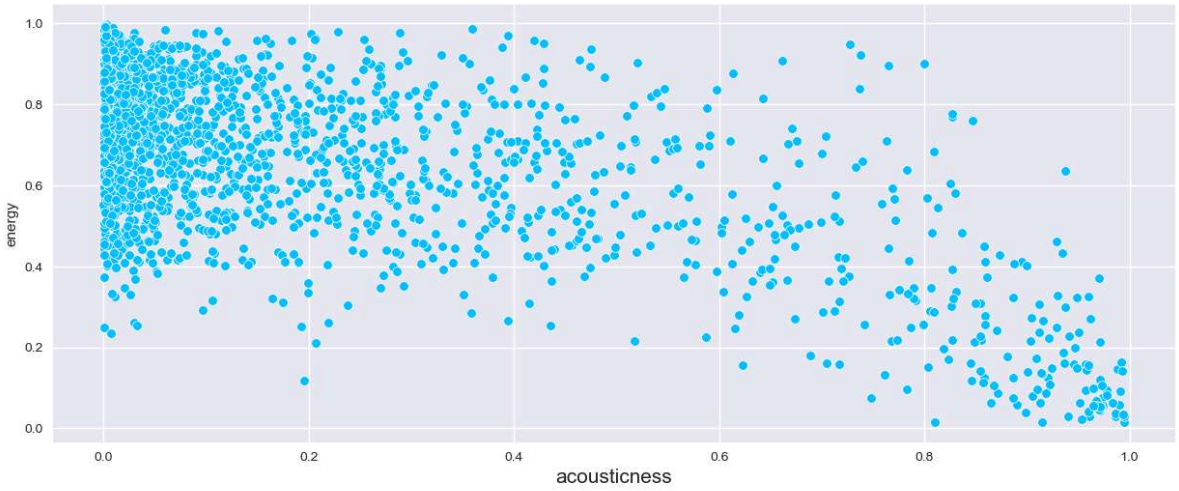
In [37]:
```
labels=['liked songs', 'disliked songs']
values=df['target'].value_counts().tolist()

px.pie(df, values=values, names=labels, title='Liked vs Disliked Analysis')
```

