In [1]:
```
import pandas as pd
import numpy as np
import warnings
warnings.filterwarnings("ignore")
import seaborn as sns
```

In [2]:
```
df=pd.read_csv('spotify_songs.csv')
df
```

Out[2]:

	track_id	track_name	track_artist	track_popularity	track_album_id	track_alb
0	6f807x0ima9a1j3VPbc7VN	I Don't Care (with Justin Bieber) - Loud Luxur...	Ed Sheeran	66	2oCs0DGTsRO98Gh5ZSl2Cx	I Don't Justin Bie
1	0r7CVbZTWZgbTCYdfa2P31	Memories - Dillon Francis Remix	Maroon 5	67	63rPSO264uRjW1X5E6cWv6	Memor Fran
2	1z1Hg7Vb0AhHDiEmnDE79l	All the Time - Don Diablo Remix	Zara Larsson	70	1HoSmj2eLcsrR0vE9gThr4	All the Diab
3	75FpbthrwQmzHlBJLuGdC7	Call You Mine - Keanu Silva Remix	The Chainsmokers	60	1nqYsOef1yKKuGOVchbsk6	Call You
4	1e8PAfcKUYoKkxPhrHqw4x	Someone You Loved - Future Humans Remix	Lewis Capaldi	69	7m7vv9wlQ4i0LFuJiE2zsQ	Som Lov Huma
...	...	...	...	...	...	
32828	7bxnKAamR3snQ1VGLuVfC1	City Of Lights - Official Radio Edit	Lush & Simon	42	2azRoBBWEEEYhqV6sb7JrT	City Of Lig
32829	5Aevni09Em4575077nkWHz	Closer - Sultan & Ned Shepard Remix	Tegan and Sara	20	6kD6KLxj7s8eCE3ABvAyf5	Close
32830	7ImMqPP3Q1yfUHvsdn7wEo	Sweet Surrender - Radio Edit	Starkillers	14	0ltWNSY9JgxoIZO4VzuCa6	Sweet (F
32831	2m69mhnfQ1Oq6lGtXuYhgX	Only For You - Maor Levi Remix	Mat Zo	15	1fGrOkHnHJcStl14zNx8Jy	On
32832	29zWqhca3zt5NsckZqDf6c	Typhoon - Original Mix	Julian Calor	27	0X3mUOm6MhxR7PzxG95rAo	Typh

32833 rows × 23 columns

In [3]:
```
df.isnull().sum()
```

Out[3]:
```
track_id                    0
track_name                  5
track_artist                5
track_popularity            0
track_album_id              0
track_album_name            5
track_album_release_date    0
playlist_name               0
playlist_id                 0
playlist_genre              0
playlist_subgenre           0
danceability                0
energy                      0
key                         0
loudness                    0
mode                        0
speechiness                 0
acousticness                0
instrumentalness            0
liveness                    0
valence                     0
tempo                       0
duration_ms                 0
dtype: int64
```

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

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 32833 entries, 0 to 32832
Data columns (total 23 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   track_id                  32833 non-null  object
 1   track_name                32828 non-null  object
 2   track_artist              32828 non-null  object
 3   track_popularity          32833 non-null  int64
 4   track_album_id            32833 non-null  object
 5   track_album_name          32828 non-null  object
 6   track_album_release_date  32833 non-null  object
 7   playlist_name             32833 non-null  object
 8   playlist_id               32833 non-null  object
 9   playlist_genre            32833 non-null  object
 10  playlist_subgenre         32833 non-null  object
 11  danceability              32833 non-null  float64
 12  energy                    32833 non-null  float64
 13  key                       32833 non-null  int64
 14  loudness                  32833 non-null  float64
 15  mode                      32833 non-null  int64
 16  speechiness               32833 non-null  float64
 17  acousticness              32833 non-null  float64
 18  instrumentalness          32833 non-null  float64
 19  liveness                  32833 non-null  float64
 20  valence                   32833 non-null  float64
 21  tempo                     32833 non-null  float64
 22  duration_ms               32833 non-null  int64
dtypes: float64(9), int64(4), object(10)
memory usage: 5.8+ MB
```

In [5]:
```
df.drop(['track_id','track_name','track_artist','track_album_id','track_album_name',
                          'playlist_id','playlist_name','playlist_id','playlist_subgenre'],inpla
```

In [6]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 32833 entries, 0 to 32832
Data columns (total 15 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   track_popularity          32833 non-null  int64
 1   track_album_release_date  32833 non-null  object
 2   playlist_genre            32833 non-null  object
 3   danceability              32833 non-null  float64
 4   energy                    32833 non-null  float64
 5   key                       32833 non-null  int64
 6   loudness                  32833 non-null  float64
 7   mode                      32833 non-null  int64
 8   speechiness               32833 non-null  float64
 9   acousticness              32833 non-null  float64
 10  instrumentalness          32833 non-null  float64
 11  liveness                  32833 non-null  float64
 12  valence                   32833 non-null  float64
 13  tempo                     32833 non-null  float64
 14  duration_ms               32833 non-null  int64
dtypes: float64(9), int64(4), object(2)
memory usage: 3.8+ MB
```

In [7]:
```
df.groupby('playlist_genre').sum()
```

Out[7]:

playlist_genre	track_popularity	track_album_release_date	danceability	energy	key	loudness	mode	speecl
edm	210499	2013-09-162015-10-232018-08-242016-12-262019-0...	3958.4120	4849.362000	32343	-32798.051	3143	52
latin	242422	2017-11-242019-03-082017-07-212019-10-182019-0...	3676.9958	3651.350775	28270	-32293.264	2897	52
pop	262931	2019-06-142019-12-132019-07-052019-07-192019-0...	3520.6345	3860.560340	29291	-34778.512	3239	40
r&b	223885	2020-01-102019-11-132019-12-272018-06-032019-1...	3639.7440	3209.364400	29330	-42713.989	2832	63
rap	248316	2020-01-102019-12-062020-01-172019-12-122020-0...	4127.6550	3738.970500	31436	-40464.880	2996	113
rock	206597	1987-01-012018-05-041987-01-011987-01-011987-0...	2577.2330	3628.158500	25790	-37572.619	3467	28

