In [1]:

```
# This Python 3 environment comes with many helpful analytics libr
aries installed
# It is defined by the kaggle/python docker image: https://github.
com/kaggle/docker-python
# For example, here's several helpful packages to load in

import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_
csv)
from scipy import stats
import squarify as sq
import matplotlib.pyplot as plt
from pandas.plotting import scatter_matrix
import seaborn as sns
import sklearn
import warnings
warnings.filterwarnings("ignore")
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import MinMaxScaler,LabelEncoder
from sklearn.model_selection import train_test_split,cross_val_sc
ore, KFold
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB, MultinomialNB,Bernoul
liNB
from sklearn.svm import LinearSVC, SVC
from sklearn import metrics
from sklearn.metrics import confusion_matrix, classification_repo
rt
%matplotlib inline
# Input data files are available in the "../input/" directory.
# For example, running this (by clicking run or pressing Shift+Ent
er) will list all files under the input directory

import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))

# Any results you write to the current directory are saved as outp
ut.
```

```
/kaggle/input/top50spotify2019/top50.csv
```

In [2]:

```
filename='/kaggle/input/top50spotify2019/top50.csv'
df=pd.read_csv(filename,encoding='ISO-8859-1')
df.head()
```

Out[2]:

	Unnamed: 0	Track.Name	Artist.Name	Genre	Beats.Per.Minute	Energy	Dan
0	1	Señorita	Shawn Mendes	canadian pop	117	55	76
1	2	China	Anuel AA	reggaeton flow	105	81	79

2	3	boyfriend (with Social House)	Ariana Grande	dance pop	190	80	40
3	4	Beautiful People (feat. Khalid)	Ed Sheeran	pop	93	65	64
4	5	Goodbyes (Feat. Young Thug)	Post Malone	dfw rap	150	65	58

In [3]:

```
#Calculates the number of rows and columns
print(df.shape)
```

```
(50, 14)
```

In [4]:

```
#Renaming the columns
df.rename(columns={'Track.Name':'track_name','Artist.Name':'artist_name','Beats.Per.Minute':'beats_per_minute','Loudness..dB..':'Loudness(dB)','Valence.':'Valence','Length.':'Length', 'Acousticness..':'Acousticness','Speechiness.':'Speechiness'},inplace=True)
df.head()
```

Out[4]:

	Unnamed: 0	track_name	artist_name	Genre	beats_per_minute	Energy	Dan
0	1	Señorita	Shawn Mendes	canadian pop	117	55	76
1	2	China	Anuel AA	reggaeton flow	105	81	79
2	3	boyfriend (with Social House)	Ariana Grande	dance pop	190	80	40
3	4	Beautiful People (feat. Khalid)	Ed Sheeran	pop	93	65	64
4	5	Goodbyes (Feat. Young Thug)	Post Malone	dfw rap	150	65	58

In [5]:

```
df.isnull().sum()
df.fillna(0)
```

Out[5]:

	Unnamed: 0	track_name	artist_name	Genre	beats_per_minute
0	1	Señorita	Shawn Mendes	canadian pop	117
1	2	China	Anuel AA	reggaeton flow	105
2	3	boyfriend (with Social House)	Ariana Grande	dance pop	190
3	4	Beautiful People (feat. Khalid)	Ed Sheeran	pop	93
4	5	Goodbyes (Feat.	Post Malone	dfw rap	150

4	5	Young Thug)	Post Malone	dfw rap	150
5	6	I Don't Care (with Justin Bieber)	Ed Sheeran	pop	102
6	7	Ransom	Lil Tecca	trap music	180
7	8	How Do You Sleep?	Sam Smith	pop	111
8	9	Old Town Road - Remix	Lil Nas X	country rap	136
9	10	bad guy	Billie Eilish	electropop	135
10	11	Callaita	Bad Bunny	reggaeton	176
11	12	Loco Contigo (feat. J. Balvin & Tyga)	DJ Snake	dance pop	96
12	13	Someone You Loved	Lewis Capaldi	pop	110
13	14	Otro Trago - Remix	Sech	panamanian pop	176
14	15	Money In The Grave (Drake ft. Rick Ross)	Drake	canadian hip hop	101
15	16	No Guidance (feat. Drake)	Chris Brown	dance pop	93
16	17	LA CANCIÓN	J Balvin	latin	176
17	18	Sunflower - Spider-Man: Into the Spider-Verse	Post Malone	dfw rap	90
18	19	Lalala	Y2K	canadian hip hop	130
19	20	Truth Hurts	Lizzo	escape room	158
20	21	Piece Of Your Heart	MEDUZA	pop house	124
21	22	Panini	Lil Nas X	country rap	154
22	23	No Me Conoce - Remix	Jhay Cortez	reggaeton flow	92
23	24	Soltera - Remix	Lunay	latin	92
24	25	bad guy (with Justin Bieber)	Billie Eilish	electropop	135
25	26	If I Can't Have You	Shawn Mendes	canadian pop	124
26	27	Dance Monkey	Tones and I	australian pop	98
27	28	It's You	Ali Gatie	canadian hip hop	96
28	29	Con Calma	Daddy Yankee	latin	94
29	30	QUE PRETENDES	J Balvin	latin	93
30	31	Takeaway	The Chainsmokers	edm	85
31	32	7 rings	Ariana Grande	dance pop	140
32	33	0.958333333333333	Maluma	reggaeton	96
33	34	The London (feat. J. Cole & Travis Scott)	Young Thug	atl hip hop	98
34	35	Never Really Over	Katy Perry	dance pop	100
35	36	Summer Days (feat. Macklemore & Patrick Stump ...	Martin Garrix	big room	114
36	37	Otro Trago	Sech	panamanian pop	176
37	38	Antisocial (with Travis Scott)	Ed Sheeran	pop	152
38	39	Sucker	Jonas Brothers	boy band	138
39	40	fuck, i'm lonely (with Anne-Marie) - from “13 ...	Lauv	dance pop	95
40	41	Higher Love	Kygo	edm	104
41	42	You Need To Calm Down	Taylor Swift	dance pop	85
42	43	Shallow	Lady Gaga	dance pop	96
43	44	Talk	Khalid	pop	136
44	45	Con Altura	ROSALÍA	r&b en	98

				espanol	
45	46	One Thing Right	Marshmello	brostep	88
46	47	Te Robaré	Nicky Jam	latin	176
47	48	Happier	Marshmello	brostep	100
48	49	Call You Mine	The Chainsmokers	edm	104
49	50	Cross Me (feat. Chance the Rapper & PnB Rock)	Ed Sheeran	pop	95

In [6]:

```
# The datatypes of the different attributes of the dataset
print(df.dtypes)
```

```
Unnamed: 0           int64
track_name          object
artist_name         object
Genre               object
beats_per_minute     int64
Energy               int64
Danceability         int64
Loudness(dB)         int64
Liveness             int64
Valence              int64
Length               int64
Acousticness         int64
Speechiness          int64
Popularity           int64
dtype: object
```