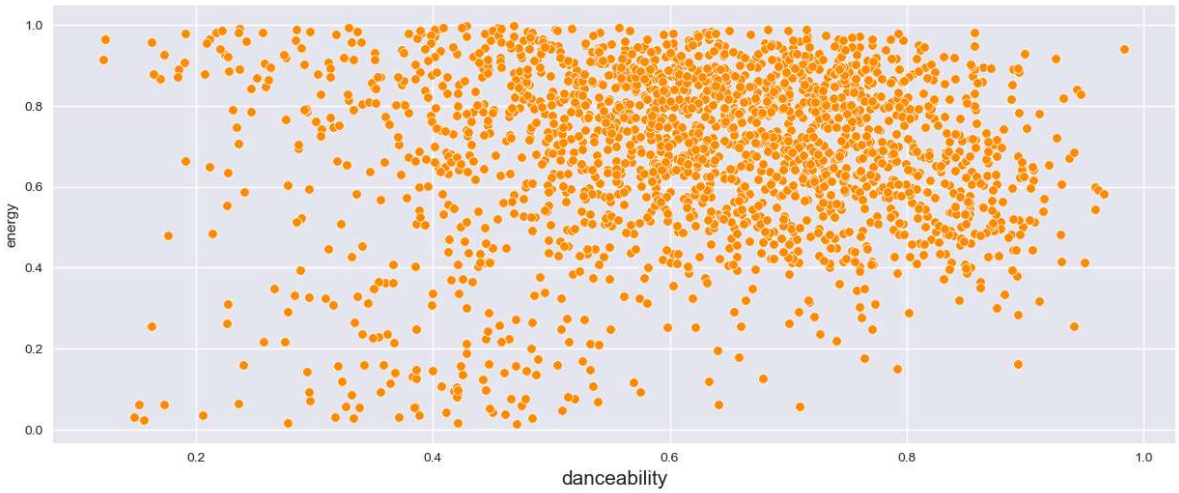


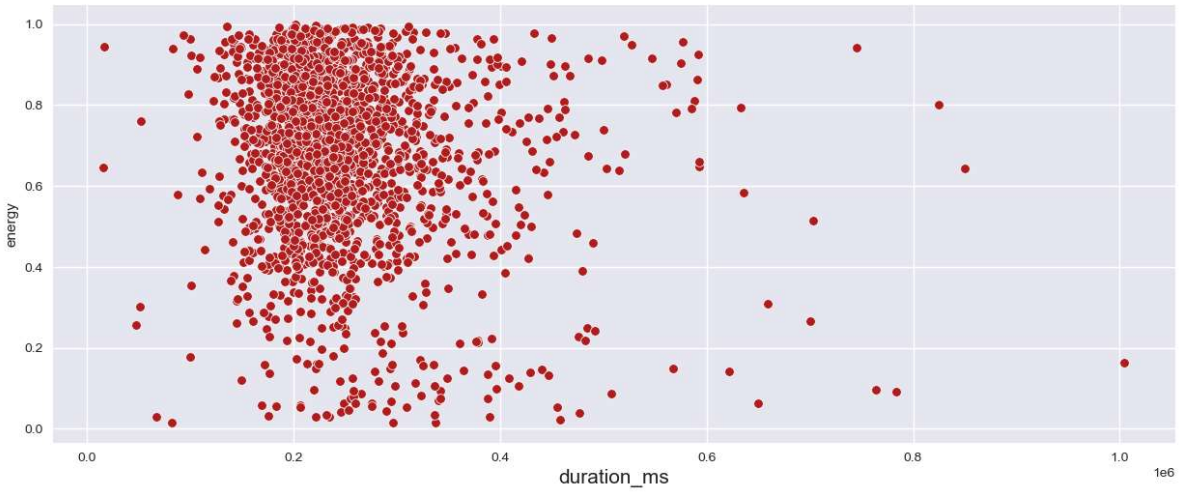
In [38]:
```
values = df['artist'].value_counts().tolist()[:20]
names = list(dict(df['artist'].value_counts()).keys())[:20]

fig = go.Bar(x = names,
             y = values,
             marker = dict(color = 'rgba(30, 215, 96, 0.75)',
                           line=dict(color='rgb(25, 20, 20)',width=1.5)))

layout = go.Layout()


fig = go.Figure(data = fig, layout = layout)
fig.update_layout(title_text='Top Artists')
fig.show()