In [ ]:

In [8]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 32833 entries, 0 to 32832
Data columns (total 15 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   track_popularity          32833 non-null  int64
 1   track_album_release_date  32833 non-null  object
 2   playlist_genre            32833 non-null  object
 3   danceability              32833 non-null  float64
 4   energy                    32833 non-null  float64
 5   key                       32833 non-null  int64
 6   loudness                  32833 non-null  float64
 7   mode                      32833 non-null  int64
 8   speechiness               32833 non-null  float64
 9   acousticness              32833 non-null  float64
 10  instrumentalness          32833 non-null  float64
 11  liveness                  32833 non-null  float64
 12  valence                   32833 non-null  float64
 13  tempo                     32833 non-null  float64
 14  duration_ms               32833 non-null  int64
dtypes: float64(9), int64(4), object(2)
memory usage: 3.8+ MB
```

In [9]:
```
df.groupby(['track_album_release_date']).sum()
```

Out[9]:

track_album_release_date	track_popularity	playlist_genre	danceability	energy	ke
1957-01-01	59	r&b	0.565	0.962	
1957-03	1	rock	0.474	0.598	
1958-03-21	73	rock	0.647	0.582	1
1960	64	r&br&br&br&b	1.545	1.859	2
1961-10-26	47	r&b	0.585	0.036	
...	...	...	...	...	.
2020-01-15	918	poppoppoppoppoppoppoppoppoprapraprapraprapraprapr...	18.724	19.891	17
2020-01-16	1132	poppoppoppoprapraprapraprapraprapraplatinlatinlat...	15.298	13.928	10
2020-01-17	4736	poppoppoppoppoppoppoppoppoprapraprapraprapraprapr...	86.589	93.460	70
2020-01-20	82	popedm	1.416	1.324	1
2020-01-29	29	pop	0.695	0.880	

4530 rows × 14 columns

In [10]:
```
def extract_year(date_str):
    try:
        return pd.to_datetime(date_str, format='%Y-%m-%d', errors='coerce').year
    except:
        return pd.to_numeric(date_str, errors='coerce')

df['track_album_release_date'] = df['track_album_release_date'].apply(extract_year)
```

In [11]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 32833 entries, 0 to 32832
Data columns (total 15 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   track_popularity          32833 non-null  int64
 1   track_album_release_date  30947 non-null  float64
 2   playlist_genre            32833 non-null  object
 3   danceability              32833 non-null  float64
 4   energy                    32833 non-null  float64
 5   key                       32833 non-null  int64
 6   loudness                  32833 non-null  float64
 7   mode                      32833 non-null  int64
 8   speechiness               32833 non-null  float64
 9   acousticness              32833 non-null  float64
 10  instrumentalness          32833 non-null  float64
 11  liveness                  32833 non-null  float64
 12  valence                   32833 non-null  float64
 13  tempo                     32833 non-null  float64
 14  duration_ms               32833 non-null  int64
dtypes: float64(10), int64(4), object(1)
memory usage: 3.8+ MB
```

In [12]:
```
df.head(5)
```

Out[12]:

	track_popularity	track_album_release_date	playlist_genre	danceability	energy	key	loudness	mode	speechiness
0	66	2019.0	pop	0.748	0.916	6	-2.634	1	0.0583
1	67	2019.0	pop	0.726	0.815	11	-4.969	1	0.0373
2	70	2019.0	pop	0.675	0.931	1	-3.432	0	0.0742
3	60	2019.0	pop	0.718	0.930	7	-3.778	1	0.1020
4	69	2019.0	pop	0.650	0.833	1	-4.672	1	0.0359

In [13]:
```
df.tail(5)
```

Out[13]:

	track_popularity	track_album_release_date	playlist_genre	danceability	energy	key	loudness	mode	speechi
32828	42	2014.0	edm	0.428	0.922	2	-1.814	1	0.
32829	20	2013.0	edm	0.522	0.786	0	-4.462	1	0.
32830	14	2014.0	edm	0.529	0.821	6	-4.899	0	0.
32831	15	2014.0	edm	0.626	0.888	2	-3.361	1	0.
32832	27	2014.0	edm	0.603	0.884	5	-4.571	0	0.