In [7]:

```
#Calculating the number of songs of each genre
print(type(df['Genre']))
popular_genre=df.groupby('Genre').size()
print(popular_genre)
genre_list=df['Genre'].values.tolist()
```

```
<class 'pandas.core.series.Series'>
Genre
atl hip hop         1
australian pop      1
big room            1
boy band            1
brostep             2
canadian hip hop    3
canadian pop        2
country rap         2
dance pop           8
dfw rap             2
edm                 3
electropop          2
escape room         1
latin               5
panamanian pop      2
pop                 7
pop house           1
r&b en espanol      1
```

```
r&b en español      1
reggaeton           2
reggaeton flow      2
trap music          1
dtype: int64
```

In [8]:

```
#Calculating the number of songs by each of the artists
print(df.groupby('artist_name').size())
popular_artist=df.groupby('artist_name').size()
print(popular_artist)
artist_list=df['artist_name'].values.tolist()
```

```
artist_name
Ali Gatie           1
Anuel AA            1
Ariana Grande       2
Bad Bunny           1
Billie Eilish       2
Chris Brown         1
DJ Snake            1
Daddy Yankee        1
Drake               1
Ed Sheeran          4
J Balvin            2
Jhay Cortez         1
Jonas Brothers      1
Katy Perry          1
Khalid              1
Kygo                1
Lady Gaga           1
Lauv                1
Lewis Capaldi       1
Lil Nas X           2
Lil Tecca           1
Lizzo               1
Lunay               1
MEDUZA              1
Maluma              1
Marshmello          2
Martin Garrix       1
Nicky Jam           1
Post Malone         2
ROSALÍA             1
Sam Smith           1
Sech                2
Shawn Mendes        2
Taylor Swift        1
The Chainsmokers    2
Tones and I         1
Y2K                 1
Young Thug          1
dtype: int64
artist_name
Ali Gatie           1
Anuel AA            1
Ariana Grande       2
Bad Bunny           1
Billie Eilish       2
```

```
Chris Brown          1
DJ Snake             1
Daddy Yankee         1
Drake                1
Ed Sheeran           4
J Balvin             2
Jhay Cortez          1
Jonas Brothers       1
Katy Perry           1
Khalid               1
Kygo                 1
Lady Gaga            1
Lauv                 1
Lewis Capaldi        1
Lil Nas X            2
Lil Tecca            1
Lizzo                1
Lunay                1
MEDUZA               1
Maluma               1
Marshmello           2
Martin Garrix        1
Nicky Jam            1
Post Malone          2
ROSALÍA              1
Sam Smith            1
Sech                 2
Shawn Mendes         2
Taylor Swift         1
The Chainsmokers     2
Tones and I          1
Y2K                  1
Young Thug           1
dtype: int64
```

In [9]:

```
df.isnull().sum()
df.fillna(0)
```

Out[9]:

	Unnamed: 0	track_name	artist_name	Genre	beats_per_minute
0	1	Señorita	Shawn Mendes	canadian pop	117
1	2	China	Anuel AA	reggaeton flow	105
2	3	boyfriend (with Social House)	Ariana Grande	dance pop	190
3	4	Beautiful People (feat. Khalid)	Ed Sheeran	pop	93
4	5	Goodbyes (Feat. Young Thug)	Post Malone	dfw rap	150
5	6	I Don't Care (with Justin Bieber)	Ed Sheeran	pop	102
6	7	Ransom	Lil Tecca	trap music	180
7	8	How Do You Sleep?	Sam Smith	pop	111
8	9	Old Town Road - Remix	Lil Nas X	country rap	136
9	10	bad guy	Billie Eilish	electropop	135
10	11	Callaita	Bad Bunny	reggaeton	176
11	12	Loco Contigo (feat. J.	DJ Snake	dance pop	96

		Balvin & Tyga)			
12	13	Someone You Loved	Lewis Capaldi	pop	110
13	14	Otro Trago - Remix	Sech	panamanian pop	176
14	15	Money In The Grave (Drake ft. Rick Ross)	Drake	canadian hip hop	101
15	16	No Guidance (feat. Drake)	Chris Brown	dance pop	93
16	17	LA CANCIÓN	J Balvin	latin	176
17	18	Sunflower - Spider-Man: Into the Spider-Verse	Post Malone	dfw rap	90
18	19	Lalala	Y2K	canadian hip hop	130
19	20	Truth Hurts	Lizzo	escape room	158
20	21	Piece Of Your Heart	MEDUZA	pop house	124
21	22	Panini	Lil Nas X	country rap	154
22	23	No Me Conoce - Remix	Jhay Cortez	reggaeton flow	92
23	24	Soltera - Remix	Lunay	latin	92
24	25	bad guy (with Justin Bieber)	Billie Eilish	electropop	135
25	26	If I Can't Have You	Shawn Mendes	canadian pop	124
26	27	Dance Monkey	Tones and I	australian pop	98
27	28	It's You	Ali Gatie	canadian hip hop	96
28	29	Con Calma	Daddy Yankee	latin	94
29	30	QUE PRETENDES	J Balvin	latin	93
30	31	Takeaway	The Chainsmokers	edm	85
31	32	7 rings	Ariana Grande	dance pop	140
32	33	0.958333333333333	Maluma	reggaeton	96
33	34	The London (feat. J. Cole & Travis Scott)	Young Thug	atl hip hop	98
34	35	Never Really Over	Katy Perry	dance pop	100
35	36	Summer Days (feat. Macklemore & Patrick Stump ...	Martin Garrix	big room	114
36	37	Otro Trago	Sech	panamanian pop	176
37	38	Antisocial (with Travis Scott)	Ed Sheeran	pop	152
38	39	Sucker	Jonas Brothers	boy band	138
39	40	fuck, i'm lonely (with Anne-Marie) - from “13 ...	Lauv	dance pop	95
40	41	Higher Love	Kygo	edm	104
41	42	You Need To Calm Down	Taylor Swift	dance pop	85
42	43	Shallow	Lady Gaga	dance pop	96
43	44	Talk	Khalid	pop	136
44	45	Con Altura	ROSALÍA	r&b en espanol	98
45	46	One Thing Right	Marshmello	brostep	88
46	47	Te Robaré	Nicky Jam	latin	176
47	48	Happier	Marshmello	brostep	100
48	49	Call You Mine	The Chainsmokers	edm	104
49	50	Cross Me (feat. Chance the Rapper & PnB Rock)	Ed Sheeran	pop	95

In [10]:

```
pd.set_option('precision', 3)
df.describe()
```

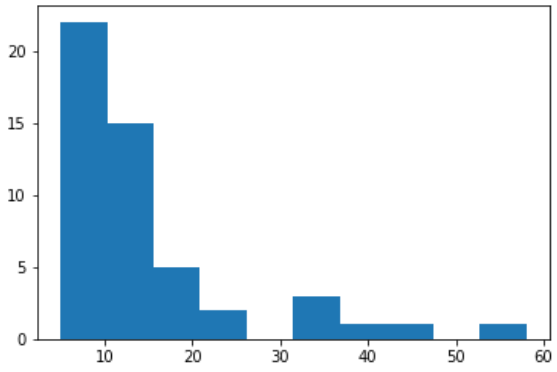
Out[10]:

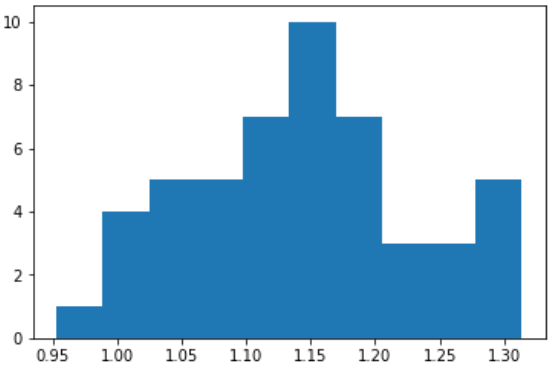
	Unnamed: 0	beats_per_minute	Energy	Danceability	Loudness(dB)	Liveness
count	50.000	50.000	50.000	50.00	50.000	50.000
mean	25.500	120.060	64.060	71.38	-5.660	14.660
std	14.577	30.898	14.232	11.93	2.056	11.118
min	1.000	85.000	32.000	29.00	-11.000	5.000
25%	13.250	96.000	55.250	67.00	-6.750	8.000
50%	25.500	104.500	66.500	73.50	-6.000	11.000
75%	37.750	137.500	74.750	79.75	-4.000	15.750
max	50.000	190.000	88.000	90.00	-2.000	58.000

In [11]:

```
#Finding out the skew for each attribute
skew=df.skew()
print(skew)
# Removing the skew by using the boxcox transformations
transform=np.asarray(df[['Liveness']].values)
df_transform = stats.boxcox(transform)[0]
# Plotting a histogram to show the difference
plt.hist(df['Liveness'],bins=10) #original data
plt.show()
plt.hist(df_transform,bins=10) #corrected skew data
plt.show()
```

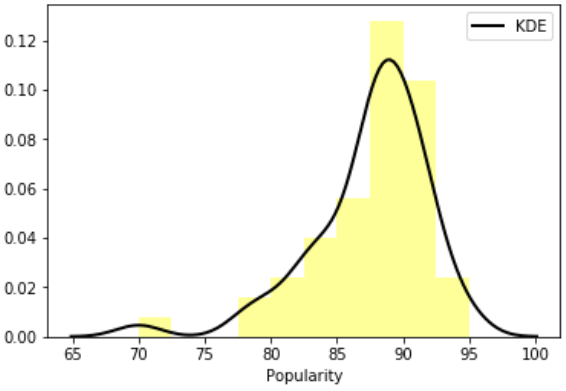
```
Unnamed: 0          0.000
beats_per_minute    0.855
Energy             -0.453
Danceability       -1.380
Loudness(dB)       -0.832
Liveness            2.204
Valence            -0.046
Length              0.749
Acousticness        1.135
Speechiness         1.378
Popularity         -1.503
dtype: float64
```

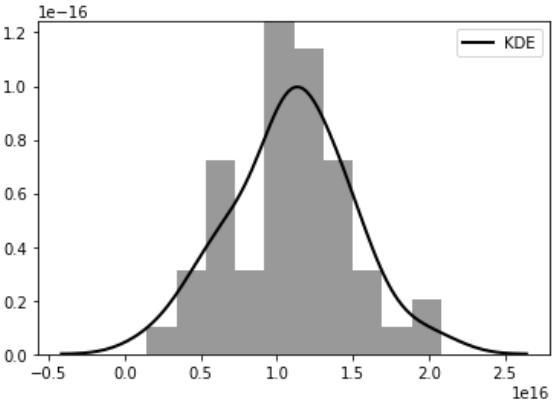




In [12]:

```
transform1=np.asarray(df[['Popularity']].values)
df_transform1 = stats.boxcox(transform1)[0]
# Plotting a histogram to show the difference
# plt.hist(df['Popularity'],bins=10) original data
# plt.show()
# plt.hist(df_transform1,bins=10) #corrected skew data
# plt.show()
sns.distplot(df['Popularity'],bins=10,kde=True,kde_kws={"color":
"k", "lw": 2, "label": "KDE"},color='yellow')
plt.show()
sns.distplot(df_transform1,bins=10,kde=True,kde_kws={"color": "k"
, "lw": 2, "label": "KDE"},color='black') #corrected skew data
plt.show()
```





In [13]:

```
pd.set_option('display.width', 100)
pd.set_option('precision', 3)
```

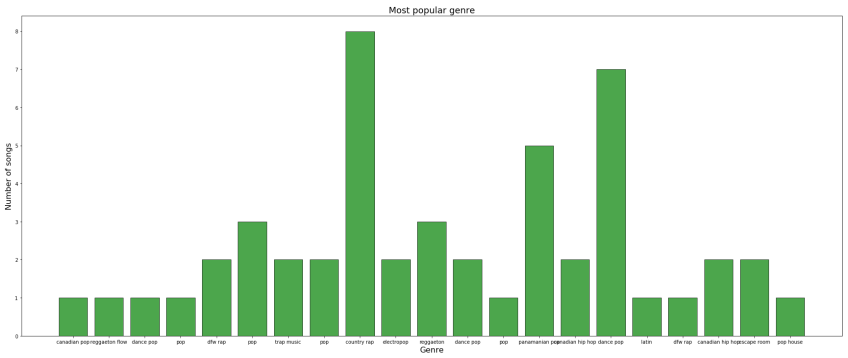
```
correlation=df.corr(method='spearman')
print(correlation)
```

```
                 Unnamed: 0  beats_per_minute  Energy  Dance
ability  Loudness(dB)  Liveness  \
Unnamed: 0            1.000           -0.263   0.132
0.053        -0.014     0.102
beats_per_minute     -0.263            1.000   0.012
-0.092         0.014    -0.033
Energy                0.132            0.012   1.000
-0.049         0.635     0.013
Danceability          0.053           -0.092  -0.049
1.000         0.009    -0.261
Loudness(dB)         -0.014            0.014   0.635
0.009         1.000     0.114
Liveness              0.102           -0.033   0.013
-0.261         0.114     1.000
Valence               0.113           -0.048   0.467
0.155         0.317    -0.187
Length                0.045           -0.198   0.189
-0.079         0.165     0.202
Acousticness          0.058           -0.010  -0.211
-0.128        -0.040     0.204
Speechiness          -0.232            0.392  -0.035
0.104        -0.063    -0.137
Popularity           -0.221            0.217  -0.044
-0.141         0.072     0.012

                 Valence  Length  Acousticness  Speechiness
Popularity
Unnamed: 0         0.113   0.045         0.058       -0.232
-0.221
beats_per_minute  -0.048  -0.198        -0.010        0.392
0.217
Energy             0.467   0.189        -0.211       -0.035
-0.044
Danceability       0.155  -0.079        -0.128        0.104
-0.141
Loudness(dB)       0.317   0.165        -0.040       -0.063
0.072
Liveness          -0.187   0.202         0.204       -0.137
0.012
Valence            1.000  -0.081        -0.053       -0.095
-0.265
Length            -0.081   1.000        -0.005        0.020
-0.122
Acousticness      -0.053  -0.005         1.000        0.017
0.036
Speechiness       -0.095   0.020         0.017        1.000
0.165
Popularity        -0.265  -0.122         0.036        0.165
1.000
```