```



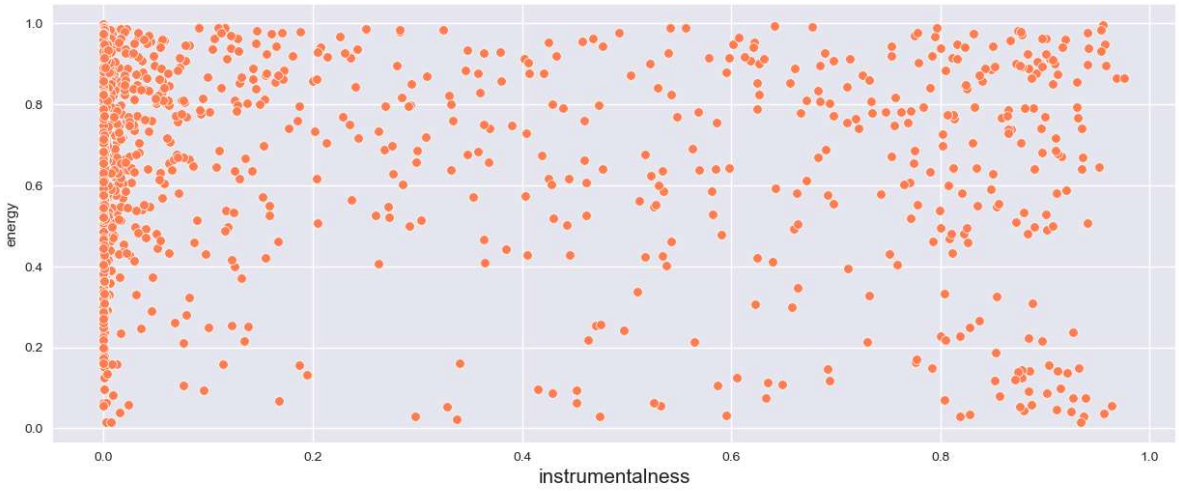
In [39]:
```
plt.style.use("seaborn")
fig, ax =plt.subplots(3,1, figsize=(15,20))

sn.scatterplot(x = df['acousticness'], y = df['energy'], ax=ax[0], color="deep
ax[0].set_xlabel("acousticness",fontsize=15);

sn.scatterplot(x = df['danceability'], y = df['energy'], ax=ax[1], color="dark
ax[1].set_xlabel("danceability",fontsize=15);

sn.scatterplot(x = df['duration_ms'], y = df['energy'], ax=ax[2], color="fireb
ax[2].set_xlabel("duration_ms",fontsize=15);
```