In [14]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 32833 entries, 0 to 32832
Data columns (total 15 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   track_popularity          32833 non-null  int64
 1   track_album_release_date  30947 non-null  float64
 2   playlist_genre            32833 non-null  object
 3   danceability              32833 non-null  float64
 4   energy                    32833 non-null  float64
 5   key                       32833 non-null  int64
 6   loudness                  32833 non-null  float64
 7   mode                      32833 non-null  int64
 8   speechiness               32833 non-null  float64
 9   acousticness              32833 non-null  float64
 10  instrumentalness          32833 non-null  float64
 11  liveness                  32833 non-null  float64
 12  valence                   32833 non-null  float64
 13  tempo                     32833 non-null  float64
 14  duration_ms               32833 non-null  int64
dtypes: float64(10), int64(4), object(1)
memory usage: 3.8+ MB
```

In [15]:
```
df.isnull().sum()
```

Out[15]:
```
track_popularity              0
track_album_release_date    1886
playlist_genre                0
danceability                  0
energy                        0
key                           0
loudness                      0
mode                          0
speechiness                   0
acousticness                  0
instrumentalness              0
liveness                      0
valence                       0
tempo                         0
duration_ms                   0
dtype: int64
```

In [16]:
```
df.min()
```

Out[16]:
```
track_popularity                   0