In [14]:

```
# Bar graph to see the number of songs of each genre
fig, ax=plt.subplots(figsize=(30,12))
length=np.arange(len(popular_genre))
plt.bar(length,popular_genre,color='green',edgecolor='black',alph
a=0.7)
```

```
plt.xticks(length,genre_list)
plt.title('Most popular genre',fontsize=18)
plt.xlabel('Genre',fontsize=16)
plt.ylabel('Number of songs',fontsize=16)
plt.show()
```

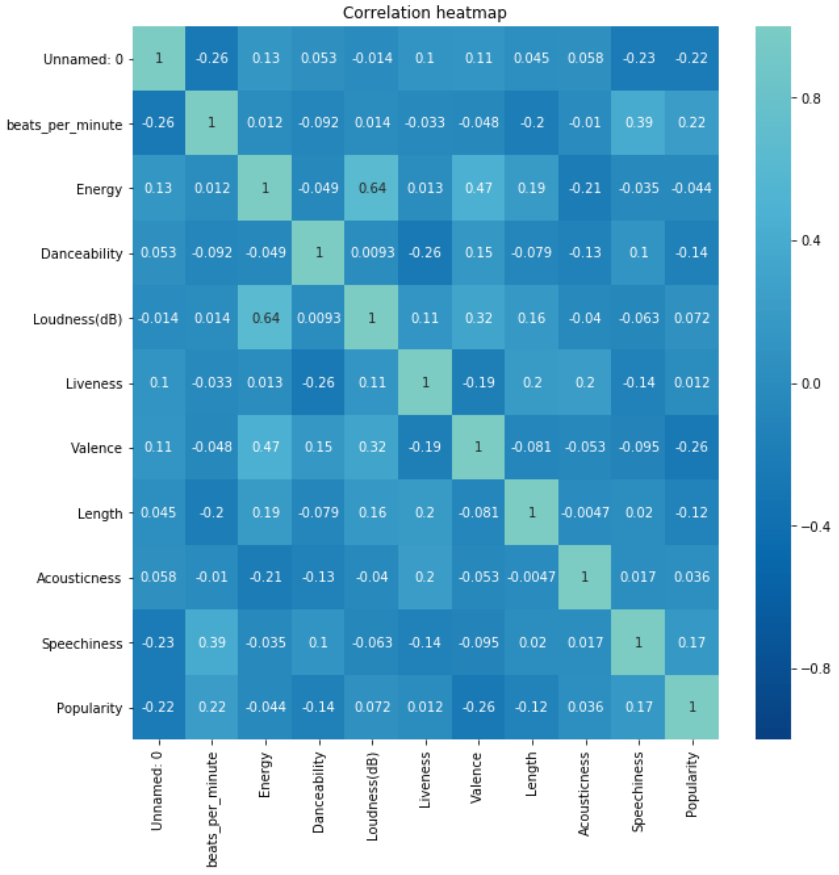


In [15]:

```
# heatmap of the correlation
plt.figure(figsize=(10,10))
plt.title('Correlation heatmap')
sns.heatmap(correlation,annot=True,vmin=-1,vmax=1,cmap="GnBu_r",center=1)
```

Out[15]:

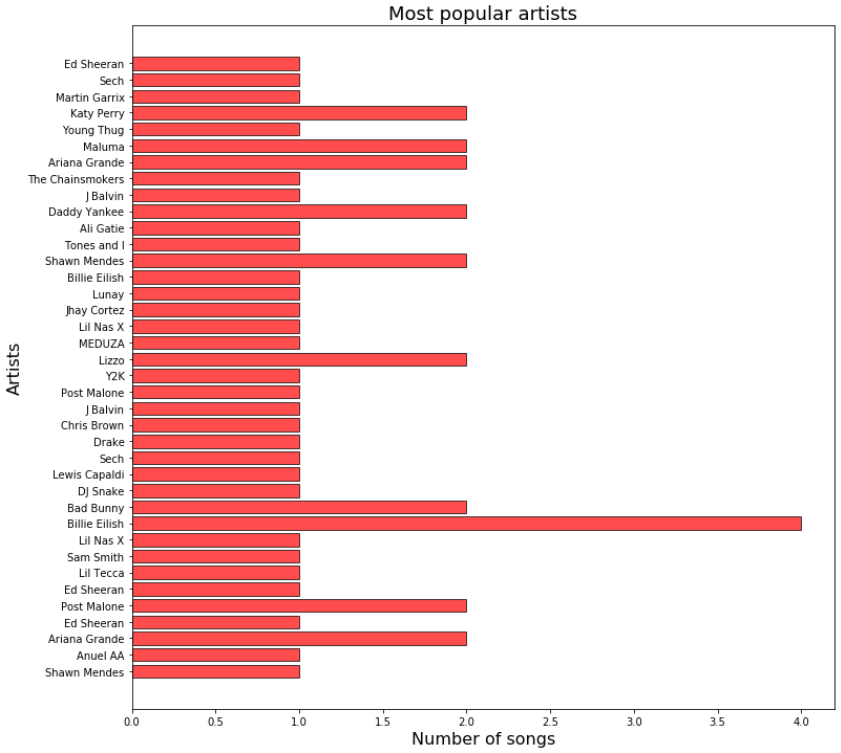
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fd774913710>
```



In [16]:

```
fig, ax=plt.subplots(figsize=(12,12))
length=np.arange(len(popular_artist))
plt.barh(length,popular_artist,color='red',edgecolor='black',alph
```