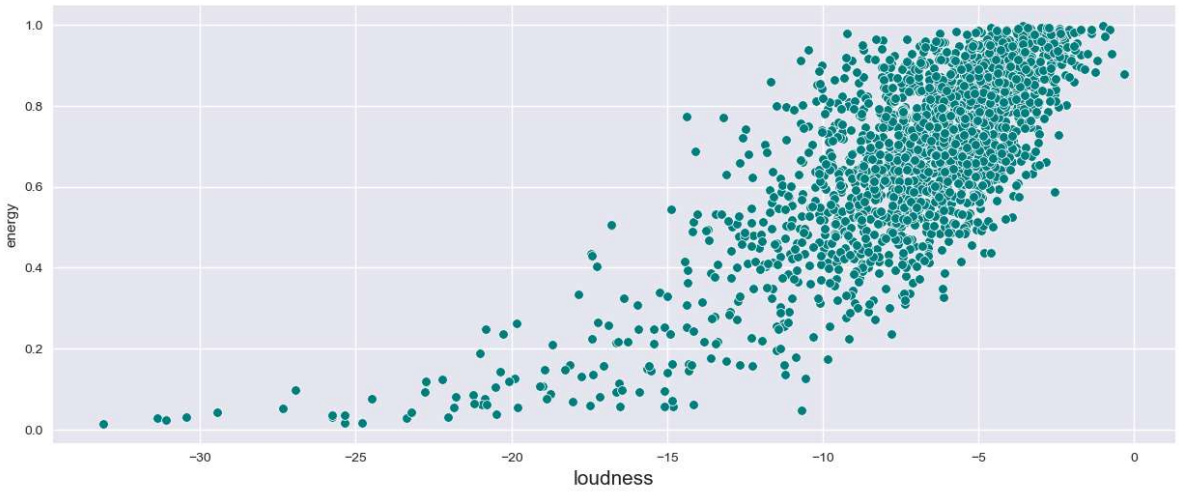

energy
acousticness
0.0
0.2
0.4
0.6
0.8
1.0

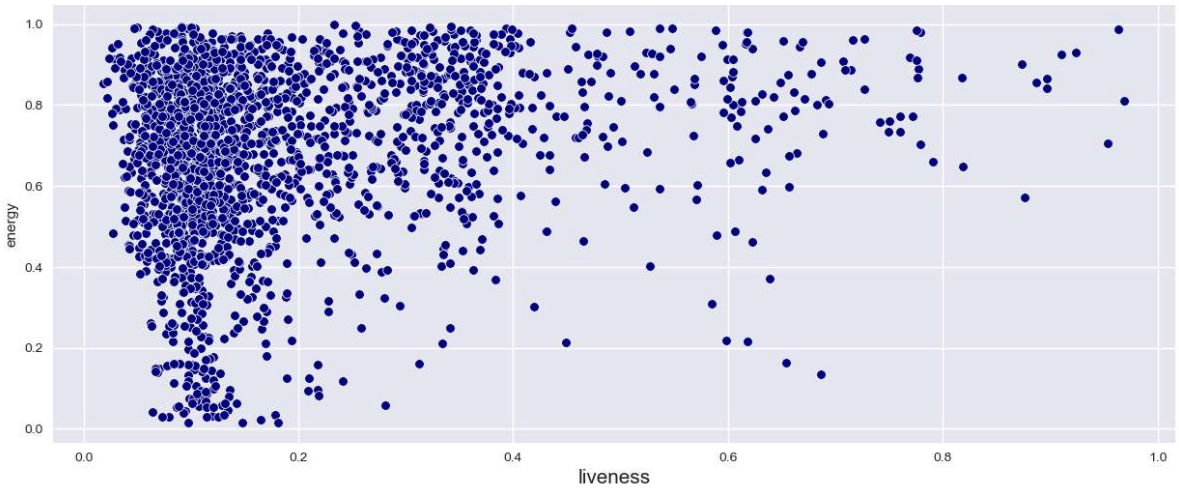

energy
danceability
0.0
0.2
0.4
0.6
0.8
1.0


energy
duration_ms
1e6
0.0
0.2
0.4
0.6
0.8
1.0

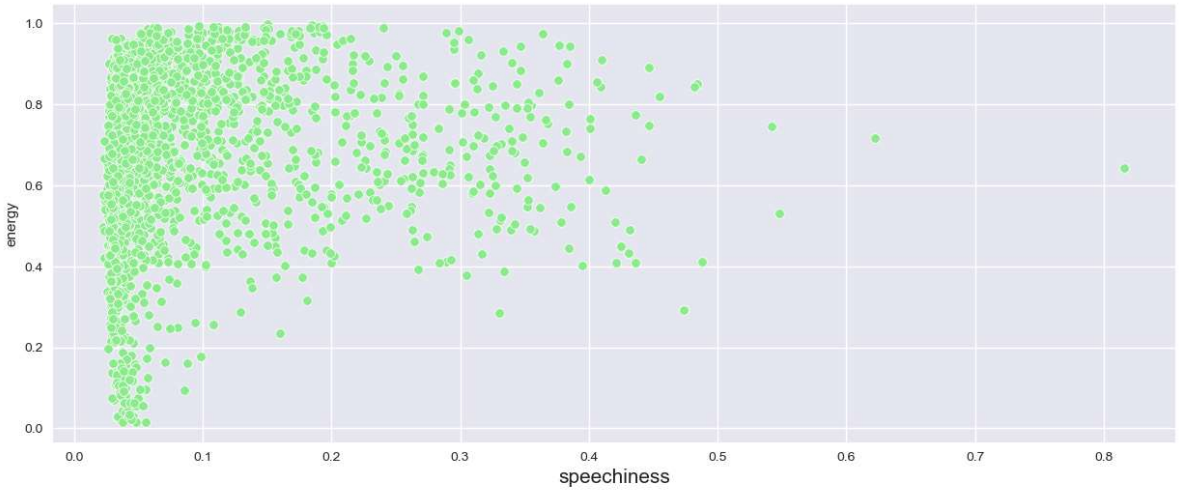
In [40]:
```
plt.style.use("seaborn")
fig, ax =plt.subplots(3,1, figsize=(15,20))

sn.scatterplot(x = df['instrumentalness'], y = df['energy'], ax=ax[0], color="
ax[0].set_xlabel("instrumentalness",fontsize=15);

sn.scatterplot(x = df['loudness'], y = df['energy'], ax=ax[1], color="teal");
ax[1].set_xlabel("loudness",fontsize=15);

sn.scatterplot(x = df['liveness'], y = df['energy'], ax=ax[2], color="navy");
ax[2].set_xlabel("liveness",fontsize=15);
```