track_album_release_date      1957.0
playlist_genre                   edm
danceability                     0.0
energy                      0.000175
key                                0
loudness                     -46.448
mode                               0
speechiness                      0.0
acousticness                     0.0
instrumentalness                 0.0
liveness                         0.0
valence                          0.0
tempo                            0.0
duration_ms                     4000
dtype: object
```

In [17]:
```
df.max()
```

Out[17]:
```
track_popularity              100
track_album_release_date    2020.0
playlist_genre               rock
danceability                0.983
energy                        1.0
key                            11
loudness                    1.275
mode                            1
speechiness                 0.918
acousticness                0.994
instrumentalness            0.994
liveness                    0.996
valence                     0.991
tempo                      239.44
duration_ms                517810
dtype: object
```

In [ ]:
```

```

In [18]:
```
df['track_album_release_date'] = df['track_album_release_date'].fillna(2012)
```

In [19]:
```
df.isnull().sum()
```

Out[19]:
```
track_popularity            0
track_album_release_date    0
playlist_genre              0
danceability                0
energy                      0
key                         0
loudness                    0
mode                        0
speechiness                 0
acousticness                0
instrumentalness            0
liveness                    0
valence                     0
tempo                       0
duration_ms                 0
dtype: int64
```

In [20]:
```
df.head(5)
```

Out[20]:

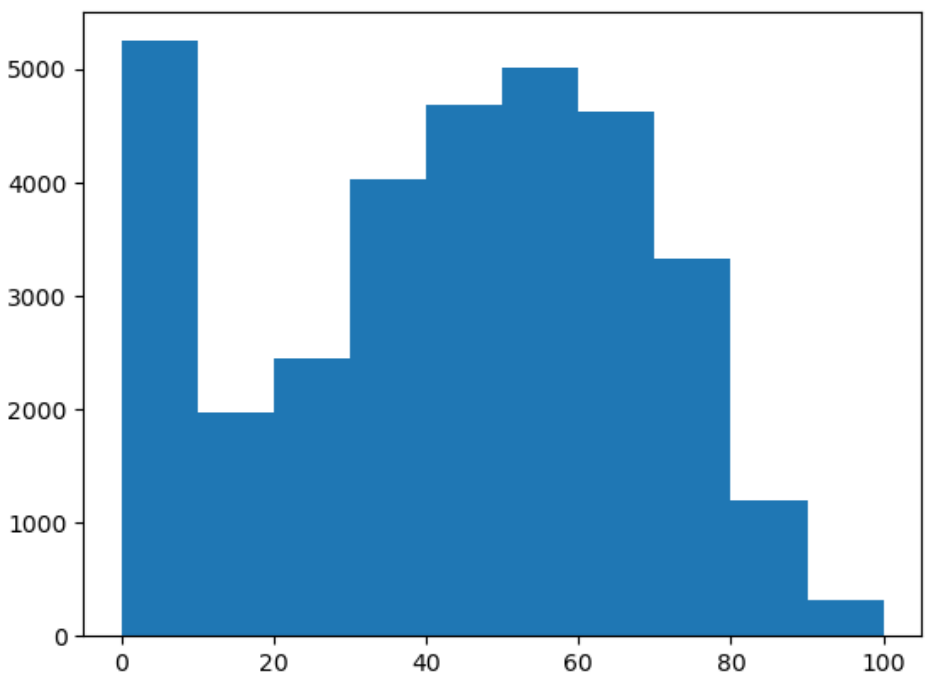
	track_popularity	track_album_release_date	playlist_genre	danceability	energy	key	loudness	mode	speechiness
0	66	2019.0	pop	0.748	0.916	6	-2.634	1	0.0583
1	67	2019.0	pop	0.726	0.815	11	-4.969	1	0.0373
2	70	2019.0	pop	0.675	0.931	1	-3.432	0	0.0742
3	60	2019.0	pop	0.718	0.930	7	-3.778	1	0.1020
4	69	2019.0	pop	0.650	0.833	1	-4.672	1	0.0359

In [21]:
```
list(df)
```

Out[21]:
```
['track_popularity',
 'track_album_release_date',
 'playlist_genre',
 'danceability',
 'energy',
 'key',
 'loudness',
 'mode',
 'speechiness',
 'acousticness',
 'instrumentalness',
 'liveness',
 'valence',
 'tempo',
 'duration_ms']
```

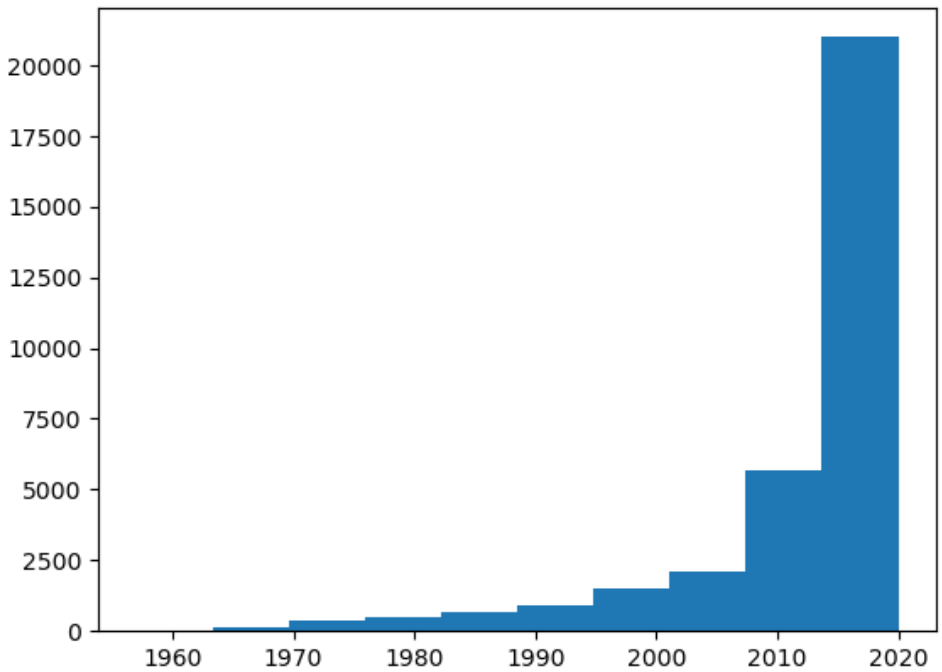
In [22]:
```
import matplotlib.pyplot as plt
plt.hist(df['track_popularity'])
```

Out[22]:
```
(array([5243., 1977., 2443., 4025., 4685., 5004., 4622., 3322., 1201.,
        311.]),
 array([  0.,  10.,  20.,  30.,  40.,  50.,  60.,  70.,  80.,  90., 100.]),
 <BarContainer object of 10 artists>)
```



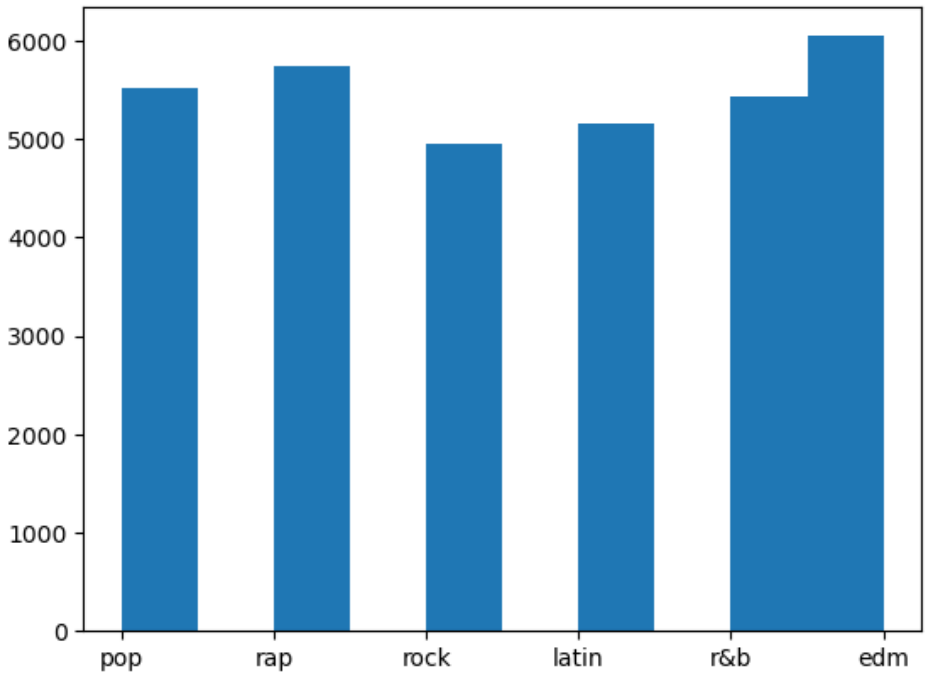
In [23]:
```
import matplotlib.pyplot as plt
plt.hist(df['track_album_release_date'])
```

Out[23]:
```
(array([7.0000e+00, 1.2600e+02, 3.5900e+02, 4.7100e+02, 6.7300e+02,
        9.2600e+02, 1.5120e+03, 2.0980e+03, 5.6790e+03, 2.0982e+04]),
 array([1957. , 1963.3, 1969.6, 1975.9, 1982.2, 1988.5, 1994.8, 2001.1,
        2007.4, 2013.7, 2020. ]),
 <BarContainer object of 10 artists>)
```



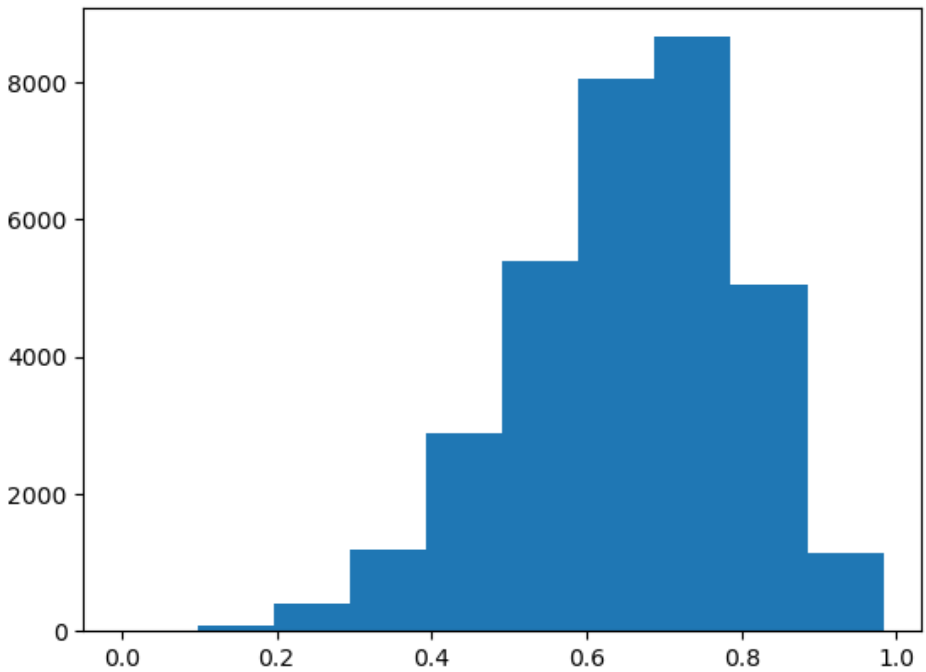
In [24]:
```
import matplotlib.pyplot as plt
plt.hist(df['playlist_genre'])
```

Out[24]:
```