```
a=0.7)
plt.yticks(length,artist_list)
plt.title('Most popular artists',fontsize=18)
plt.ylabel('Artists',fontsize=16)
plt.xlabel('Number of songs',fontsize=16)
plt.show()
```

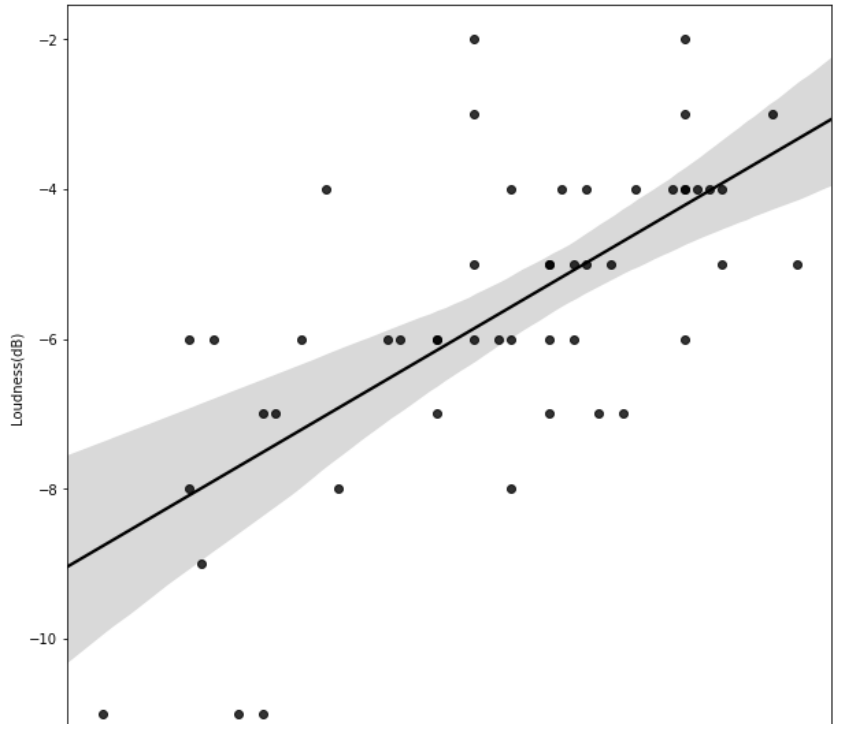


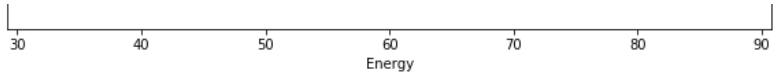
In [17]:

```
# Analysing the relationship between energy and loudness
fig=plt.subplots(figsize=(10,10))
sns.regplot(x='Energy',y='Loudness(dB)',data=df,color='black')
```

Out[17]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fd770ea8438>
```



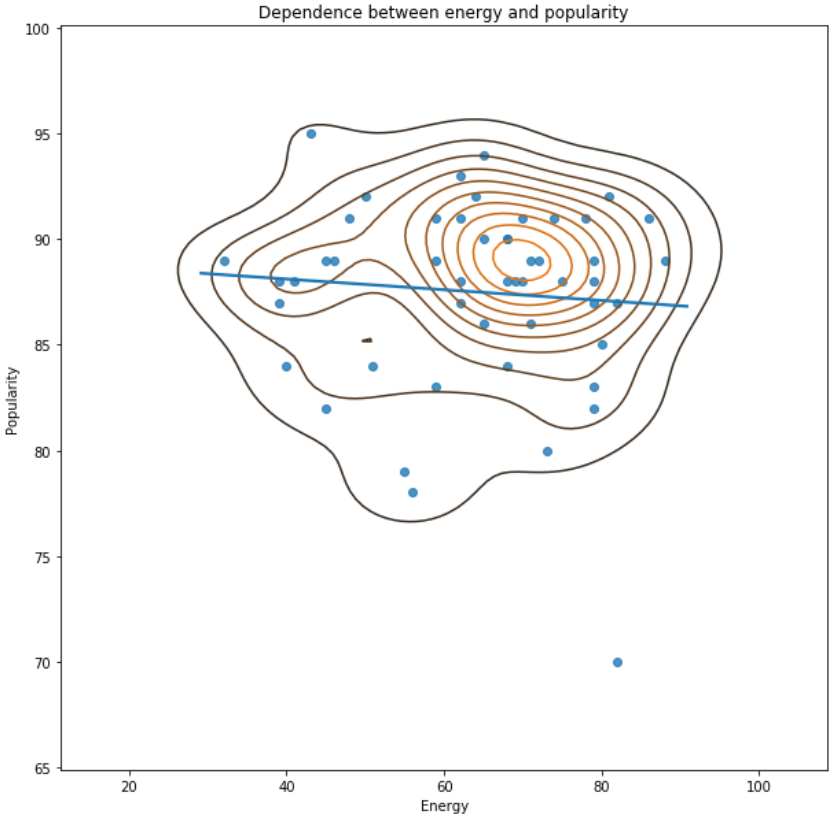


In [18]:

```
fig=plt.subplots(figsize=(10,10))
plt.title('Dependence between energy and popularity')
sns.regplot(x='Energy', y='Popularity',
            ci=None, data=df)
sns.kdeplot(df.Energy,df.Popularity)
```

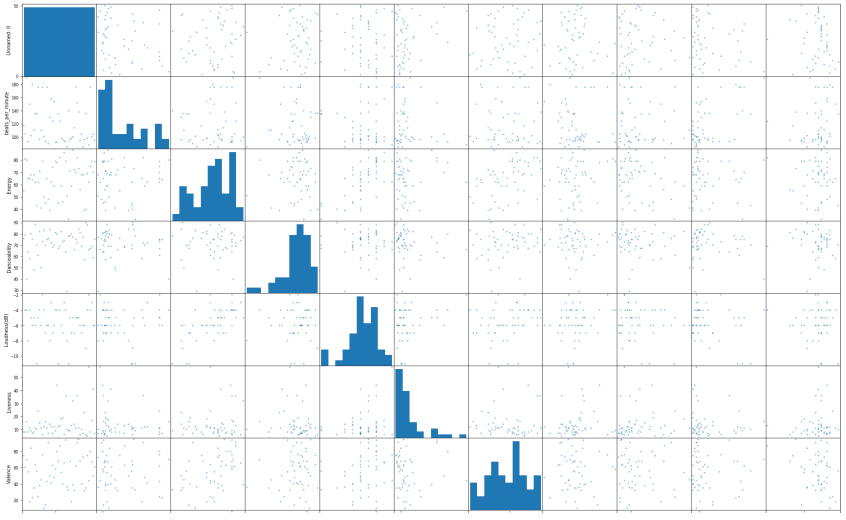
Out[18]:

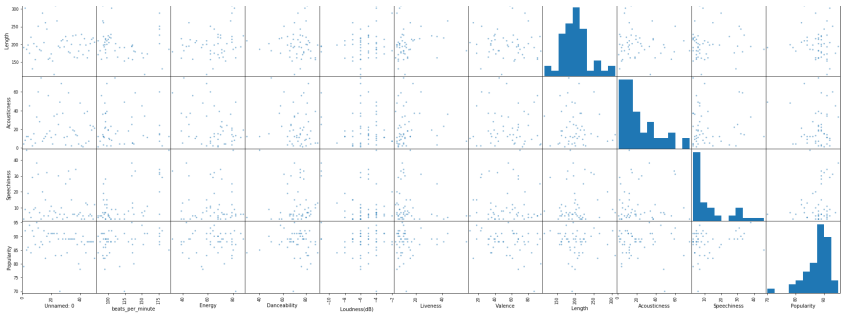
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fd770f47b38>
```



In [19]:

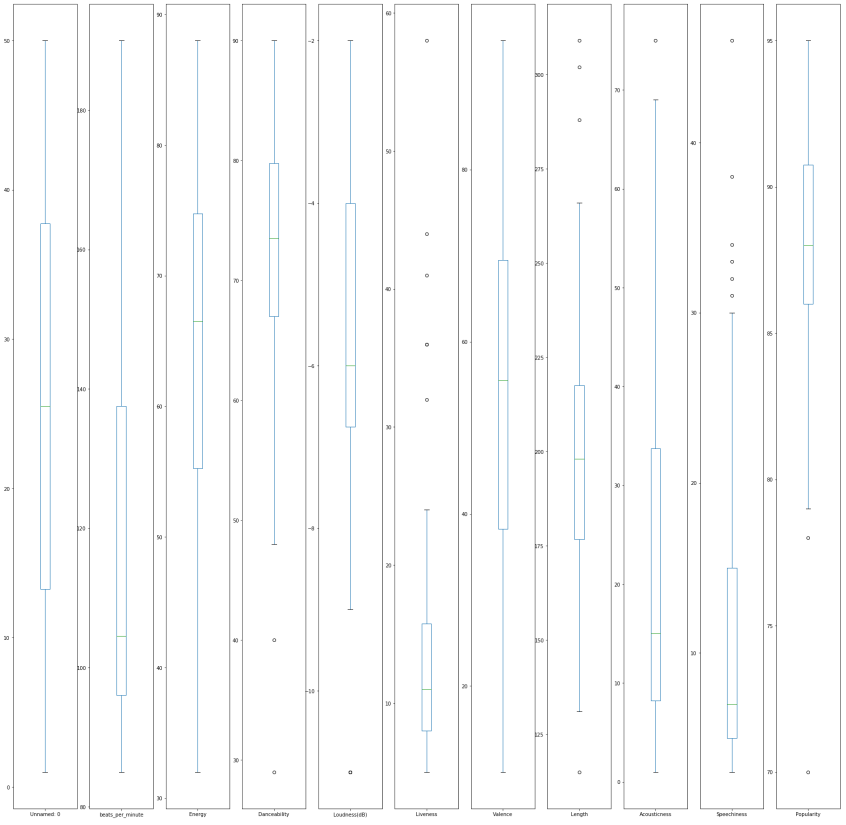
```
scatter_matrix(df)
plt.gcf().set_size_inches(30, 30)
plt.show()
```





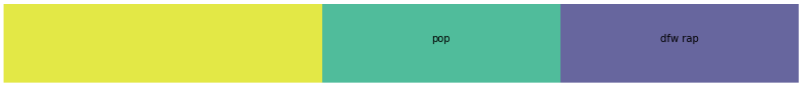
In [20]:

```
df.plot(kind='box', subplots=True)
plt.gcf().set_size_inches(30,30)
plt.show()
```



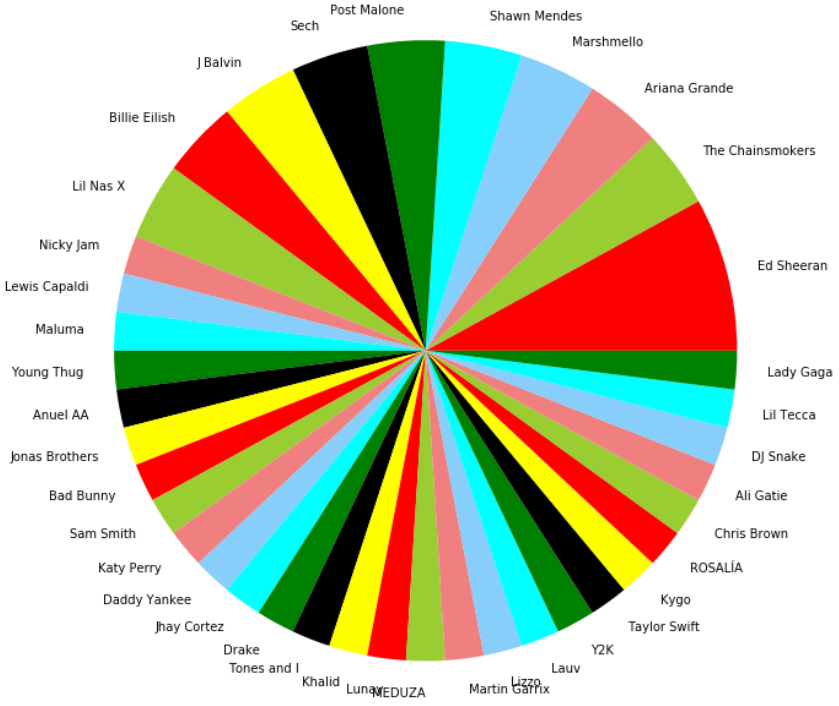
In [21]:

```
plt.figure(figsize=(14,8))
sq.plot(sizes=df.Genre.value_counts(), label=df["Genre"].unique
(), alpha=.8 )
plt.axis('off')
plt.show()
```



In [22]:

```
#Pie charts
labels = df.artist_name.value_counts().index
sizes = df.artist_name.value_counts().values
colors = ['red', 'yellowgreen', 'lightcoral', 'lightskyblue','cya
n', 'green', 'black','yellow']
plt.figure(figsize = (10,10))
plt.pie(sizes, labels=labels, colors=colors)
autopct=('%1.1f%%')
plt.axis('equal')
plt.show()
```



In [23]:

```
#Linear regression, first create test and train dataset
x=df.loc[:,['Energy','Danceability','Length','Loudness(dB)','Acou
sticness']].values
y=df.loc[:,'Popularity'].values
```

In [24]:

```
# Creating a test and training dataset
X_train, X_test, y_train, y_test = train_test_split(x, y, test_si
ze=0.30)
```

In [25]:

```
# Linear regression
regressor = LinearRegression()
regressor.fit(X_train, y_train)
print(regressor.intercept_)
print(regressor.coef_)
```

```
102.26857557268886
```

```
[-0.05209151 -0.04721667 -0.04042977 -0.02023469 -0.06181807]
```

In [26]:

```
#Displaying the difference between the actual and the predicted
y_pred = regressor.predict(X_test)
df_output = pd.DataFrame({'Actual': y_test, 'Predicted': y_pred})
print(df_output)
```

	Actual	Predicted
0	83	82.689
1	91	85.180
2	88	85.640
3	92	90.095
4	90	84.711
5	92	81.696
6	87	87.345
7	93	82.403
8	89	85.668
9	86	85.569
10	94	86.368
11	91	81.529
12	89	86.257
13	90	88.237
14	87	82.670

In [27]:

```
#Checking the accuracy of Linear Regression
print('Mean Absolute Error:', metrics.mean_absolute_error(y_test,
y_pred))
print('Mean Squared Error:', metrics.mean_squared_error(y_test, y
_pred))
print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_er
ror(y_test, y_pred)))
```

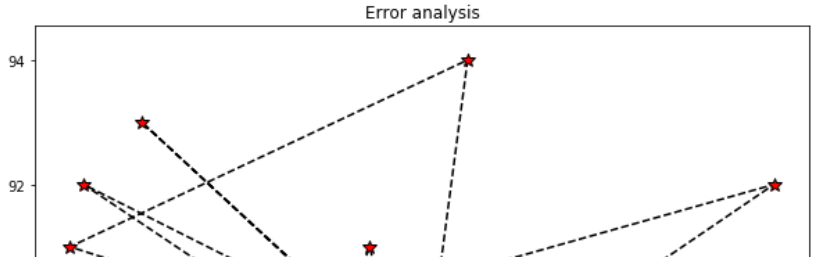
```
Mean Absolute Error: 4.442246487057812
Mean Squared Error: 31.890021234522354
Root Mean Squared Error: 5.647125041516467
```

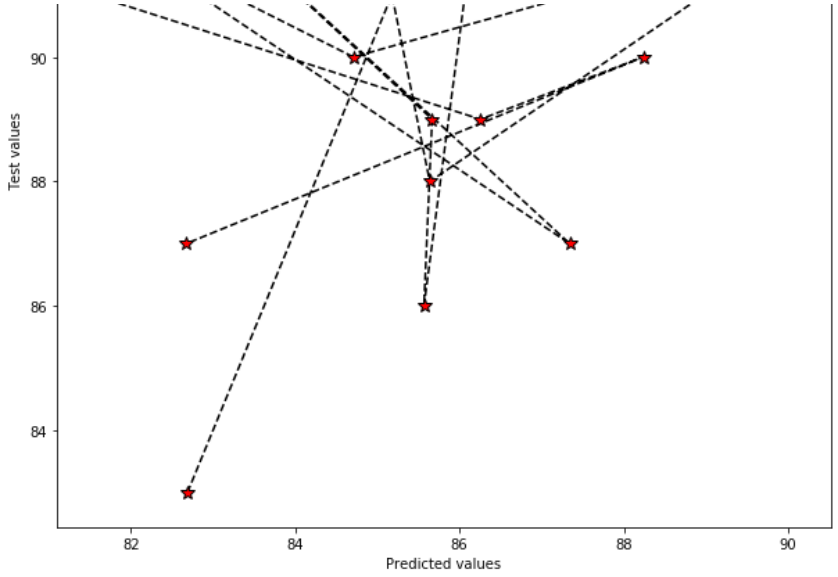
In [28]:

```
plt.figure(figsize=(10,10))
plt.plot(y_pred,y_test,color='black',linestyle='dashed',marker=
'*',markerfacecolor='red',markersize=10)
plt.title('Error analysis')
plt.xlabel('Predicted values')
plt.ylabel('Test values')
```

Out[28]:

```
Text(0, 0.5, 'Test values')
```





In [29]:

```
# Cross validation score
x=df.loc[:,['Energy','Danceability']].values
y=df.loc[:,'Popularity'].values
regressor=LinearRegression()
mse=cross_val_score(regressor,X_train,y_train,scoring='neg_mean_squared_error',cv=5)
mse_mean=np.mean(mse)
print(mse_mean)
diff=metrics.mean_squared_error(y_test, y_pred)-abs(mse_mean)
print(diff)
```

```
-32.650937347082944
-0.7609161125605901
```

In [30]:

```
x=df.loc[:,['artist_name']].values
y=df.loc[:,'Genre'].values
```

In [31]:

```
# Label encoding of features
x.shape
encoder=LabelEncoder()
x = encoder.fit_transform(x)
x=pd.DataFrame(x)
x
```

Out[31]:

	0
0	32
1	1
2	2
3	9
4	28
5	9
6	20
7	30
8	19
9	4

10	3
11	6
12	18
13	31
14	8
15	5
16	10
17	28
18	36
19	21
20	23
21	19
22	11
23	22
24	4
25	32
26	35
27	0
28	7
29	10
30	34
31	2
32	24
33	37
34	13
35	26
36	31
37	9
38	12
39	17
40	15
41	33
42	16
43	14
44	29
45	25
46	27
47	25
48	34
49	9

In [32]:

```
# Label Encoding of target
Encoder_y=LabelEncoder()
Y = Encoder_y.fit_transform(y)
Y=pd.DataFrame(Y)
Y
```

Out[32]:

	0
0	6
1	19
2	8
3	15
4	9
5	15

5	15
6	20
7	15
8	7
9	11
10	18
11	8
12	15
13	14
14	5
15	8
16	13
17	9
18	5
19	12
20	16
21	7
22	19
23	13
24	11
25	6
26	1
27	5
28	13
29	13
30	10
31	8
32	18
33	0
34	8
35	2
36	14
37	15
38	3
39	8
40	10
41	8
42	8
43	15
44	17
45	4
46	13
47	4
48	10
49	15

In [33]:

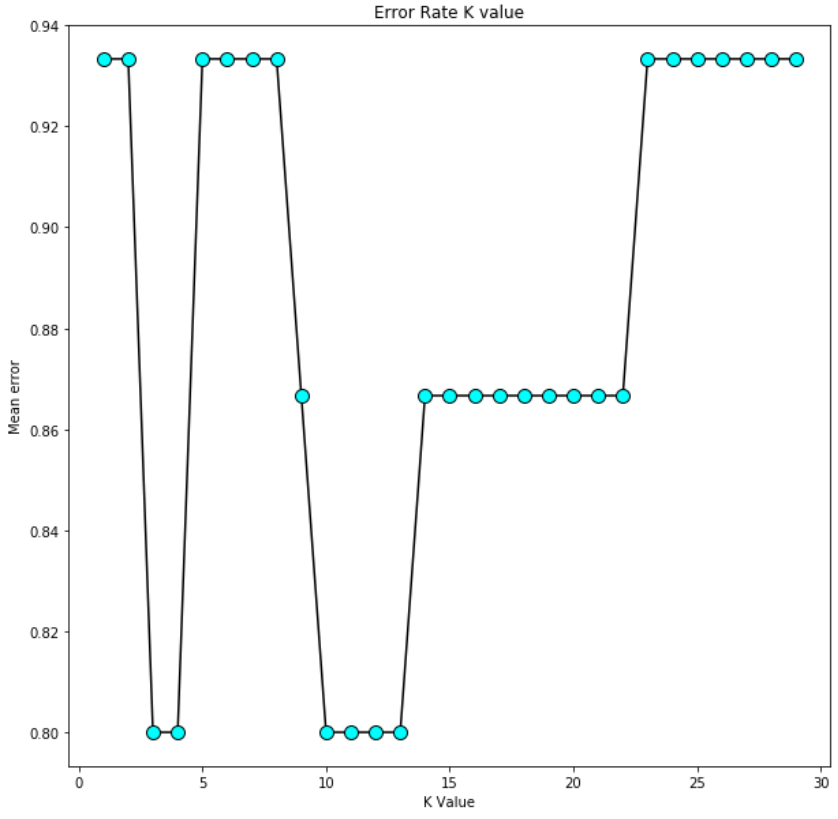
```
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size =
0.3,random_state = 1)

#Scaling
from sklearn.preprocessing import StandardScaler
sc=StandardScaler()
sc.fit(x_train)
x_train=sc.transform(x_train)
x_test=sc.transform(x_test)
```

In [34]:
```
# KNN Classification
# sorted(sklearn.neighbors.VALID_METRICS['brute'])
knn = KNeighborsClassifier(n_neighbors = 17)
knn.fit(x_train,y_train)
y_pred=knn.predict(x_test)
```

In [35]:
```
error=[]
for i in range(1,30):
    knn=KNeighborsClassifier(n_neighbors=i)
    knn.fit(X_train,y_train)
    pred_i=knn.predict(X_test)
    error.append(np.mean(pred_i!=y_test))
```

In [36]:
```
plt.figure(figsize=(10,10))
plt.plot(range(1,30),error,color='black',marker='o',markerfacecol
or='cyan',markersize=10)
plt.title('Error Rate K value')
plt.xlabel('K Value')
plt.ylabel('Mean error')
```

Out[36]:
```
Text(0, 0.5, 'Mean error')
```



In [37]:
```
x=df.loc[:,['Energy','Length','Danceability','beats_per_minute',
'Acousticness']].values
y=df.loc[:,'Popularity'].values
```

In [38]:

```
# Creating a test and training dataset
X_train, X_test, y_train, y_test = train_test_split(x, y, test_si
ze=0.30)
```

In [39]:

```
gnb = GaussianNB()
gnb.fit(X_train, y_train)
y_pred=gnb.predict(X_test)
df_output = pd.DataFrame({'Actual': y_test, 'Predicted': y_pred})
print(df_output)
```

	Actual	Predicted
0	78	83
1	82	88
2	84	92
3	93	88
4	87	91
5	91	88
6	89	89
7	91	87
8	90	91
9	89	86
10	92	91
11	90	88
12	91	91
13	70	88
14	89	88

In [40]:

```
# Testing the accuracy of Naive Bayes
scores=cross_val_score(gnb,X_train,y_train,scoring='accuracy',cv=
3).mean()*100
print(scores)
```

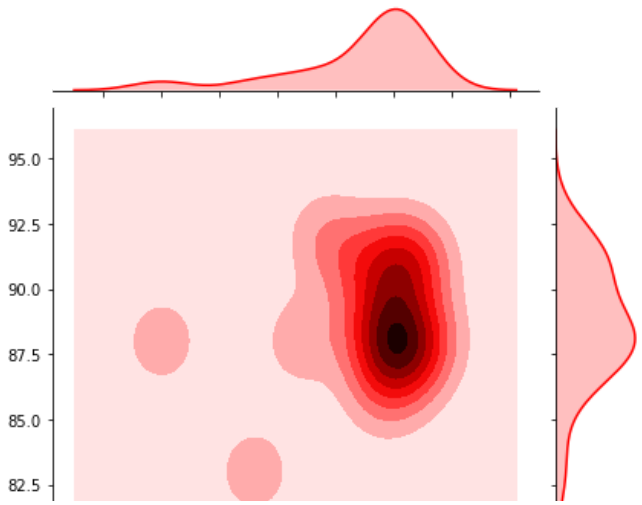
```
14.48165869218501
```

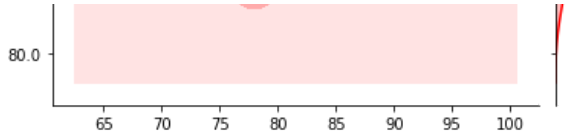
In [41]:

```
sns.jointplot(x=y_test, y=y_pred, kind="kde", color="r")
```

Out[41]:

```
<seaborn.axisgrid.JointGrid at 0x7fd7644af048>
```





In [42]:
```
x=df.loc[:,['Energy','Length','Danceability','beats_per_minute',
'Acousticness']].values
y=df.loc[:,'Popularity'].values
```

In [43]:
```
X_train, X_test, y_train, y_test = train_test_split(x, y, test_si
ze=0.30)
```

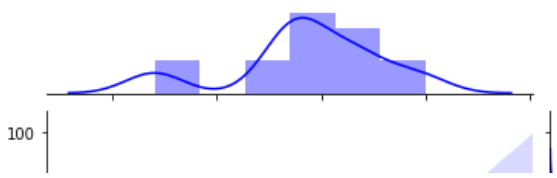
In [44]:
```
# Linear SVM model
LinSVC = LinearSVC(penalty='l2', loss='squared_hinge', dual=True)
LinSVC.fit(X_train, y_train)
y_pred=gnb.predict(X_test)
df_output = pd.DataFrame({'Actual': y_test, 'Predicted': y_pred})
print(df_output)
```

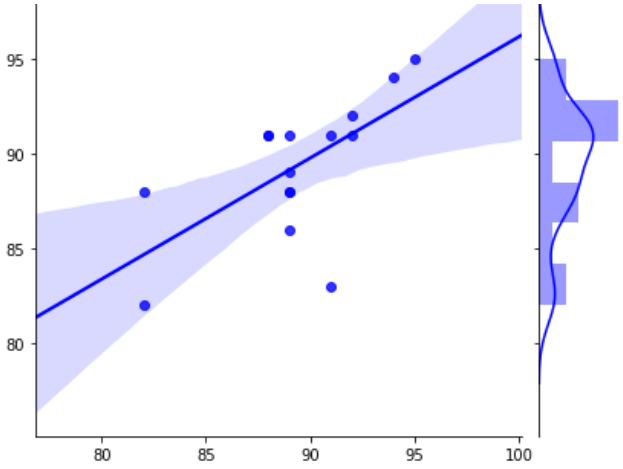
	Actual	Predicted
0	89	86
1	95	95
2	91	91
3	94	94
4	88	91
5	82	88
6	91	83
7	89	88
8	88	91
9	92	91
10	89	89
11	92	92
12	82	82
13	89	88
14	89	91

In [45]:
```
# Testing the accuracy
scores=cross_val_score(LinSVC,X_train,y_train,scoring='accuracy',
cv=3).mean()*100
print(scores)
```

```
8.465608465608465
```

In [46]:
```
sns.jointplot(x=y_test, y=y_pred, kind="reg", color="b");
```





In [ ]:

Data

Data Sources

- Top 50 Spotify Songs - 2019
  - top50.csv — 14 columns

Top 50 Spotify Songs - 2019

Top 50 songs listened in 2019 on spotify

Last Updated: 6 months ago (Version 1)

About this Dataset

- Check the data extracted by **year**: https://www.kaggle.com/leonardopena/top-spotify-songs-from-20102019-by-year
- And by **country**: https://www.kaggle.com/leonardopena/top-50-spotify-songs-by-each-country

Context