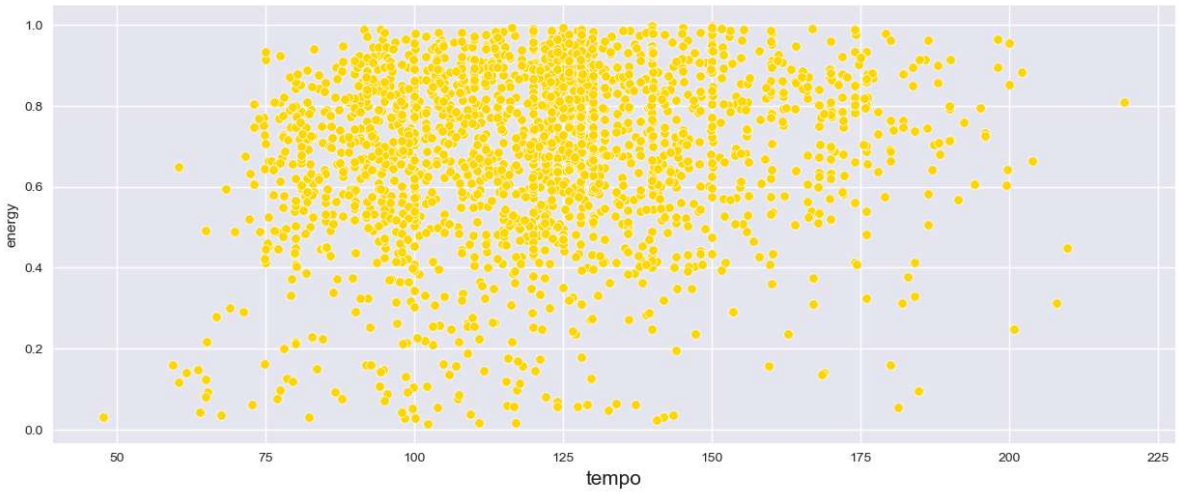

energy
instrumentalness

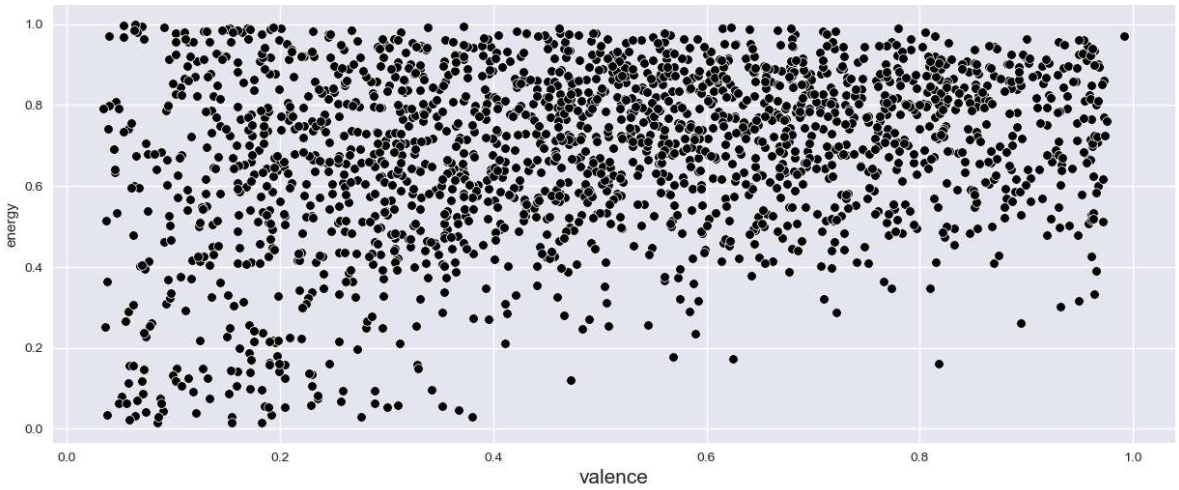

energy
loudness


energy
liveness

In [41]:
```
plt.style.use("seaborn")
fig, ax =plt.subplots(3,1, figsize=(15,20))

sn.scatterplot(x = df['speechiness'], y = df['energy'], ax=ax[0], color="light
ax[0].set_xlabel("speechiness",fontsize=15);

sn.scatterplot(x = df['tempo'], y = df['energy'], ax=ax[1], color="gold");
ax[1].set_xlabel("tempo",fontsize=15);

sn.scatterplot(x = df['valence'], y = df['energy'], ax=ax[2], color="black");
ax[2].set_xlabel("valence",fontsize=15);
```


energy
speechiness


energy
tempo


energy
valence