(array([5507.,    0., 5746.,    0., 4951.,    0., 5155.,    0., 5431.,
        6043.]),
 array([0. , 0.5, 1. , 1.5, 2. , 2.5, 3. , 3.5, 4. , 4.5, 5. ]),
 <BarContainer object of 10 artists>)
```



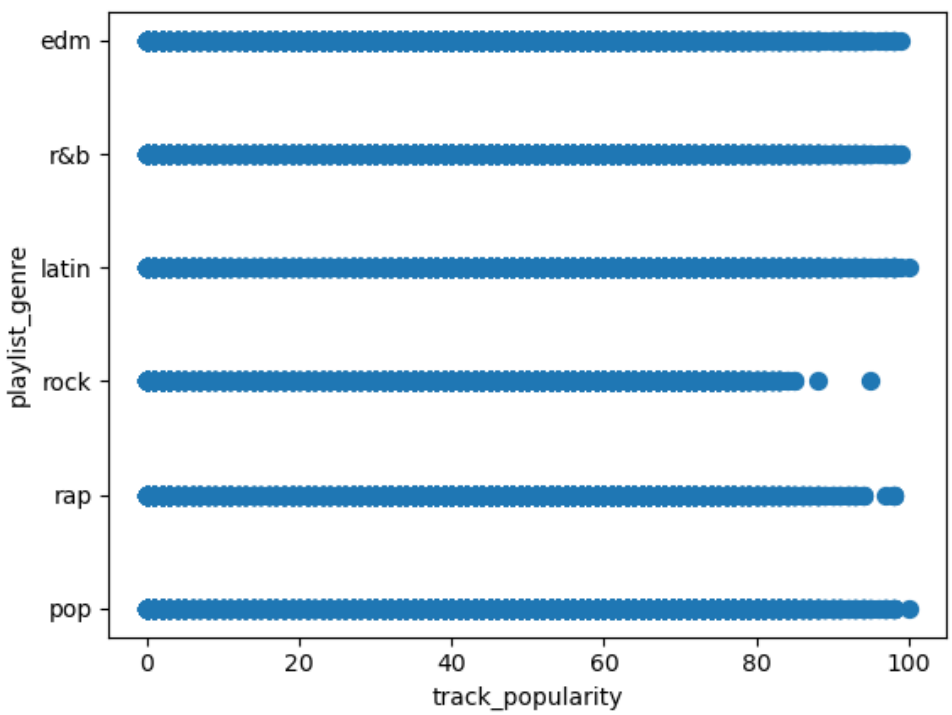
In [25]:
```
import matplotlib.pyplot as plt
plt.hist(df['danceability'])
```

Out[25]:
```
(array([3.000e+00, 6.900e+01, 4.110e+02, 1.193e+03, 2.894e+03, 5.398e+03,
        8.034e+03, 8.651e+03, 5.049e+03, 1.131e+03]),
 array([0.    , 0.0983, 0.1966, 0.2949, 0.3932, 0.4915, 0.5898, 0.6881,
        0.7864, 0.8847, 0.983 ]),
 <BarContainer object of 10 artists>)
```



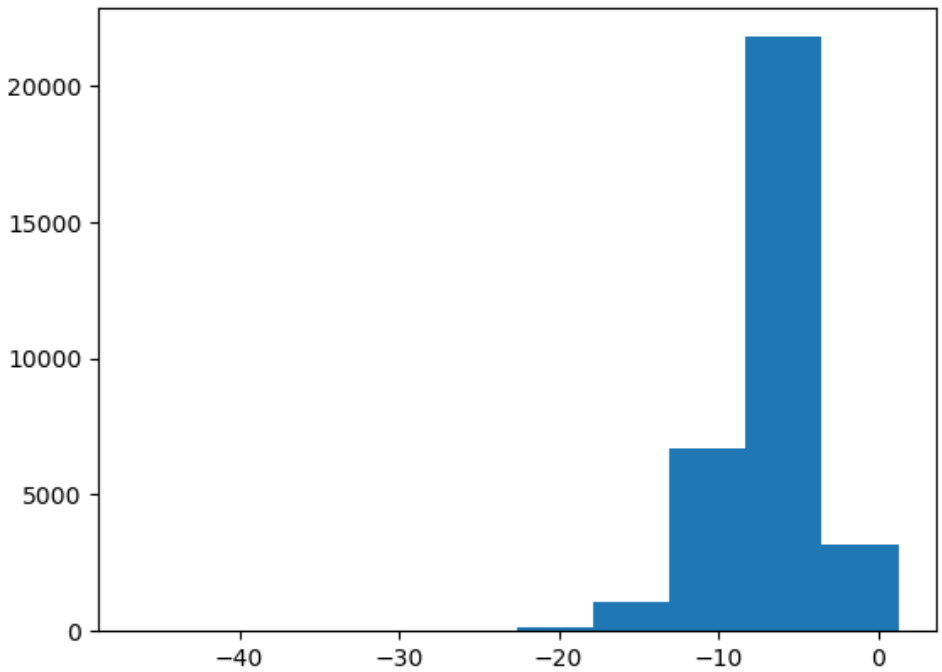
In [26]:
```
import matplotlib.pyplot as plt
df.plot.scatter(x='track_popularity',y='playlist_genre',s=50)
```

Out[26]:
```
<Axes: xlabel='track_popularity', ylabel='playlist_genre'>
```



In [27]:
```
import matplotlib.pyplot as plt
plt.hist(df['loudness'])
```

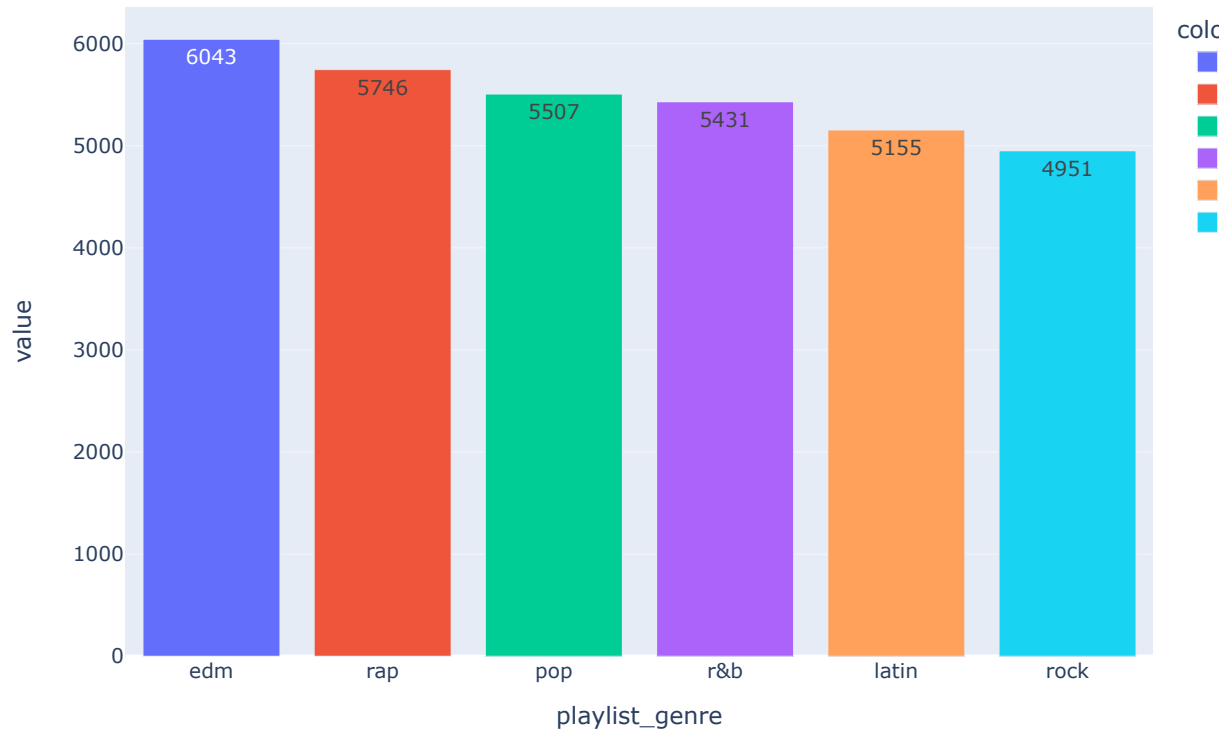
Out[27]:
```
(array([1.000e+00, 0.000e+00, 5.000e+00, 2.000e+00, 2.100e+01, 1.170e+02,
        1.087e+03, 6.677e+03, 2.178e+04, 3.143e+03]),
 array([-46.448 , -41.6757, -36.9034, -32.1311, -27.3588, -22.5865,
        -17.8142, -13.0419,  -8.2696,  -3.4973,   1.275 ]),
 <BarContainer object of 10 artists>)
```



In [28]:
```
df.playlist_genre.value_counts()
```

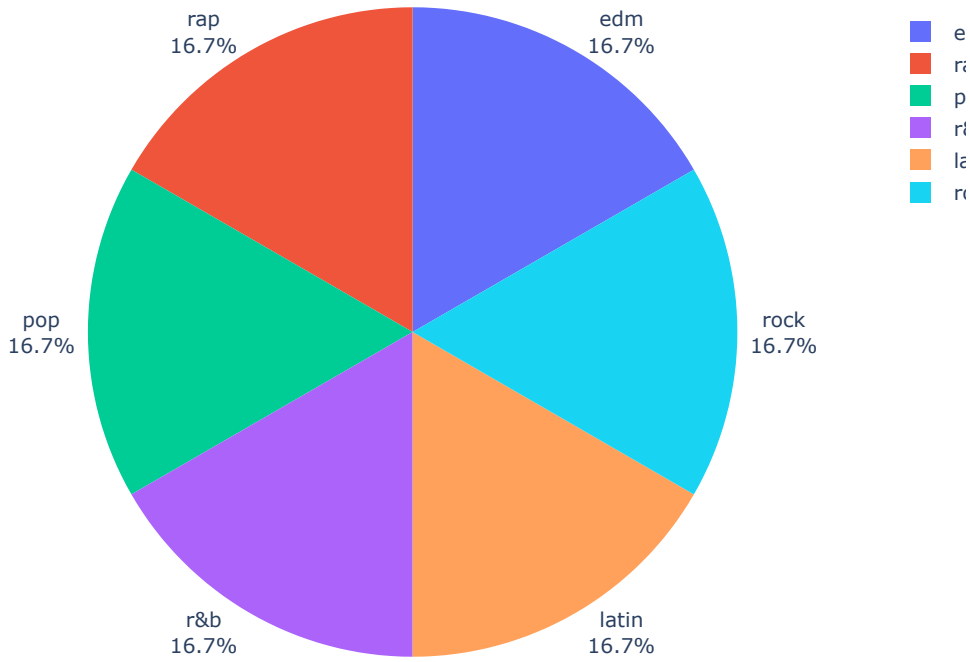
Out[28]:
```
playlist_genre
edm      6043
rap      5746
pop      5507
r&b      5431
latin    5155
rock     4951
Name: count, dtype: int64
```

In [29]:
```
import plotly.express as px
fig=px.bar(df.playlist_genre.value_counts(),color=['red','blue','green','orange','teal','black'
fig.update_layout(width = 800)
fig.show()
```



In [30]:

```
import plotly.express as px
fig=px.pie(df.playlist_genre.value_counts().values,df.playlist_genre.value_counts().index,color
fig.update_layout(width = 800)
fig.update_traces(textposition ='outside', textinfo = 'percent+label')
fig.show()
```



In [31]:

```
df.groupby('mode').sum()
```

Out[31]:

	track_popularity	track_album_release_date	playlist_genre	danceability
mode				
0	601356	28705085.0	poppoppoppoppoppoppoppoppoppoppoppoppoppoppoppopp...	9475.9716
1	793294	37361117.0	poppoppoppoppoppoppoppoppoppoppoppoppoppoppoppopp...	12024.7027

In [32]:
```
df.groupby('playlist_genre').sum()
```

Out[32]:

playlist_genre	track_popularity	track_album_release_date	danceability	energy	key	loudness	mode	speecl
edm	210499	12187047.0	3958.4120	4849.362000	32343	-32798.051	3143	52
latin	242422	10387623.0	3676.9958	3651.350775	28270	-32293.264	2897	52
pop	262931	11095162.0	3520.6345	3860.560340	29291	-34778.512	3239	40
r&b	223885	10918823.0	3639.7440	3209.364400	29330	-42713.989	2832	63
rap	248316	11568673.0	4127.6550	3738.970500	31436	-40464.880	2996	113
rock	206597	9908874.0	2577.2330	3628.158500	25790	-37572.619	3467	28

In [33]:
```
df.groupby('key').sum()
```

Out[33]:

key	track_popularity	track_album_release_date	playlist_genre	danceability	
0	148823	6948817.0	poppoppoppoppoppoppoppoppoppoppoppoppoppoppoppoppopp...	2240.3720	23
1	172406	8072418.0	poppoppoppoppoppoppoppoppoppoppoppoppoppoppoppoppopp...	2733.3045	28
2	117551	5685082.0	poppoppoppoppoppoppoppoppoppoppoppoppoppoppoppoppopp...	1804.2540	19
3	37798	1837775.0	poppoppoppoppoppoppoppoppoppoppoppoppoppoppoppoppopp...	565.7511	6
4	92504	4426188.0	poppoppoppoppoppoppoppoppoppoppoppoppoppoppoppoppopp...	1389.4380	15
5	113896	5394182.0	poppoppoppoppoppoppoppoppoppoppoppoppoppoppoppoppopp...	1744.2180	18
6	113693	5374194.0	poppoppoppoppoppoppoppoppoppoppoppoppoppoppoppoppopp...	1767.2700	18
7	137112	6743752.0	poppoppoppoppoppoppoppoppoppoppoppoppoppoppoppoppopp...	2188.8507	23
8	108323	4892192.0	poppoppoppoppoppoppoppoppoppoppoppoppoppoppoppoppopp...	1606.2380	17
9	127843	6086585.0	poppoppoppoppoppoppoppoppoppoppoppoppoppoppoppoppopp...	1942.0910	20
10	97643	4574948.0	poppoppoppoppoppoppoppoppoppoppoppoppoppoppoppoppopp...	1523.3030	15
11	127058	6030069.0	poppoppoppoppoppoppoppoppoppoppoppoppoppoppoppoppopp...	1995.5840	21

In [34]:
```
df.groupby('track_album_release_date').sum()
```

Out[34]:

	track_popularity	playlist_genre	danceability	en
track_album_release_date				
1957.0	59	r&b	0.5650	0.96
1958.0	73	rock	0.6470	0.58
1961.0	47	r&b	0.5850	0.03
1963.0	145	rockrockrock	2.0460	2.05
1964.0	342	rockrockrockrockrockrockrock	4.5610	6.34
...	...	...	...	
2016.0	81346	poppoppoppoppoppoppoppoppoppoppoppoppoppoppoppopp...	1374.9870	1471.47
2017.0	101379	poppoppoppoppoppoppoppoppoppoppoppoppoppoppoppopp...	1608.8787	1656.22
2018.0	151950	poppoppoppoppoppoppoppoppoppoppoppoppoppoppoppopp...	2248.5375	2234.90
2019.0	466737	poppoppoppoppoppoppoppoppoppoppoppoppoppoppoppopp...	6150.8681	6325.26
2020.0	36472	poppoppoppoppoppoppoppoppoppoppoppoppoppoppoppopp...	527.5090	526.21

61 rows × 14 columns

In [35]:
```
df['mode']=df['mode'].map({1:'high',0:'low'})
```

In [36]:
```
df=pd.get_dummies(df,dtype=int)
df
```

Out[36]:

	track_popularity	track_album_release_date	danceability	energy	key	loudness	speechiness	acousticness	ir
0	66	2019.0	0.748	0.916	6	-2.634	0.0583	0.102000	
1	67	2019.0	0.726	0.815	11	-4.969	0.0373	0.072400	
2	70	2019.0	0.675	0.931	1	-3.432	0.0742	0.079400	
3	60	2019.0	0.718	0.930	7	-3.778	0.1020	0.028700	
4	69	2019.0	0.650	0.833	1	-4.672	0.0359	0.080300	
...	...	...	...	...	...	...	...	...	
32828	42	2014.0	0.428	0.922	2	-1.814	0.0936	0.076600	
32829	20	2013.0	0.522	0.786	0	-4.462	0.0420	0.001710	
32830	14	2014.0	0.529	0.821	6	-4.899	0.0481	0.108000	
32831	15	2014.0	0.626	0.888	2	-3.361	0.1090	0.007920	
32832	27	2014.0	0.603	0.884	5	-4.571	0.0385	0.000133	

32833 rows × 21 columns

In [37]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 32833 entries, 0 to 32832
Data columns (total 21 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   track_popularity          32833 non-null  int64
 1   track_album_release_date  32833 non-null  float64
 2   danceability              32833 non-null  float64
 3   energy                    32833 non-null  float64
 4   key                       32833 non-null  int64
 5   loudness                  32833 non-null  float64
 6   speechiness               32833 non-null  float64
 7   acousticness              32833 non-null  float64
 8   instrumentalness          32833 non-null  float64
 9   liveness                  32833 non-null  float64
 10  valence                   32833 non-null  float64
 11  tempo                     32833 non-null  float64
 12  duration_ms               32833 non-null  int64
 13  playlist_genre_edm        32833 non-null  int32
 14  playlist_genre_latin      32833 non-null  int32
 15  playlist_genre_pop        32833 non-null  int32
 16  playlist_genre_r&b        32833 non-null  int32
 17  playlist_genre_rap        32833 non-null  int32
 18  playlist_genre_rock       32833 non-null  int32
 19  mode_high                 32833 non-null  int32
 20  mode_low                  32833 non-null  int32
dtypes: float64(10), int32(8), int64(3)
memory usage: 4.3 MB
```

In [38]:
```
y=df['playlist_genre_edm']#copied to check
X=df.drop('playlist_genre_edm',axis=1)#removed for model
```

In [39]:
```
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.32,random_state=42)
```

In [ ]:

In [40]:
```
X_test.head(5)
```

Out[40]:

	track_popularity	track_album_release_date	danceability	energy	key	loudness	speechiness	acousticness	in
30056	45	2019.0	0.520	0.789	0	-7.717	0.0432	0.004910	
11827	17	1978.0	0.651	0.661	9	-11.405	0.0511	0.265000	
23571	30	2015.0	0.640	0.758	10	-5.204	0.1600	0.665000	
14741	35	2020.0	0.398	0.966	4	-2.352	0.0453	0.000006	
25570	62	2015.0	0.447	0.625	10	-8.212	0.3230	0.035100	

In [41]:
```
X_train.shape
```

Out[41]: (22326, 20)

In [ ]:

In [42]:
```
y_train.shape
```

Out[42]:
```
(22326,)
```

In [43]:
```
y_train
```

Out[43]:
```
28688    1
28567    1
28837    1
6624     0
7072     0
        ..
16850    0
6265     0
11284    0
860      0
15795    0
Name: playlist_genre_edm, Length: 22326, dtype: int32
```

In [44]:
```
from sklearn.linear_model import LogisticRegression
reg=LogisticRegression()
reg.fit(X_train,y_train)
```

Out[44]:
```
▾ LogisticRegression
LogisticRegression()
```

In [45]:
```
y_pred=reg.predict(X_test)
y_pred
```

Out[45]:
```
array([0, 0, 0, ..., 0, 0, 0])
```

In [46]:
```
from sklearn.metrics import confusion_matrix
confusion_matrix(y_test,y_pred)
```

Out[46]:
```
array([[8559,    0],
       [1947,    1]], dtype=int64)
```

In [47]:
```
from sklearn.metrics import accuracy_score
accuracy_score(y_test,y_pred)
```

Out[47]:
```
0.8146949652612544
```

In [48]:
```
y=df['playlist_genre_latin']#copied to check
X=df.drop('playlist_genre_latin',axis=1)#removed for model
```

In [49]:
```
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.32,random_state=42)
from sklearn.linear_model import LogisticRegression
reg=LogisticRegression()
reg.fit(X_train,y_train)
y_pred=reg.predict(X_test)
y_pred
```

Out[49]:
```
array([0, 0, 0, ..., 0, 0, 0])
```

In [50]:
```
from sklearn.metrics import confusion_matrix
confusion_matrix(y_test,y_pred)
```

Out[50]:
```
array([[8889,    0],
       [1618,    0]], dtype=int64)
```

In [51]:
```
from sklearn.metrics import accuracy_score
accuracy_score(y_test,y_pred)
```

Out[51]:
```
0.846007423622347
```

In [52]:
```
y=df['playlist_genre_pop']#copied to check
X=df.drop('playlist_genre_pop',axis=1)#removed for model
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.32,random_state=42)
from sklearn.linear_model import LogisticRegression
reg=LogisticRegression()
reg.fit(X_train,y_train)
y_pred=reg.predict(X_test)
y_pred
```

Out[52]:
```
array([0, 0, 0, ..., 0, 0, 0])
```

In [53]:
```
from sklearn.metrics import confusion_matrix
confusion_matrix(y_test,y_pred)
```

Out[53]:
```
array([[8743,    0],
       [1764,    0]], dtype=int64)
```

In [54]:
```
from sklearn.metrics import accuracy_score
accuracy_score(y_test,y_pred)
```

Out[54]:
```
0.832111925383078
```

In [55]:
```
y=df['playlist_genre_r&b']#copied to check
X=df.drop('playlist_genre_r&b',axis=1)#removed for model
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.32,random_state=42)
from sklearn.linear_model import LogisticRegression
reg=LogisticRegression()
reg.fit(X_train,y_train)
y_pred=reg.predict(X_test)
y_pred
```

Out[55]:
```
array([0, 0, 0, ..., 0, 0, 0])
```

In [56]:
```
from sklearn.metrics import confusion_matrix
confusion_matrix(y_test,y_pred)
```

Out[56]:
```
array([[8808,    0],
       [1699,    0]], dtype=int64)
```

In [57]:
```
from sklearn.metrics import accuracy_score
accuracy_score(y_test,y_pred)
```

Out[57]:
```
0.8382982773389169
```

In [58]:
```
y=df['playlist_genre_rap']#copied to check
X=df.drop('playlist_genre_rap',axis=1)#removed for model
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.32,random_state=42)
from sklearn.linear_model import LogisticRegression
reg=LogisticRegression()
reg.fit(X_train,y_train)
y_pred=reg.predict(X_test)
y_pred
```

Out[58]:
```
array([0, 0, 0, ..., 0, 0, 0])
```

In [59]:
```
from sklearn.metrics import confusion_matrix
confusion_matrix(y_test,y_pred)
```

Out[59]:
```
array([[8658,    0],
       [1849,    0]], dtype=int64)
```

In [60]:
```
from sklearn.metrics import accuracy_score
accuracy_score(y_test,y_pred)
```

Out[60]:
```
0.82402208051775
```

In [61]:
```
y=df['playlist_genre_rock']#copied to check
X=df.drop('playlist_genre_rock',axis=1)#removed for model
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.32,random_state=42)
from sklearn.linear_model import LogisticRegression
reg=LogisticRegression()
reg.fit(X_train,y_train)
y_pred=reg.predict(X_test)
y_pred
```

Out[61]:
```
array([0, 0, 0, ..., 0, 0, 0])
```

In [62]:
```
from sklearn.metrics import confusion_matrix
confusion_matrix(y_test,y_pred)
```

Out[62]:
```
array([[8866,   12],
       [1615,   14]], dtype=int64)
```

In [63]:
```
from sklearn.metrics import accuracy_score
accuracy_score(y_test,y_pred)
```

Out[63]:
```
0.845150851813077
```

In [ ]: