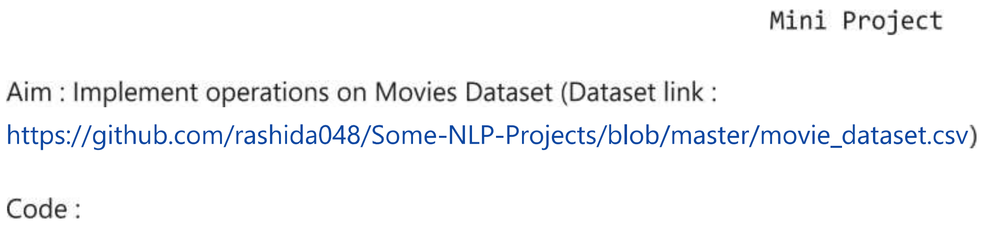


Mini Project

Aim : Implement operations on Movies Dataset (Dataset link : https://github.com/rashida048/Some-NLP-Projects/blob/master/movie_dataset.csv)

Code :

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from pandas import DataFrame, Series
```

In [2]:
```
df1 = pd.read_csv("movies.csv")
df1
```

Out[2]:

	index	budget	genres	homepage	
0	0	237000000	Action Adventure Fantasy Science Fiction	http://www.avatarmovie.com/	
1	1	300000000	Adventure Fantasy Action	http://disney.go.com/disneypictures/pirates/	
2	2	245000000	Action Adventure Crime	http://www.sonypictures.com/movies/spectre/	2
3	3	250000000	Action Crime Drama Thriller	http://www.thedarkknightrises.com/	
4	4	260000000	Action Adventure Science Fiction	http://movies.disney.com/john-carter	
...	...	...	...	...	
4798	4798	220000	Action Crime Thriller	NaN	
4799	4799	9000	Comedy Romance	NaN	
4800	4800	0	Comedy Drama Romance TV Movie	http://www.hallmarkchannel.com/signedsealeddel...	2
4801	4801	0	NaN	http://shanghaicalling.com/	1

	index	budget	genres	homepage
4802	4802	0	Documentary	NaN

4803 rows × 24 columns

In [3]:
```
df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4803 entries, 0 to 4802
Data columns (total 24 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   index                 4803 non-null   int64  
 1   budget                4803 non-null   int64  
 2   genres                4775 non-null   object 
 3   homepage              1712 non-null   object 
 4   id                    4803 non-null   int64  
 5   keywords              4391 non-null   object 
 6   original_language     4803 non-null   object 
 7   original_title        4803 non-null   object 
 8   overview              4800 non-null   object 
 9   popularity            4803 non-null   float64
 10  production_companies  4803 non-null   object 
 11  production_countries  4803 non-null   object 
 12  release_date          4802 non-null   object 
 13  revenue               4803 non-null   int64  
 14  runtime               4801 non-null   float64
 15  spoken_languages      4803 non-null   object 
 16  status                4803 non-null   object 
 17  tagline               3959 non-null   object 
 18  title                 4803 non-null   object 
 19  vote_average          4803 non-null   float64
 20  vote_count            4803 non-null   int64  
 21  cast                  4760 non-null   object 
 22  crew                  4803 non-null   object 
 23  director              4773 non-null   object 
dtypes: float64(3), int64(5), object(16)
memory usage: 900.7+ KB
```

In [4]:
```
df1.describe()
```

Out[4]:

	index	budget	id	popularity	revenue	runtin
count	4803.000000	4.803000e+03	4803.000000	4803.000000	4.803000e+03	4801.00000
mean	2401.000000	2.904504e+07	57165.484281	21.492301	8.226064e+07	106.8758
std	1386.651002	4.072239e+07	88694.614033	31.816650	1.628571e+08	22.6119
min	0.000000	0.000000e+00	5.000000	0.000000	0.000000e+00	0.0000
25%	1200.500000	7.900000e+05	9014.500000	4.668070	0.000000e+00	94.0000
50%	2401.000000	1.500000e+07	14629.000000	12.921594	1.917000e+07	103.0000
75%	3601.500000	4.000000e+07	58610.500000	28.313505	9.291719e+07	118.0000
max	4802.000000	3.800000e+08	459488.000000	875.581305	2.787965e+09	338.0000

In [5]:

```
df1.head()
```

Out[5]:

	index	budget	genres	homepage	id	key
0	0	237000000	Action Adventure Fantasy Science Fiction	http://www.avatarmovie.com/	19995	spa
1	1	300000000	Adventure Fantasy Action	http://disney.go.com/disneypictures/pirates/	285	eas
2	2	245000000	Action Adventure Crime	http://www.sonypictures.com/movies/spectre/	206647	spy or
3	3	250000000	Action Crime Drama Thriller	http://www.thedarkknightrises.com/	49026	dc te
4	4	260000000	Action Adventure Science Fiction	http://movies.disney.com/john-carter	49529	ba me

5 rows × 24 columns

In [6]:
```
df1.tail()
```

Out[6]:

	index	budget	genres	homepage	
4798	4798	220000	Action Crime Thriller	NaN	93
4799	4799	9000	Comedy Romance	NaN	727
4800	4800	0	Comedy Drama Romance TV Movie	http://www.hallmarkchannel.com/signedsealeddel...	2316
4801	4801	0	NaN	http://shanghaicalling.com/	1261
4802	4802	0	Documentary	NaN	259

5 rows × 24 columns

In [12]:

```
top_left_corner_df = df1.iloc[:4, :4]
print(top_left_corner_df)
```

```
   index     budget                                   genres  \
0      0  237000000  Action Adventure Fantasy Science Fiction   
1      1  300000000                  Adventure Fantasy Action   
2      2  245000000                    Action Adventure Crime   
3      3  250000000               Action Crime Drama Thriller   

                                      homepage  
0                  http://www.avatarmovie.com/  
1  http://disney.go.com/disneypictures/pirates/  
2   http://www.sonypictures.com/movies/spectre/  
3             http://www.thedarkknightrises.com/  
```

In [13]:

```
df1.to_csv()
```

```
2     [{'name': 'Thomas Newman', 'gender': 2, 'depar...         Sam Mendes
3     [{'name': 'Hans Zimmer', 'gender': 2, 'departm...  Christopher Nolan
4     [{'name': 'Andrew Stanton', 'gender': 2, 'depa...     Andrew Stanton
...                                                 ...                ...
4798  [{'name': 'Robert Rodriguez', 'gender': 0, 'de...   Robert Rodriguez
4799  [{'name': 'Edward Burns', 'gender': 2, 'depart...       Edward Burns
4800  [{'name': 'Carla Hetland', 'gender': 0, 'depar...        Scott Smith
4801  [{'name': 'Daniel Hsia', 'gender': 2, 'departm...        Daniel Hsia
4802  [{'name': 'Clark Peterson', 'gender': 2, 'depa...   Brian Herzlinger

[4803 rows x 24 columns]
```

In [16]:
```
df1.count()
```

Out[16]:
```
index                   4803
budget                  4803
genres                  4775
homepage                1712
id                      4803
keywords                4391
original_language       4803
original_title          4803
overview                4800
popularity              4803
production_companies    4803
production_countries    4803
release_date            4802
revenue                 4803
runtime                 4801
spoken_languages        4803
status                  4803
tagline                 3959
title                   4803
vote_average            4803
vote_count              4803
cast                    4760
crew                    4803
director                4773
dtype: int64
```

In [19]:
```
df1.dropna()
```

Out[19]:

	index	budget	genres	homepage	id
0	0	237000000	Action Adventure Fantasy Science Fiction	http://www.avatarmovie.com/	19995
1	1	300000000	Adventure Fantasy Action	http://disney.go.com/disneypictures/pirates/	285
2	2	245000000	Action Adventure Crime	http://www.sonypictures.com/movies/spectre/	206647
3	3	250000000	Action Crime Drama Thriller	http://www.thedarkknightrises.com/	49026
4	4	260000000	Action Adventure Science Fiction	http://movies.disney.com/john-carter	49529
...	...	...	...	...	...
4772	4772	31192	Drama Action Comedy	http://downterrace.blogspot.com/	42151
4773	4773	27000	Comedy	http://www.miramax.com/movie/clerks/	2292
4781	4781	22000	Comedy Romance	https://www.facebook.com/DrySpellMovie	255266
4791	4791	13	Horror	http://tincanmanthemovie.com/	157185

	index	budget	genres	homepage	id
4796	4796	7000	Science Fiction Drama Thriller	http://www.primermovie.com	14337

1432 rows × 24 columns

In [20]:
```
df1.any()
```

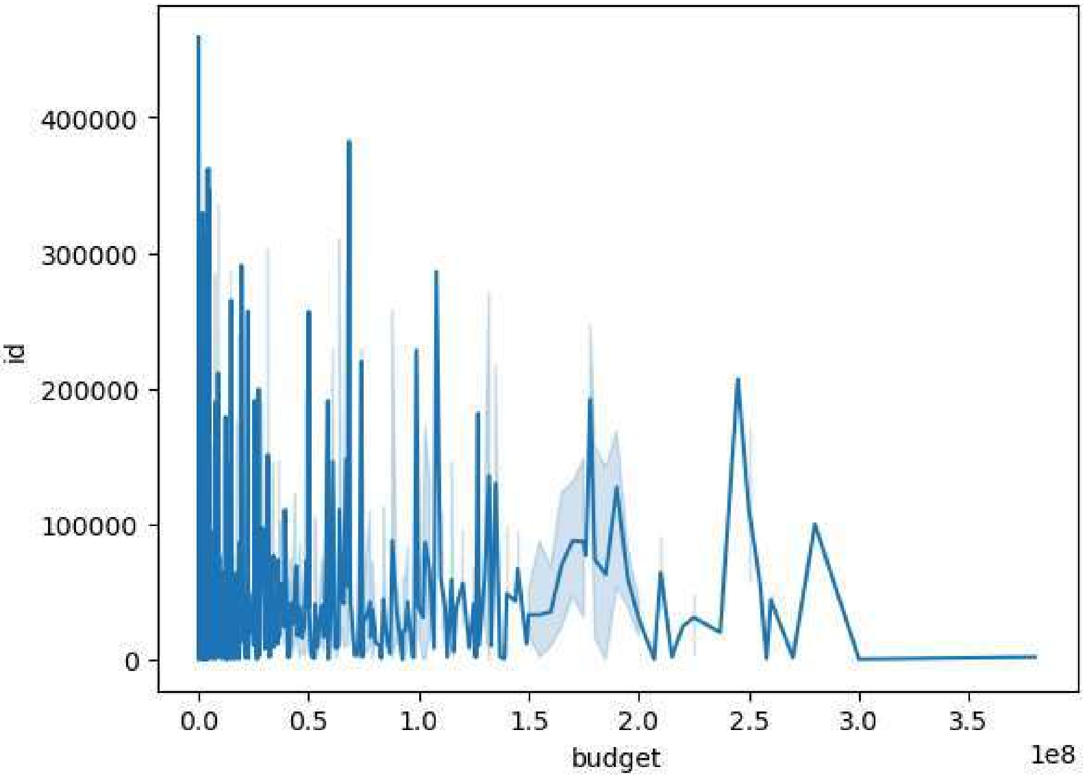
Out[20]:
```
index                   True
budget                  True
genres                  True
homepage                True
id                      True
keywords                True
original_language       True
original_title          True
overview                True
popularity              True
production_companies    True
production_countries    True
release_date            True
revenue                 True
runtime                 True
spoken_languages        True
status                  True
tagline                 True
title                   True
vote_average            True
vote_count              True
cast                    True
crew                    True
director                True
dtype: bool
```

In [21]:
```
mr = df1.get(40)
print(mr)
```

```
None
```

In [27]:
```
import seaborn as sea
sea.lineplot(x="budget", y="id", data=df1)
```

Out[27]:
```
<Axes: xlabel='budget', ylabel='id'>
```


400000
300000
200000
100000
0
id
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
budget
1e8

In [30]:

```
df1.max
```

```
1     [{'name': 'Dariusz Wolski', 'gender': 2, 'depa...     Gore Verbinski
2     [{'name': 'Thomas Newman', 'gender': 2, 'depar...         Sam Mendes
3     [{'name': 'Hans Zimmer', 'gender': 2, 'departm...  Christopher Nolan
4     [{'name': 'Andrew Stanton', 'gender': 2, 'depa...     Andrew Stanton
...                                                 ...                ...
4798  [{'name': 'Robert Rodriguez', 'gender': 0, 'de...   Robert Rodriguez
4799  [{'name': 'Edward Burns', 'gender': 2, 'depart...       Edward Burns
4800  [{'name': 'Carla Hetland', 'gender': 0, 'depar...        Scott Smith
4801  [{'name': 'Daniel Hsia', 'gender': 2, 'departm...        Daniel Hsia
4802  [{'name': 'Clark Peterson', 'gender': 2, 'depa...   Brian Herzlinger

[4803 rows x 24 columns]>
```

In [31]:

```
df1.min
```

Out[32]:

	index	budget	genres	homepage	id	keywords	o
0	0	0.0	Drama	http://www.missionimpossible.com/	5	independent film	
1	1	NaN	NaN	http://www.thehungergames.movie/	11	NaN	
2	2	NaN	NaN	NaN	12	NaN	
3	3	NaN	NaN	NaN	13	NaN	
4	4	NaN	NaN	NaN	14	NaN	
...	...	...	...	...	...	...	
4798	4798	NaN	NaN	NaN	426067	NaN	
4799	4799	NaN	NaN	NaN	426469	NaN	
4800	4800	NaN	NaN	NaN	433715	NaN	
4801	4801	NaN	NaN	NaN	447027	NaN	
4802	4802	NaN	NaN	NaN	459488	NaN	

4803 rows × 24 columns

	index	budget	genres	homepage	
1	1	300000000	Adventure Fantasy Action	http://disney.go.com/disneypictures/pirates/	2
17	17	380000000	Adventure Action Fantasy	http://disney.go.com/pirates/index-on-stranger...	18

4803 rows × 24 columns

```
In [36]: df1.iloc[5]
```

```
Out[36]: index                                                                   5
         budget                                                          258000000
         genres                                            Fantasy Action Adventure
         homepage                      http://www.sonypictures.com/movies/spider-man3/
         id                                                                    559
         keywords                    dual identity amnesia sandstorm love of one's ...
         original_language                                                      en
         original_title                                               Spider-Man 3
         overview                    The seemingly invincible Spider-Man goes up ag...
         popularity                                                     115.699814
         production_companies        [{"name": "Columbia Pictures", "id": 5}, {"nam...
         production_countries        [{"iso_3166_1": "US", "name": "United States o...
         release_date                                                   2007-05-01
         revenue                                                         890871626
         runtime                                                             139.0
         spoken_languages            [{"iso_639_1": "en", "name": "English"}, {"iso...
         status                                                           Released
         tagline                                                 The battle within.
         title                                                        Spider-Man 3
         vote_average                                                          5.9
         vote_count                                                           3576
         cast                        Tobey Maguire Kirsten Dunst James Franco Thoma...
         crew                        [{'name': 'Francine Maisler', 'gender': 1, 'de...
         director                                                        Sam Raimi
         Name: 5, dtype: object
```

```
In [37]: df1[0:3]
```

Out[37]:

	index	budget	genres	homepage	id	key
0	0	237000000	Action Adventure Fantasy Science Fiction	http://www.avatarmovie.com/	19995	(spa
1	1	300000000	Adventure Fantasy Action	http://disney.go.com/disneypictures/pirates/	285	eas
2	2	245000000	Action Adventure Crime	http://www.sonypictures.com/movies/spectre/	206647	spy or

3 rows × 24 columns

```
In [40]: df1.loc[:, ["budget","id"]]
```

Out[40]:

	budget	id
0	237000000	19995
1	300000000	285
2	245000000	206647
3	250000000	49026
4	260000000	49529
...	...	...
4798	220000	9367
4799	9000	72766
4800	0	231617
4801	0	126186
4802	0	25975

4803 rows × 2 columns

```
In [41]: df1.iloc[:30, :]
```

Out[46]:

	index	genres
1	1	Adventure Fantasy Action
2	2	Action Adventure Crime
4	4	Action Adventure Science Fiction

In [47]:
```
df1.iloc[1:3, :]
```

Out[47]:

	index	budget	genres	homepage	id	key
1	1	300000000	Adventure Fantasy Action	http://disney.go.com/disneypictures/pirates/	285	eas
2	2	245000000	Action Adventure Crime	http://www.sonypictures.com/movies/spectre/	206647	spy or

2 rows × 24 columns

In [48]:
```
df1[df1.columns[2:4]].iloc[5:10]
```

Out[48]:

	genres	homepage
5	Fantasy Action Adventure	http://www.sonypictures.com/movies/spider-man3/
6	Animation Family	http://disney.go.com/disneypictures/tangled/
7	Action Adventure Science Fiction	http://marvel.com/movies/movie/193/avengers_ag...
8	Adventure Fantasy Family	http://harrypotter.warnerbros.com/harrypottera...
9	Action Adventure Fantasy	http://www.batmanvsupermandawnofjustice.com/

In [49]:
```
df1.isnull()
```

Out[49]:

	index	budget	genres	homepage	id	keywords	original_language	original_t
0	False	False	False	False	False	False	False	Fa
1	False	False	False	False	False	False	False	Fa
2	False	False	False	False	False	False	False	Fa
3	False	False	False	False	False	False	False	Fa
4	False	False	False	False	False	False	False	Fa
...	...	...	...	...	...	...	...	
4798	False	False	False	True	False	False	False	Fa
4799	False	False	False	True	False	True	False	Fa
4800	False	False	False	False	False	False	False	Fa
4801	False	False	True	False	False	True	False	Fa
4802	False	False	False	True	False	False	False	Fa

4803 rows × 24 columns

In [50]:
```
df1.isnull().any()
```

Out[50]:
```
index                   False
budget                  False
genres                   True
homepage                 True
id                      False
keywords                 True
original_language       False
original_title          False
overview                 True
popularity              False
production_companies    False
production_countries    False
release_date             True
revenue                 False
runtime                  True
spoken_languages        False
status                  False
tagline                  True
title                   False
vote_average            False
vote_count              False
cast                     True
crew                    False
director                 True
dtype: bool
```

In [51]:
```
df1.isnull().sum().sum()
```

Out[51]:
```
4454
```

In [52]:
```
df1.isnull().sum()
```

```
Out[52]: index                     0
         budget                    0
         genres                   28
         homepage               3091
         id                        0
         keywords                412
         original_language         0
         original_title            0
         overview                  3
         popularity                0
         production_companies      0
         production_countries      0
         release_date              1
         revenue                   0
         runtime                   2
         spoken_languages          0
         status                    0
         tagline                 844
         title                     0
         vote_average              0
         vote_count                0
         cast                     43
         crew                      0
         director                 30
         dtype: int64
```

```
In [53]: df1.isnull().sum(axis=1)
```

```
Out[53]: 0       0
         1       0
         2       0
         3       0
         4       0
                ..
         4798    1
         4799    2
         4800    1
         4801    2
         4802    2
         Length: 4803, dtype: int64
```

```
In [54]: df1.isna().sum()
```

```
Out[54]: index                    0
         budget                   0
         genres                  28
         homepage              3091
         id                       0
         keywords               412
         original_language        0
         original_title           0
         overview                 3
         popularity               0
         production_companies     0
         production_countries     0
         release_date             1
         revenue                  0
         runtime                  2
         spoken_languages         0
         status                   0
         tagline                844
         title                    0
         vote_average             0
         vote_count               0
         cast                    43
         crew                     0
         director                30
         dtype: int64
```

```
In [55]: df1.groupby(['budget'])['id'].apply(lambda x:x.isnull().sum())
```

```
Out[55]: budget
         0            0
         1            0
         2            0
         3            0
         4            0
                     ..
         260000000    0
         270000000    0
         280000000    0
         300000000    0
         380000000    0
         Name: id, Length: 436, dtype: int64
```

```
In [56]: df1.dtypes
```

Out[56]:
```
index                     int64
budget                    int64
genres                   object
homepage                 object
id                        int64
keywords                 object
original_language        object
original_title           object
overview                 object
popularity              float64
production_companies     object
production_countries     object
release_date             object
revenue                   int64
runtime                 float64
spoken_languages         object
status                   object
tagline                  object
title                    object
vote_average            float64
vote_count                int64
cast                     object
crew                     object
director                 object
dtype: object
```

In [57]:
```
df1['budget']= df1['budget'].astype("int")
df1['budget']
```

Out[57]:
```
0       237000000
1       300000000
2       245000000
3       250000000
4       260000000
          ...
4798       220000
4799         9000
4800            0
4801            0
4802            0
Name: budget, Length: 4803, dtype: int32
```

In [66]:
```
df1['genres'].unique()
```

Out[66]:
```
array(['Action Adventure Fantasy Science Fiction',
       'Adventure Fantasy Action', 'Action Adventure Crime', ...,
       'Thriller Horror Comedy', 'Foreign Thriller',
       'Comedy Drama Romance TV Movie'], dtype=object)
```

In [67]:
```
label_encoder = preprocessing.LabelEncoder()
```

In [70]:
```
df1['genres']= label_encoder.fit_transform(df1['genres'])
```

In [71]:
```
df1['genres'].unique()
```

Out[71]:
```
array([  59,  327,   29, ..., 1122,  878,  477])
```

In [8]:
```
from sklearn import preprocessing
features_df=df1.drop(columns=['genres'])
```

	index	budget	m score	budget	genres	
4796	4796	7000.0	7000.0	7000	Science Fiction Drama Thriller	http://www.p

1432 rows × 26 columns

In [30]:

```
import numpy as np
import matplotlib.pyplot as plt
print(np.where(df1['index']>90))
print(np.where(df1['budget']<25))
print(np.where(df1['id']<30))
```

```
(array([  91,   92,   93, ..., 4800, 4801, 4802], dtype=int64),)
(array([ 265,  265,  321, ..., 4801, 4802, 4802], dtype=int64), array([0, 1, 0,
..., 1, 0, 1], dtype=int64))
(array([ 199,  322,  328,  557,  809,  828, 1525, 2516, 2638, 2799, 2912,
       3766, 4023], dtype=int64),)
```

In [33]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from scipy import stats
```

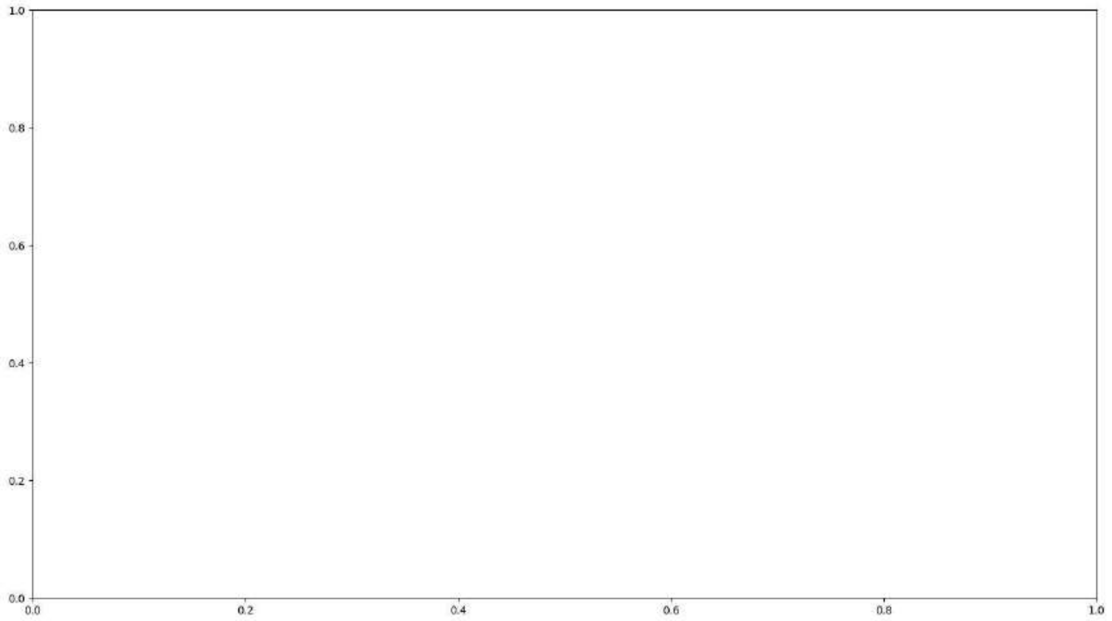
In [36]:

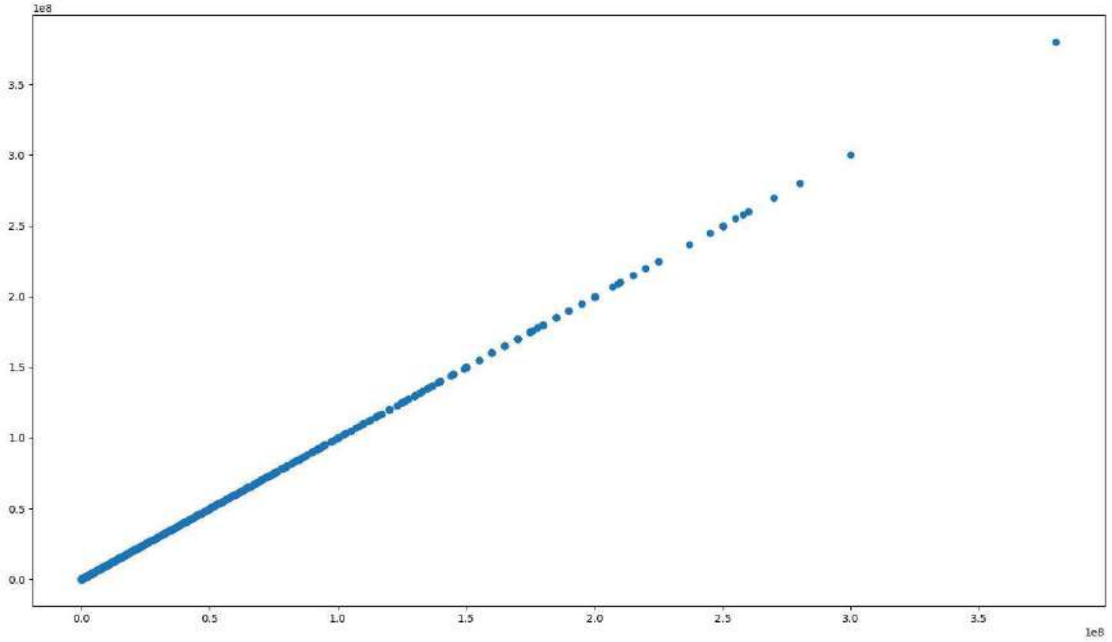
```
z = np.abs(stats.zscore(df1['budget']))
print(z)
```

```
        budget    budget
0     2.073467  5.107181
1     2.073467  6.654402
2     2.073467  5.303653
3     2.073467  5.426449
4     2.073467  5.672039
...        ...       ...
4798  0.897317  0.707916
4799  0.905174  0.713098
4800  0.905509  0.713319
4801  0.905509  0.713319
4802  0.905509  0.713319

[4803 rows x 2 columns]
```

In [41]:

```
ig, ax = plt.subplots(figsize = (18,10))
ax.scatter(df1['budget'], df1['budget'])
plt.show()

ax.set_xlabel('(Proportion non-retail business acres)/(town)')
ax.set_ylabel('(Full-value property-tax rate)/($10,000)')
```





```
Out[41]:  Text(4.444444444444452, 0.5, '(Full-value property-tax rate)/($10,000)')
```

```
In [43]:  threshold = 0.18
          sample_outliers = np.where(z <threshold)
          sample_outliers
```

```
Out[43]:  (array([  83,   83,  379, ..., 4036, 4039, 4586], dtype=int64),
           array([0, 1, 0, ..., 1, 1, 1], dtype=int64))
```

```
In [44]:  sorted_rscore= sorted(df1['budget'])
```

```
In [45]:  sorted_rscore
```

```
Out[45]:  ['budget', 'budget']
```

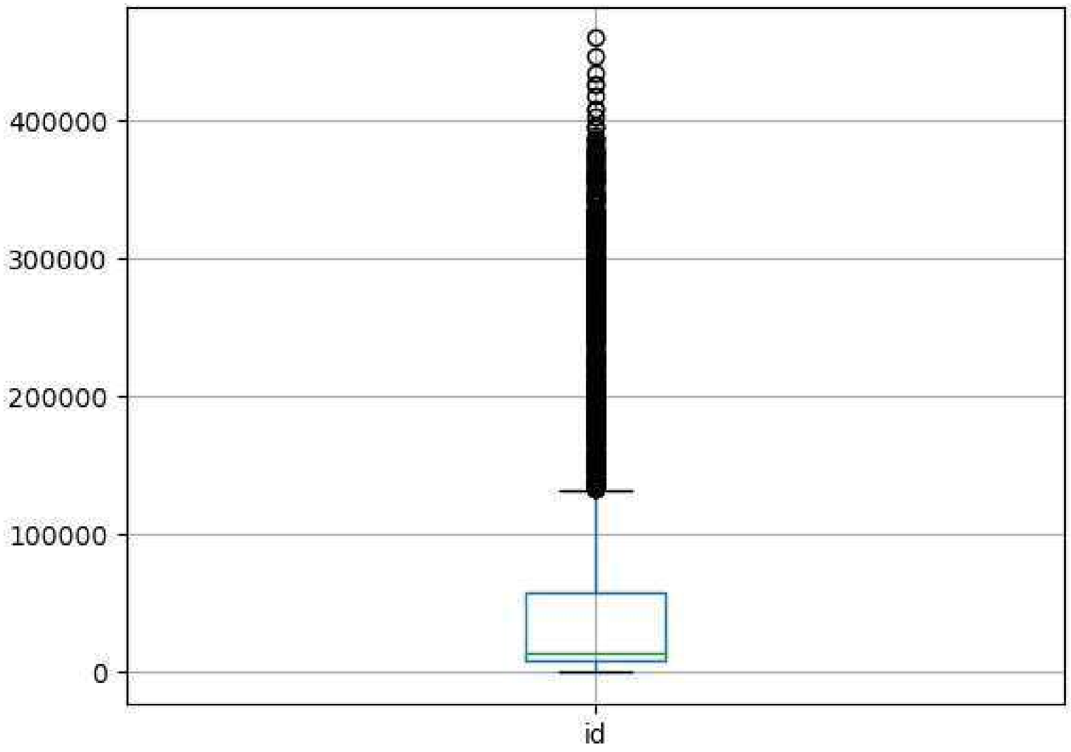
```
In [49]:  IQR = q3-q1
          lwr_bound = q1-(1.5*IQR)
```

index	budget	m score	budget	genres

4803 rows × 26 columns

In [12]:

```
col = ['id']
df1.boxplot(col)
median=np.median(sorted_rscore)
median
refined_df1=df1
```



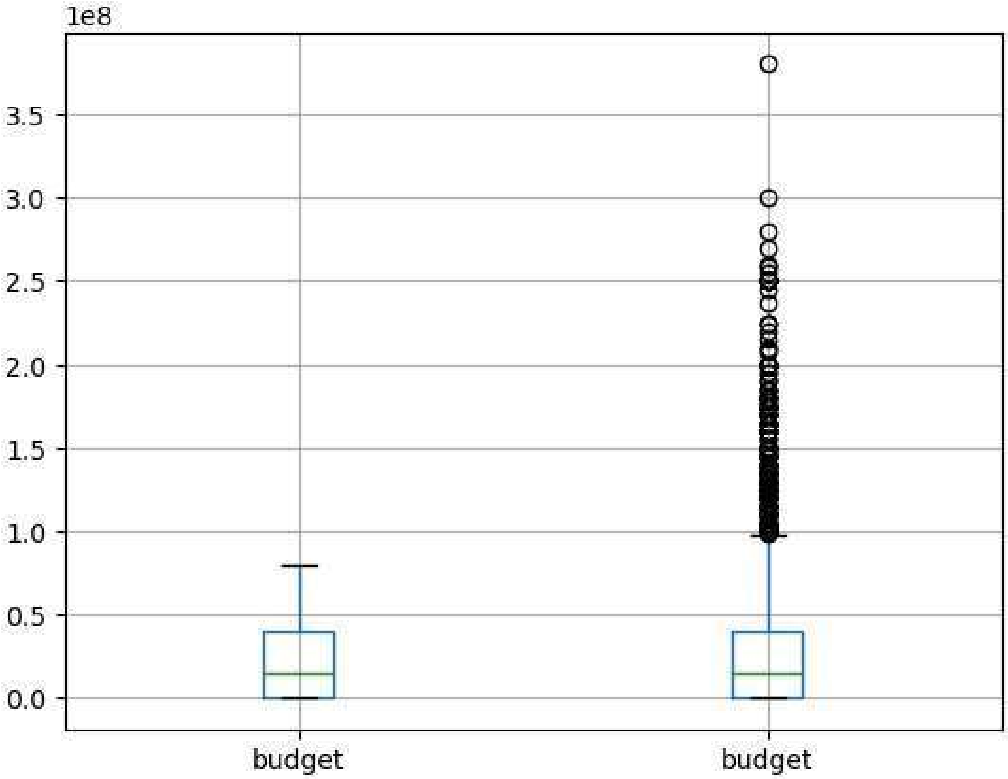
In [13]:

```
refined_df1['id'] = np.where(refined_df1['id'] <lwr_bound, median,refined_df1['i
refined_df1
```

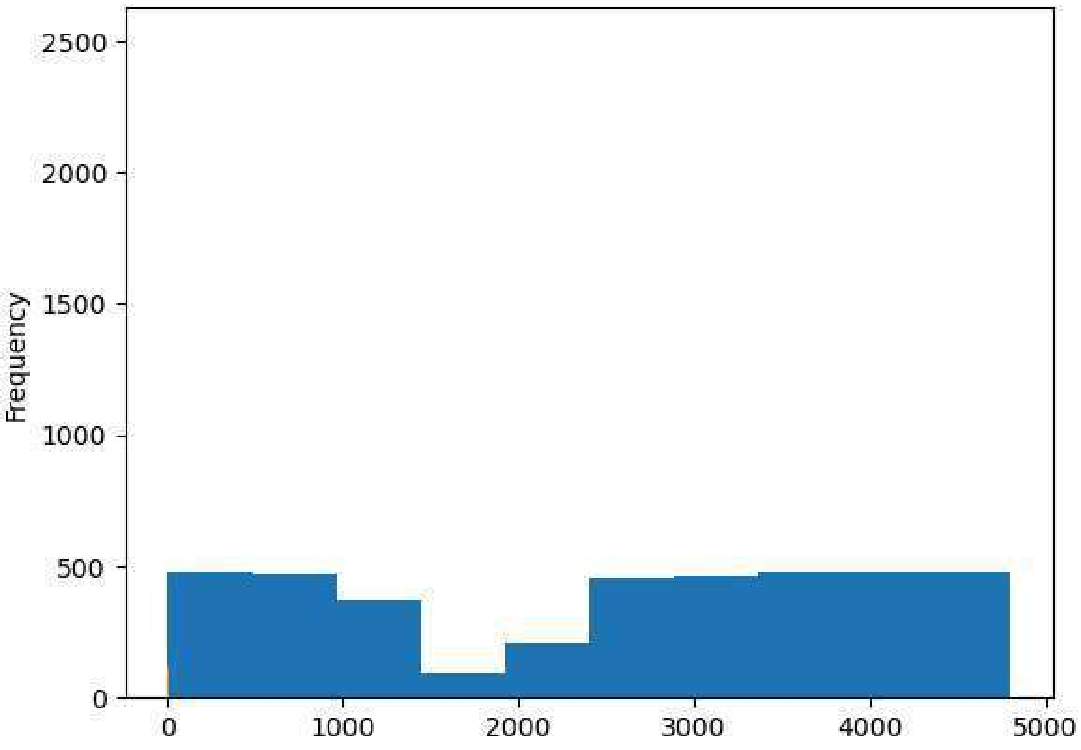
index	budget	m score	budget	genres

4803 rows × 26 columns

```
In [18]: col = ['budget']
         refined_df1.boxplot(col)
         plt.show()
```



```
In [52]: import matplotlib.pyplot as plt
         new_df['index'].plot(kind = 'hist')
         df1['log_math'] = np.log10(df1['index'])
         df1['log_math'].plot(kind = 'hist')
         plt.show()
```



```
In [16]: x=np.array([95,85,80,70,60])
         y=np.array([85,95,70,65,70])
```

```
In [17]: model= np.polyfit(x, y, 1)
         model
```

```
Out[17]: array([ 0.64383562, 26.78082192])
```

```
In [18]: predict = np.poly1d(model)
         predict(65)
```

```
Out[18]: 68.63013698630137
```

```
In [19]: y_pred= predict(x)
         y_pred
```

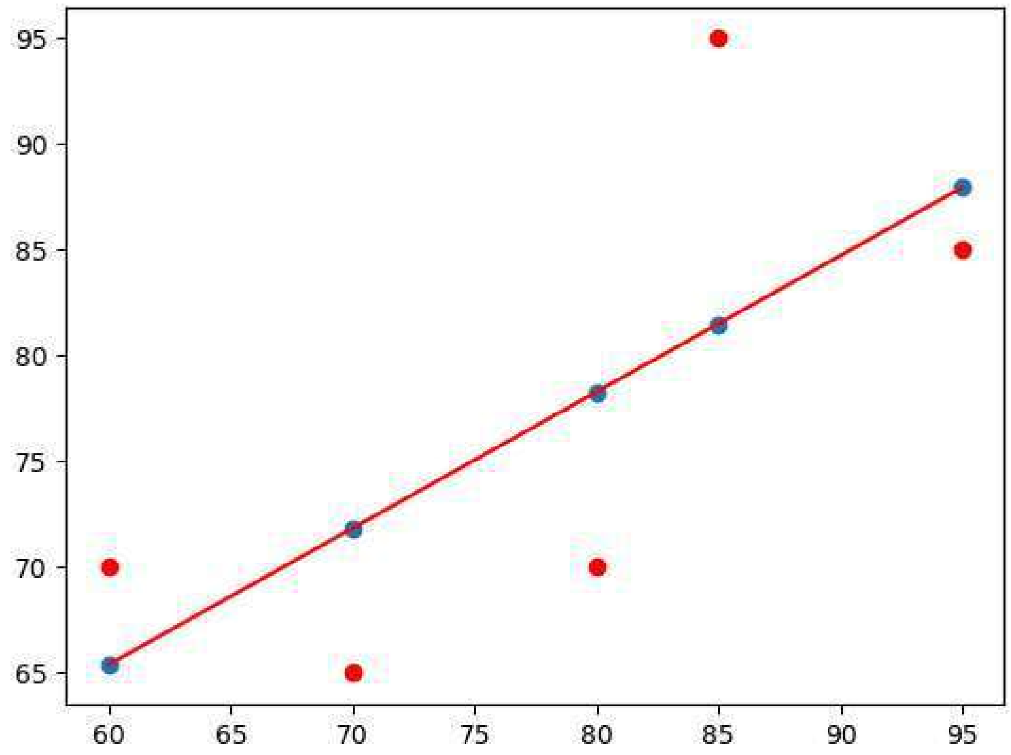
```
Out[19]: array([87.94520548, 81.50684932, 78.28767123, 71.84931507, 65.4109589 ])
```

```
In [20]: from sklearn.metrics import r2_score
         r2_score(y, y_pred)
```

```
Out[20]: 0.4803218090889326
```

```
In [21]: y_line = model[1] + model[0]* x
         plt.plot(x, y_line, c = 'r')
         plt.scatter(x, y_pred)
         plt.scatter(x,y,c='r')
```

```
Out[21]: <matplotlib.collections.PathCollection at 0x15f1de62850>
```



In [25]:

```
x = df1.drop(['budget'], axis = 1)
y = df1['budget']
```

In [34]:

```
print(df1.isnull().sum())
```

```
index                      0
budget                     0
genres                    28
homepage                3091
id                         0
keywords                 412
original_language          0
original_title             0
overview                   3
popularity                 0
production_companies       0
production_countries       0
release_date               1
revenue                    0
runtime                    2
spoken_languages           0
status                     0
tagline                  844
title                      0
vote_average               0
vote_count                 0
cast                      43
crew                       0
director                  30
dtype: int64
```

In [36]:

```
X = df1.iloc[:,0:13]
y = df1.iloc[:,-1]
```

```
X
y
```

Out[36]:

```
0           James Cameron
1          Gore Verbinski
2              Sam Mendes
3       Christopher Nolan
4          Andrew Stanton
               ...
4798     Robert Rodriguez
4799          Edward Burns
4800           Scott Smith
4801           Daniel Hsia
4802     Brian Herzlinger
Name: director, Length: 4803, dtype: object
```

In [37]:

```
df1["tagline"].value_counts(normalize=True)
```

Out[37]:

```
tagline
Based on a true story.                             0.000758
From zero to hero.                                 0.000505
The only way out is down.                          0.000505
Be careful what you wish for.                      0.000505
What could go wrong?                               0.000505
                                                     ...
Life is Not Child-Proof.                           0.000253
Every war has a beginning.                         0.000253
First came love... then came Reverend Frank.       0.000253
Get off the bench and get into the game.           0.000253
A New Yorker in Shanghai                           0.000253
Name: proportion, Length: 3944, dtype: float64
```

In [38]:

```
x=df1.drop(["tagline"],axis=1)
y=df1["tagline"]
```

In [39]:

```
x
```

	index	budget	genres	homepage
4802	4802	0	Documentary	NaN

4803 rows × 23 columns

In [40]:

```
y
```

Out[40]:

```
0                                    Enter the World of Pandora.
1                 At the end of the world, the adventure begins.
2                                              A Plan No One Escapes
3                                                    The Legend Ends
4                              Lost in our world, found in another.
                                 ...
4798    He didn't come looking for trouble, but troubl...
4799    A newlywed couple's honeymoon is upended by th...
4800                                                             NaN
4801                                         A New Yorker in Shanghai
4802                                                             NaN
Name: tagline, Length: 4803, dtype: object
```

In [41]:

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.20,random_
```

In [42]:

```
print(X_train.shape)
print(X_test.shape)
print(y_train.shape)
print(y_test.shape)
```

```
(3842, 13)
(961, 13)
(3842,)
(961,)
```

In [43]:

```
from sklearn.preprocessing import MinMaxScaler
scaler=MinMaxScaler()
scaler
```

Out[43]:

```
▾ MinMaxScaler
MinMaxScaler()
```

In [44]:

```
from sklearn.datasets import make_classification
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.pipeline import make_pipeline
from sklearn.preprocessing import StandardScaler

X, y = make_classification(random_state=42)
X_train, X_test, y_train, y_test = train_test_split(X, y, random_state=42)
pipe = make_pipeline(StandardScaler(), LogisticRegression())
pipe.fit(X_train, y_train)
```

Out[44]:

```
▸ Pipeline
  ▸ StandardScaler
  ▸ LogisticRegression
```

In [45]:

```python
from sklearn.linear_model import LogisticRegression
logreg = LogisticRegression()
logreg.fit(X_train,y_train)
```

Out[45]:

```
▾ LogisticRegression
LogisticRegression()
```

In [46]:

```python
y_pred=logreg.predict(X_test)
```

In [47]:

```python
from sklearn.linear_model import LinearRegression
```

In [48]:

```python
from sklearn.preprocessing import StandardScaler
from sklearn.pipeline import make_pipeline
model = make_pipeline(StandardScaler(with_mean=False), LinearRegression())
model.fit(X_train, y_train)
```

Out[48]:

```
▸ Pipeline
  ▸ StandardScaler
  ▸ LinearRegression
```

In [49]:

```python
model.score(X_test,y_test)
```

Out[49]:

```
0.6803193862233878
```

In [50]:

```python
X_train
```

Out[50]:

```
array([[-1.06239353, -2.68317954,  0.33848384, ..., -0.35316629,
         0.32579632,  0.1943843 ],
       [-0.79047446, -0.07873421, -1.69246463, ...,  1.09419152,
        -0.12578692,  0.05572491],
       [-0.22096417, -0.54561186, -0.57117899, ...,  0.64084286,
        -0.28110029,  1.79768653],
       ...,
       [ 0.84064355,  0.37531604, -0.96697614, ...,  0.42545756,
         0.76041466,  0.78580016],
       [ 0.49403019,  0.63067073,  1.1487657 , ..., -2.84854262,
        -0.37061433,  0.77169871],
       [-0.42018682, -0.24038388,  0.9843224 , ..., -0.99835404,
         0.23421473,  1.55050049]])
```

In [51]:

```python
y_train
```

In [54]:

```
from sklearn.metrics import precision_score,ConfusionMatrixDisplay, confusion_ma
cm= confusion_matrix(y_test, y_pred)
disp = ConfusionMatrixDisplay(confusion_matrix = cm)
print("Confusion matrix :")
print(cm)
```

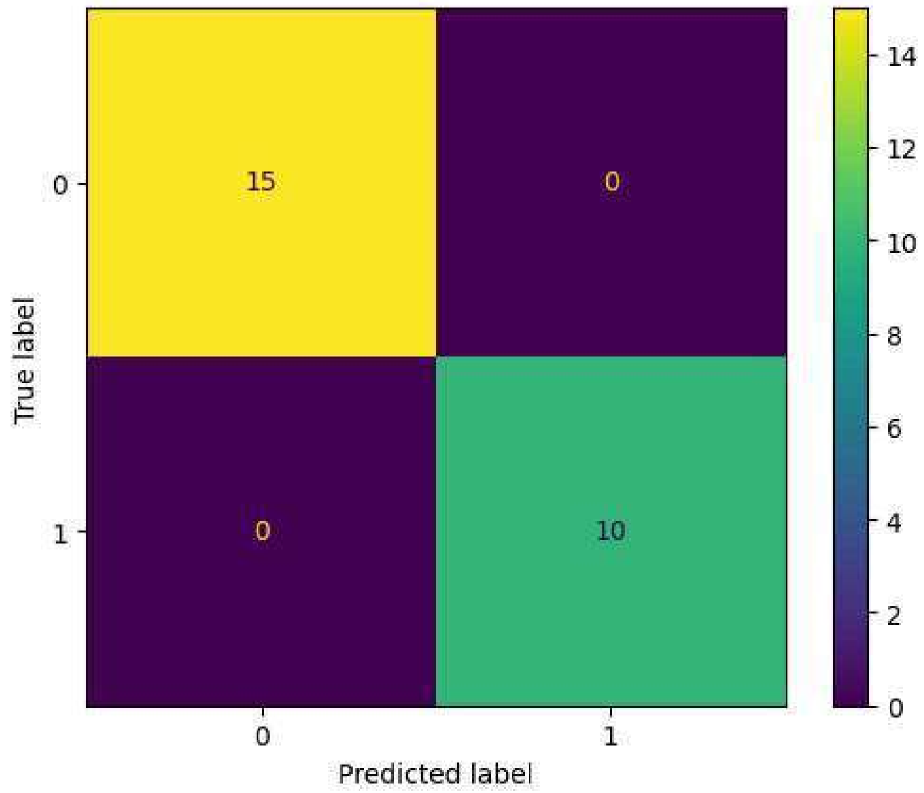
```
Confusion matrix :
[[15  0]
 [ 0 10]]
```

In [55]:

```
disp.plot()
```

Out[55]:

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x1236b1f11d0
>
```



In [56]:

```
true_negative =cm[0][0]
false_negative =cm[1][0]
false_positive =cm[0][1]
true_positive =cm[1][1]
```

In [57]:

```
Accuracy = (true_positive + true_negative) / (true_positive +false_positive + tr
Accuracy
# Precison
Precision = true_positive/(true_positive+false_positive)
Precision
# Recall
Recall = true_positive/(true_positive+false_negative)
Recall
# F1 Score
F1_Score = 2*(Recall * Precision) / (Recall + Precision)
F1_Score
```

Out[57]:

```
1.0
```

In [58]:
```
print("Accuracy:", Accuracy)
print("Confusion Matrix:")
print(cm)
print("\nClassification Report:")
print(classification_report(y_test, y_pred))
```

```
Accuracy: 1.0
Confusion Matrix:
[[15  0]
 [ 0 10]]

Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        15
           1       1.00      1.00      1.00        10

    accuracy                           1.00        25
   macro avg       1.00      1.00      1.00        25
weighted avg       1.00      1.00      1.00        25
```

In [59]:
```
Accuracy
```

Out[59]:
```
1.0
```

In [73]:
```
Precision
```

Out[73]:
```
1.0
```

In [74]:
```
Recall
```

Out[74]:
```
1.0
```

In [75]:
```
F1_Score
```

Out[75]:
```
1.0
```

In [76]:
```
from sklearn.metrics import f1_score, confusion_matrix, roc_auc_score, roc_curve
import matplotlib as plt
```

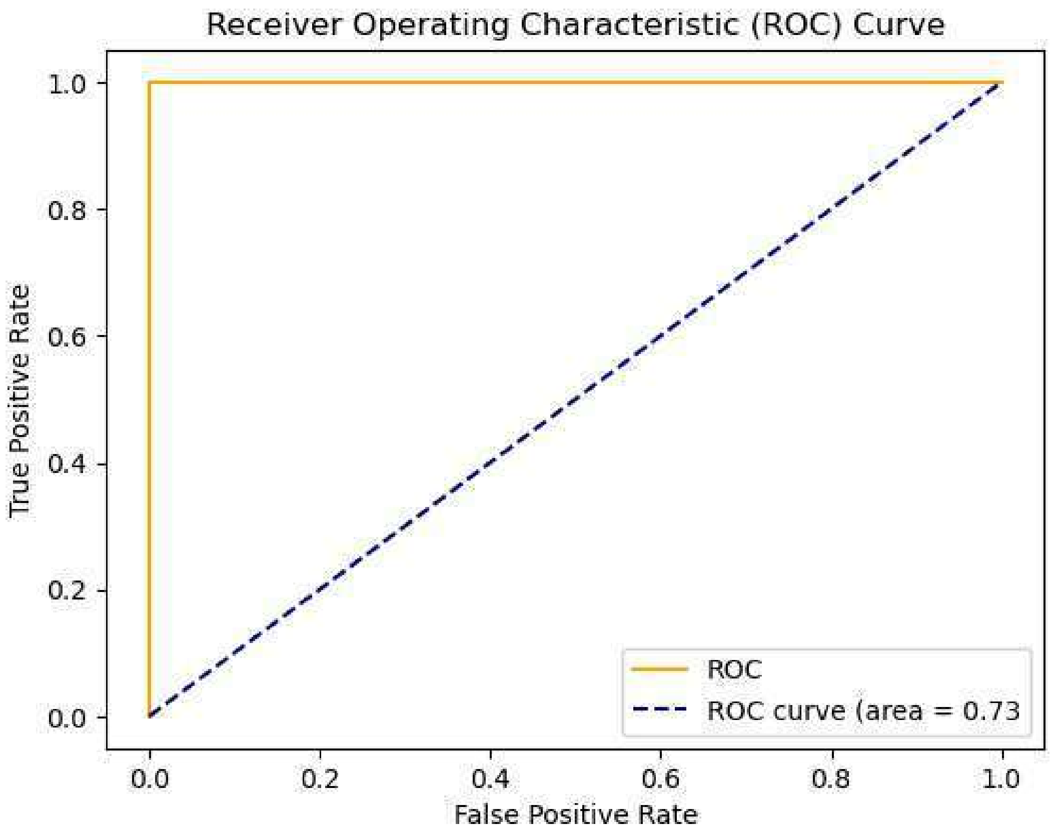
In [77]:
```
auc_score=roc_auc_score(y_test,y_pred)
```

In [78]:
```
fpr,tpr,threasholds=roc_curve(y_test,y_pred)
```

In [79]:
```
threasholds
```

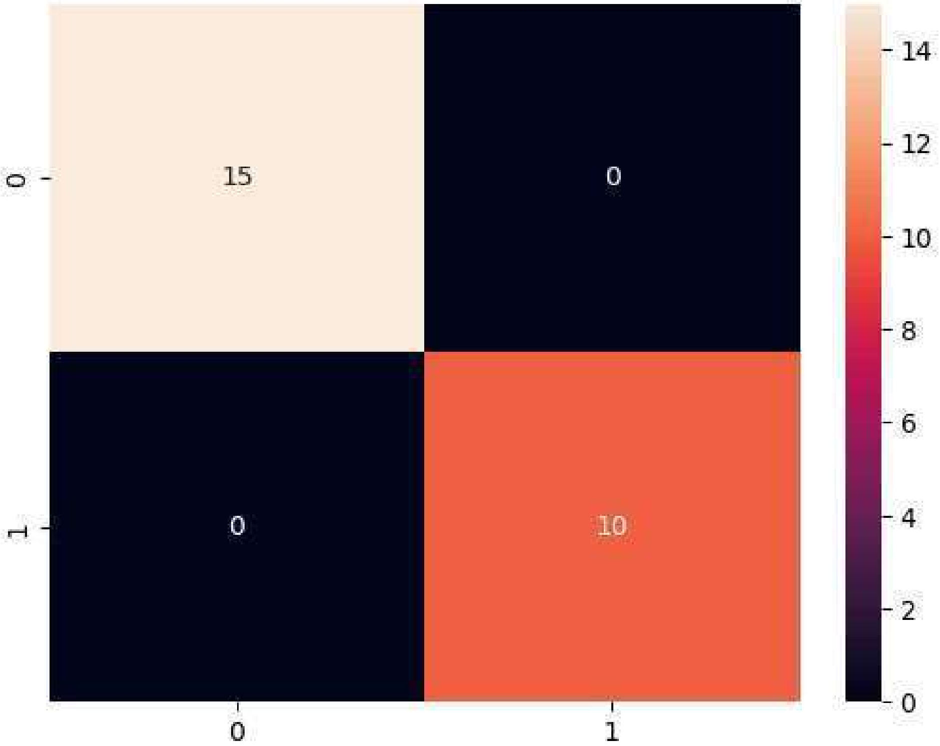
Out[79]:
```
array([inf,  1.,  0.])
```

In [80]:
```
import matplotlib.pyplot as plt
plt.plot(fpr, tpr, color='orange', label='ROC')
plt.plot([0, 1], [0, 1], color='darkblue', linestyle='--',label='ROC curve (area
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver Operating Characteristic (ROC) Curve')
plt.legend()
plt.show()
```

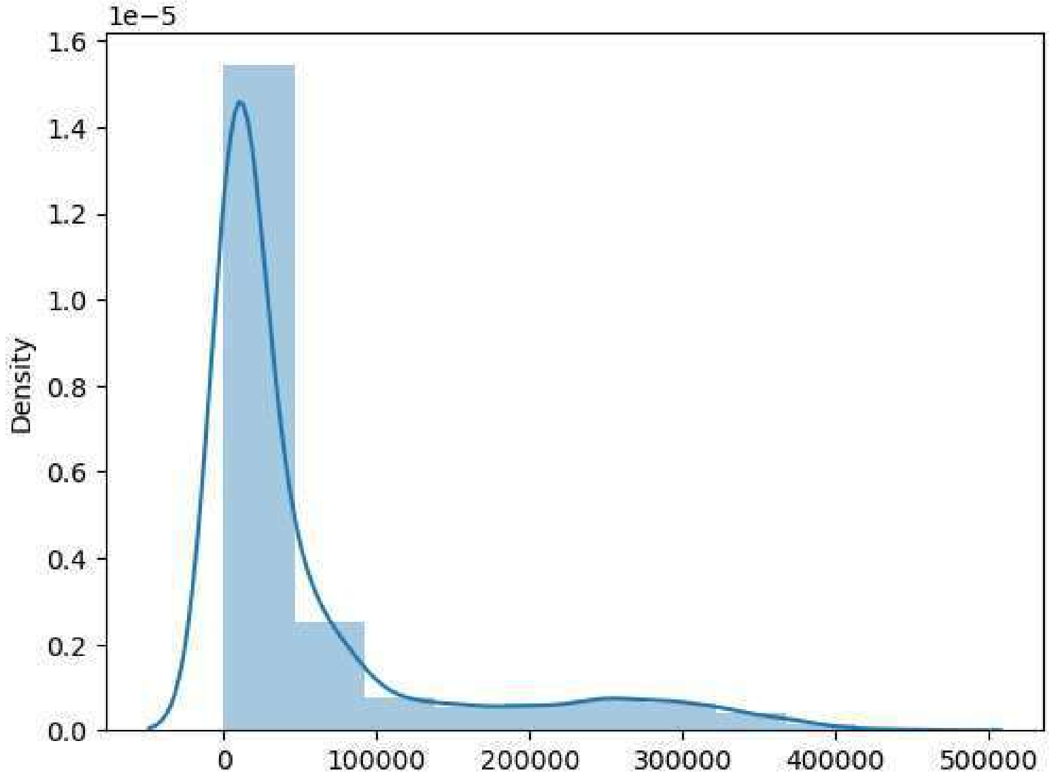


```
In [81]: import seaborn as sns
         sns.heatmap(cm, annot=True)
```

```
Out[81]: <Axes: >
```



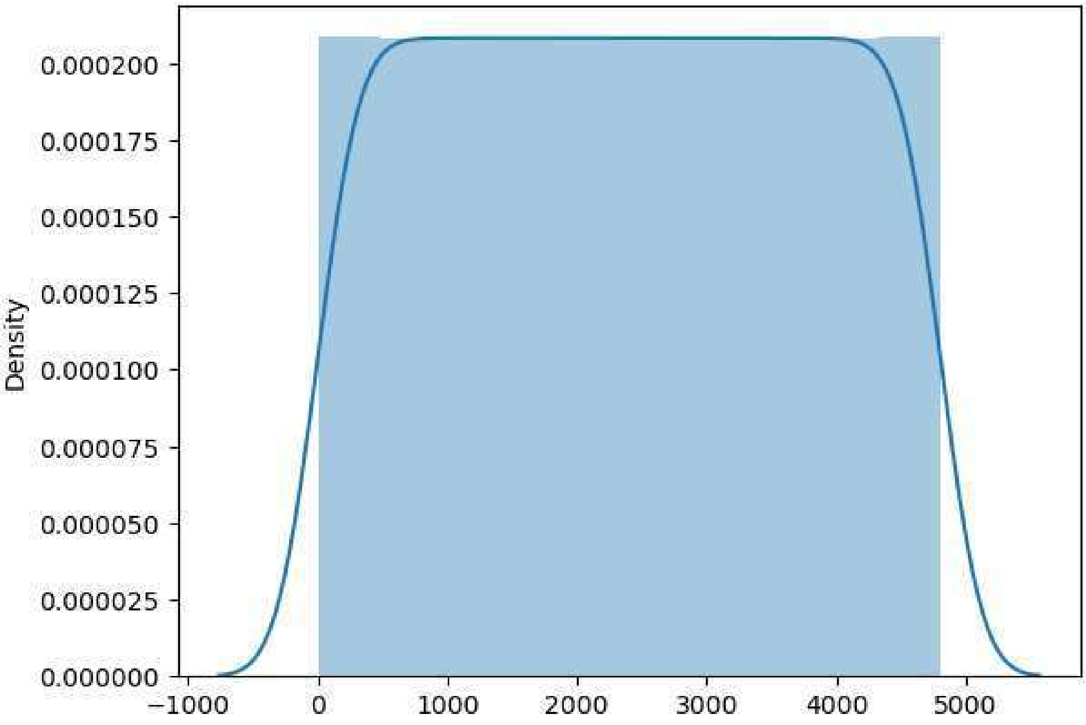
```
In [34]: from sklearn.naive_bayes import GaussianNB
         model = GaussianNB()
```



In [50]:
```
sns.distplot(x = df1['runtime'], bins = 10)
```

```
C:\Users\Welcome\AppData\Local\Temp\ipykernel_12292\1852036295.py:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(x = df1['runtime'], bins = 10)
C:\Users\Welcome\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119: FutureWarn
ing: use_inf_as_na option is deprecated and will be removed in a future version.
Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```

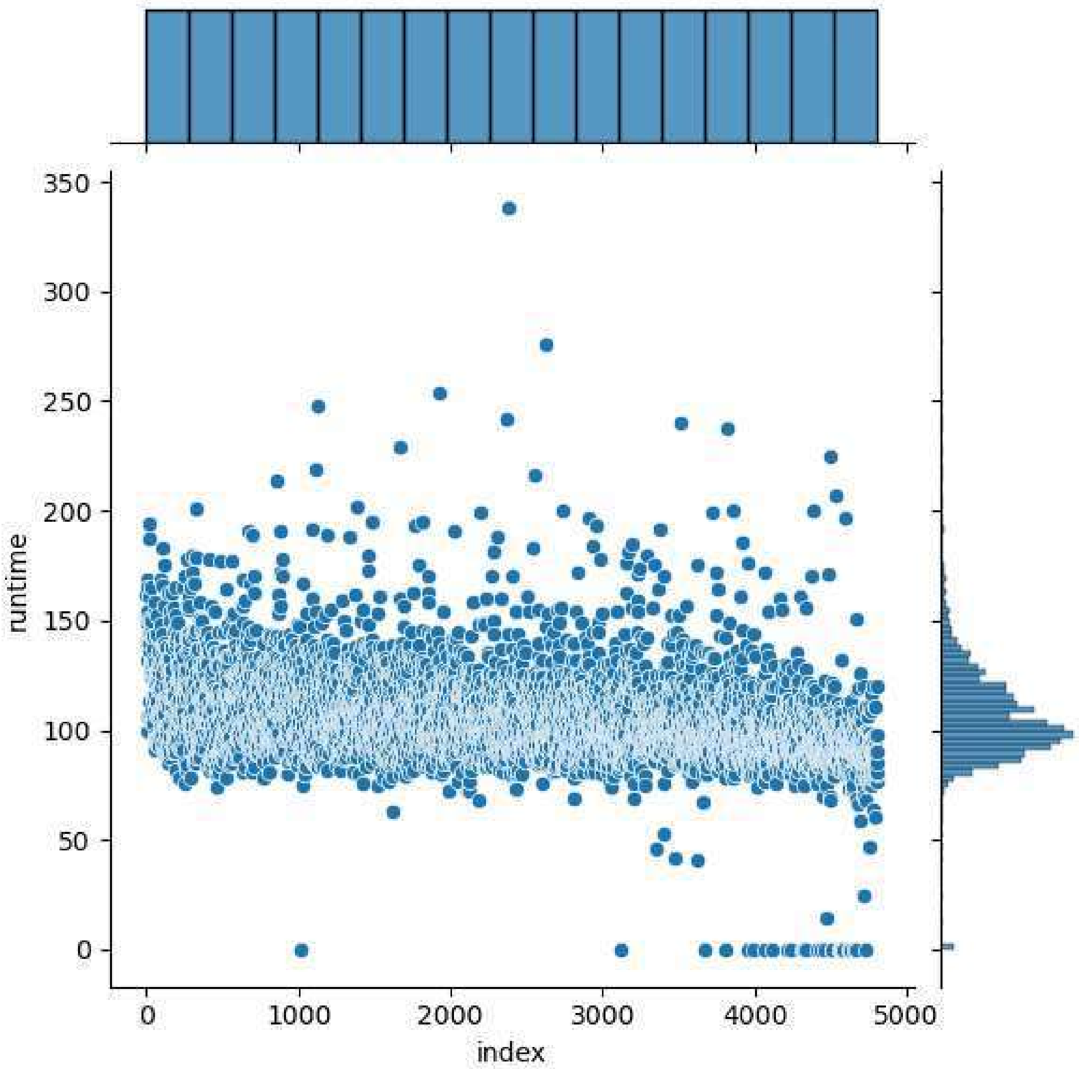
Out[50]:
```
<Axes: ylabel='Density'>
```

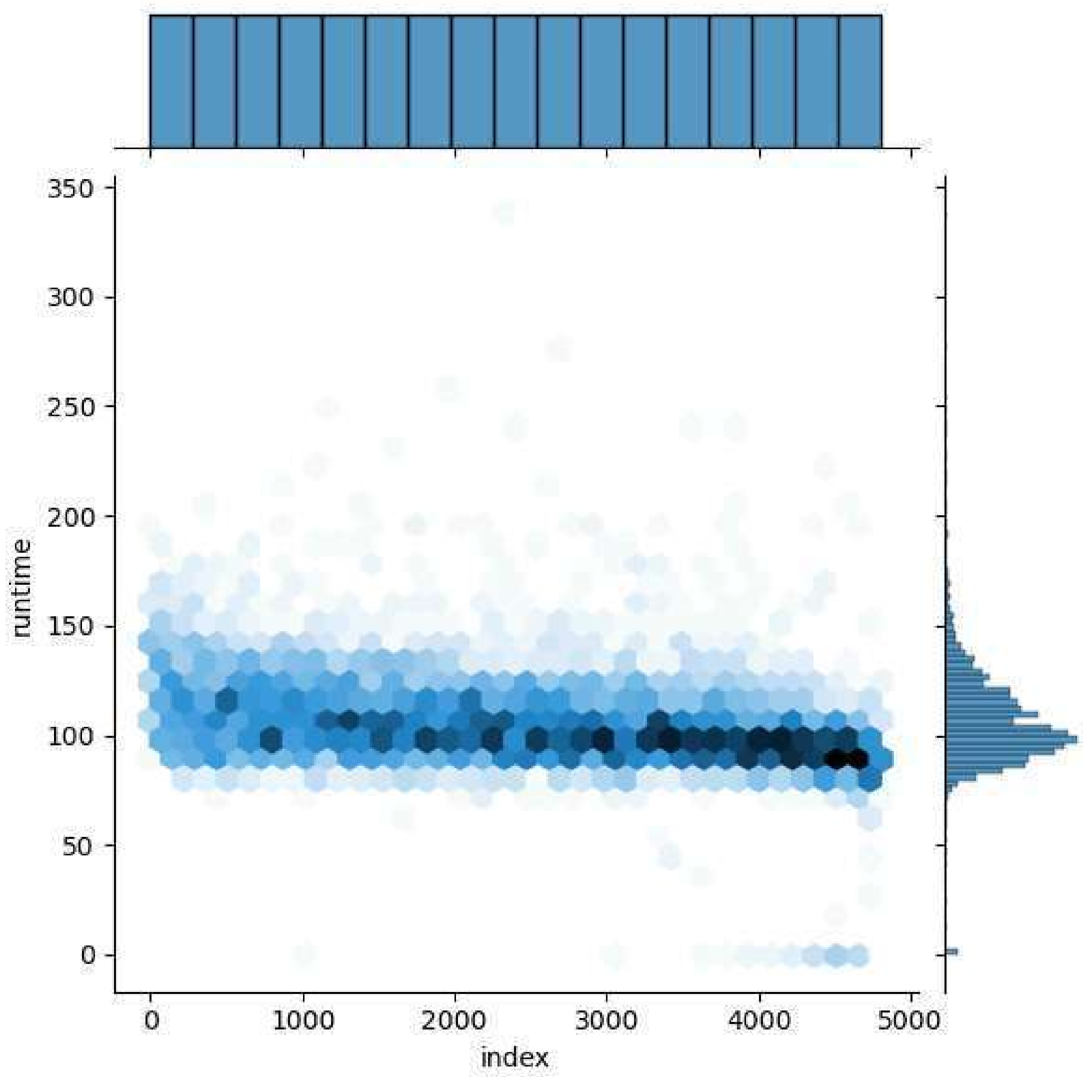


In [52]:
```
sns.jointplot(x = df1['index'], y = df1['runtime'], kind ='scatter')
sns.jointplot(x = df1['index'], y = df1['runtime'], kind = 'hex')
```

```
C:\Users\Welcome\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119: FutureWarn
ing: use_inf_as_na option is deprecated and will be removed in a future version.
Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\Users\Welcome\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119: FutureWarn
ing: use_inf_as_na option is deprecated and will be removed in a future version.
Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\Users\Welcome\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119: FutureWarn
ing: use_inf_as_na option is deprecated and will be removed in a future version.
Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\Users\Welcome\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119: FutureWarn
ing: use_inf_as_na option is deprecated and will be removed in a future version.
Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```

Out[52]:
```
<seaborn.axisgrid.JointGrid at 0x228f93116d0>
```

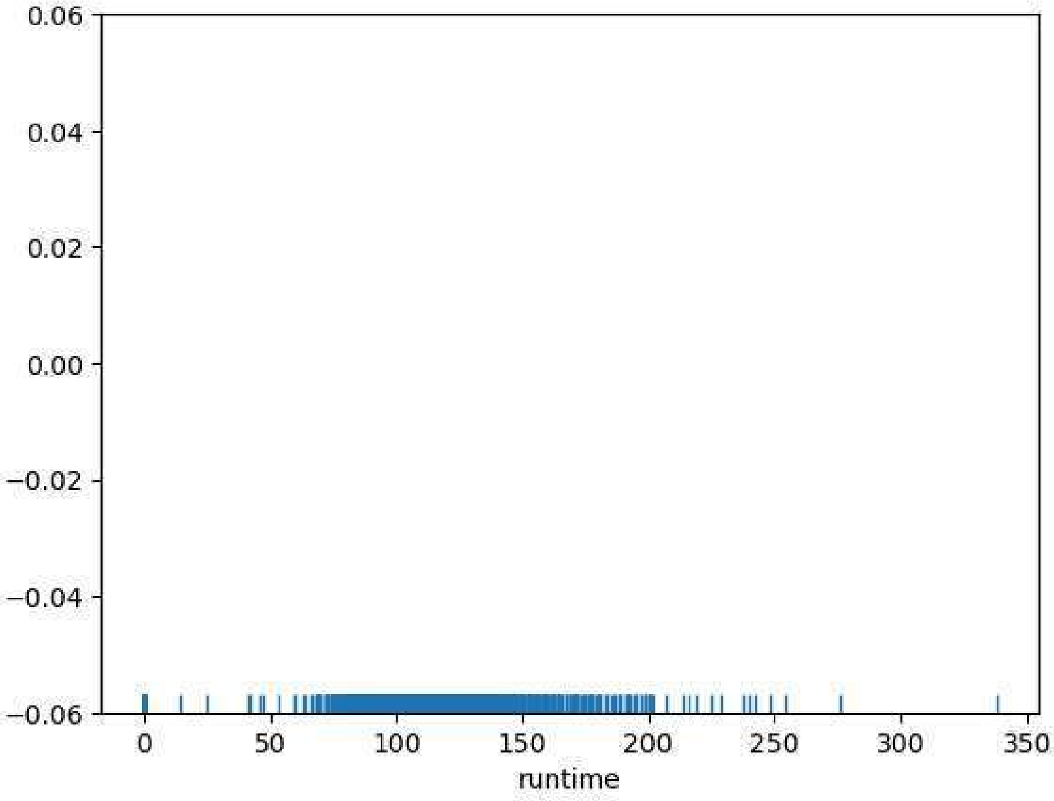

350
300
250
200
150
100
50
0
runtime
0
1000
2000
3000
4000
5000
index



```
In [53]: sns.rugplot(df1['budget'])
```

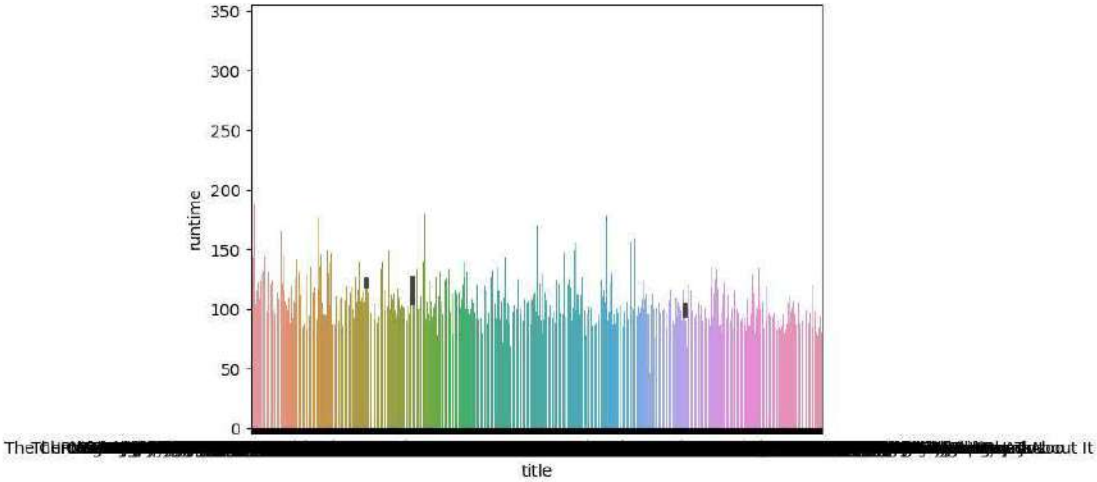
```
C:\Users\Welcome\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119: FutureWarn
ing: use_inf_as_na option is deprecated and will be removed in a future version.
Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```

```
Out[53]: <Axes: xlabel='budget'>
```



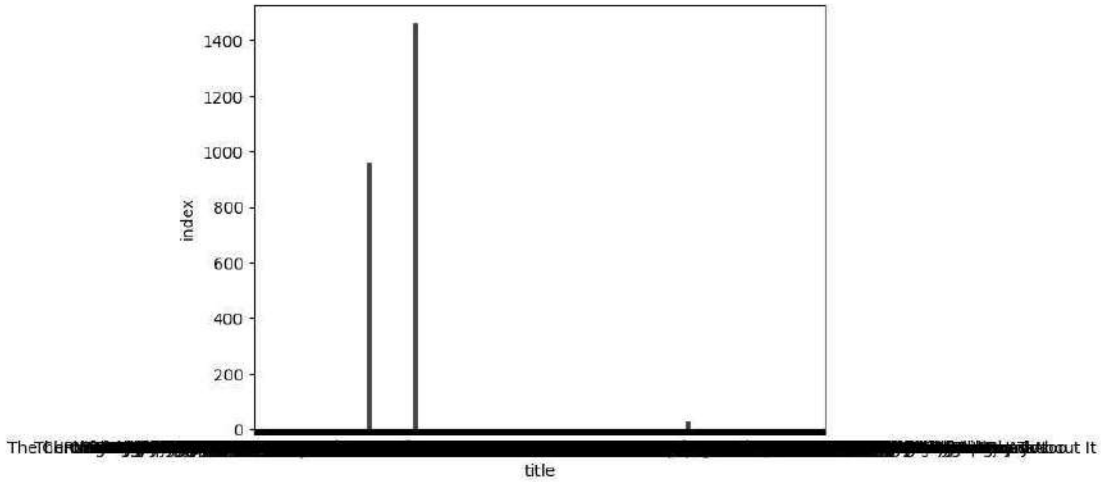
```
In [14]: sns.barplot(x='title', y='runtime', data=df1)
```

```
Out[14]: <Axes: xlabel='title', ylabel='runtime'>
```



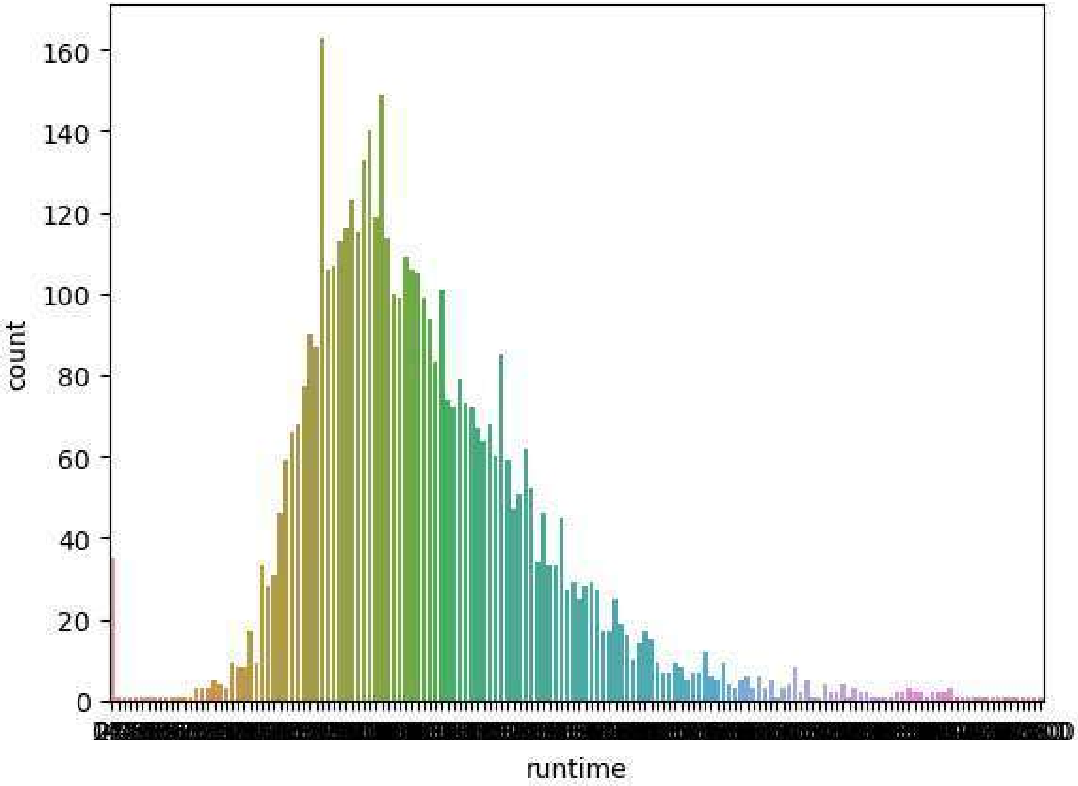
```
In [8]: import numpy as np
        import matplotlib.pyplot as plt
        import seaborn as sns
        sns.barplot(x='title', y='index', data=df1, estimator=np.std)
```

```
Out[8]: <Axes: xlabel='title', ylabel='index'>
```



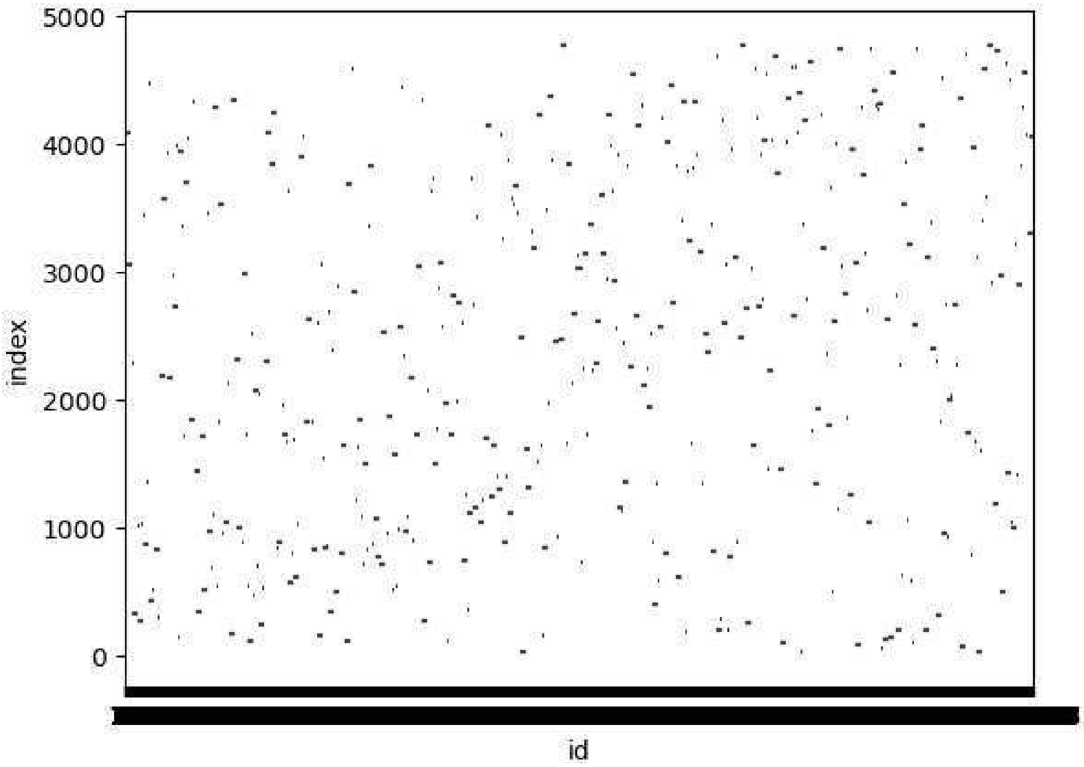
In [12]:
```
sns.countplot(x='runtime', data=df1)
```

Out[12]:
```
<Axes: xlabel='runtime', ylabel='count'>
```



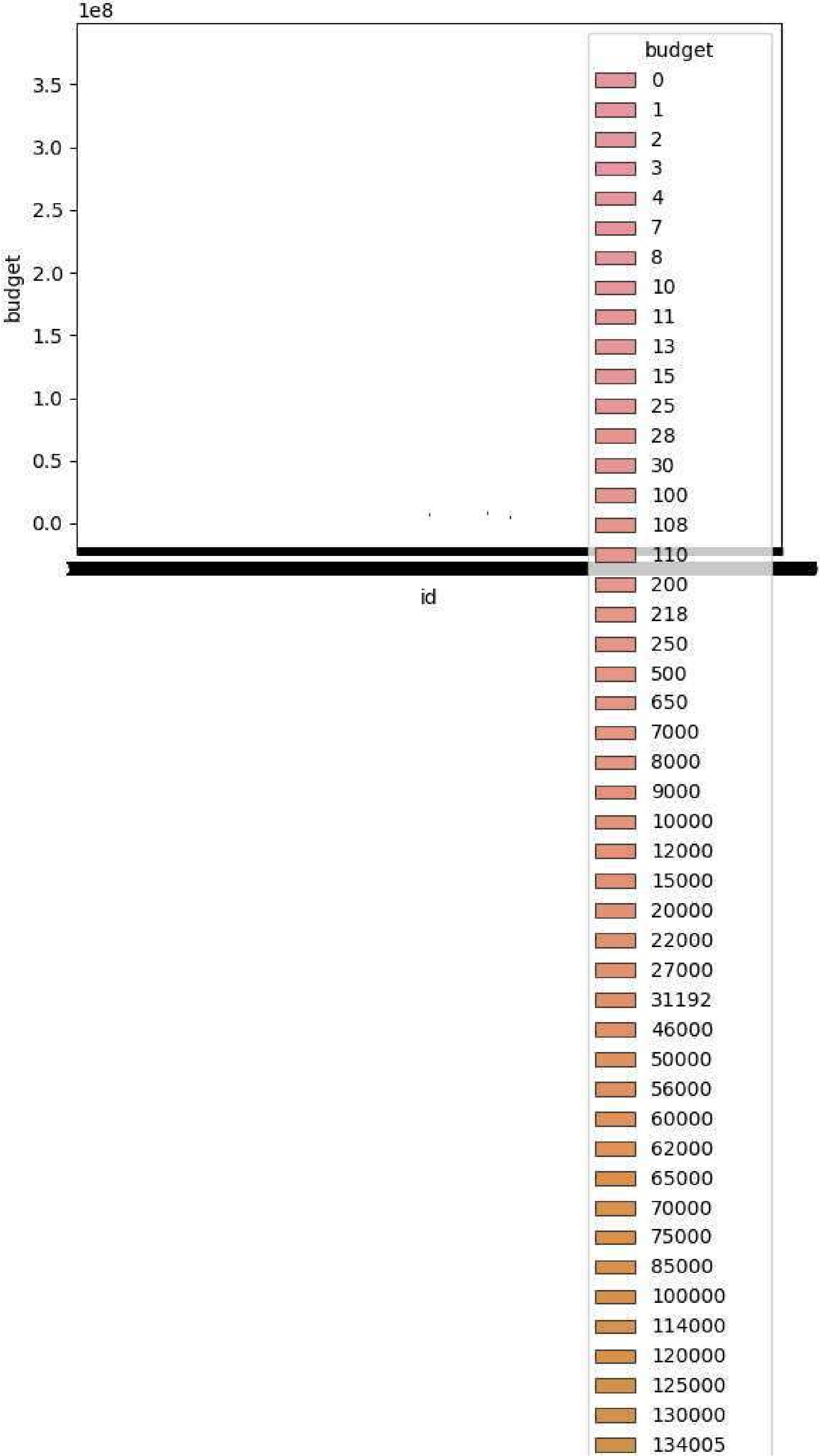
In [13]:
```
sns.boxplot(x='id', y='index', data=df1)
```

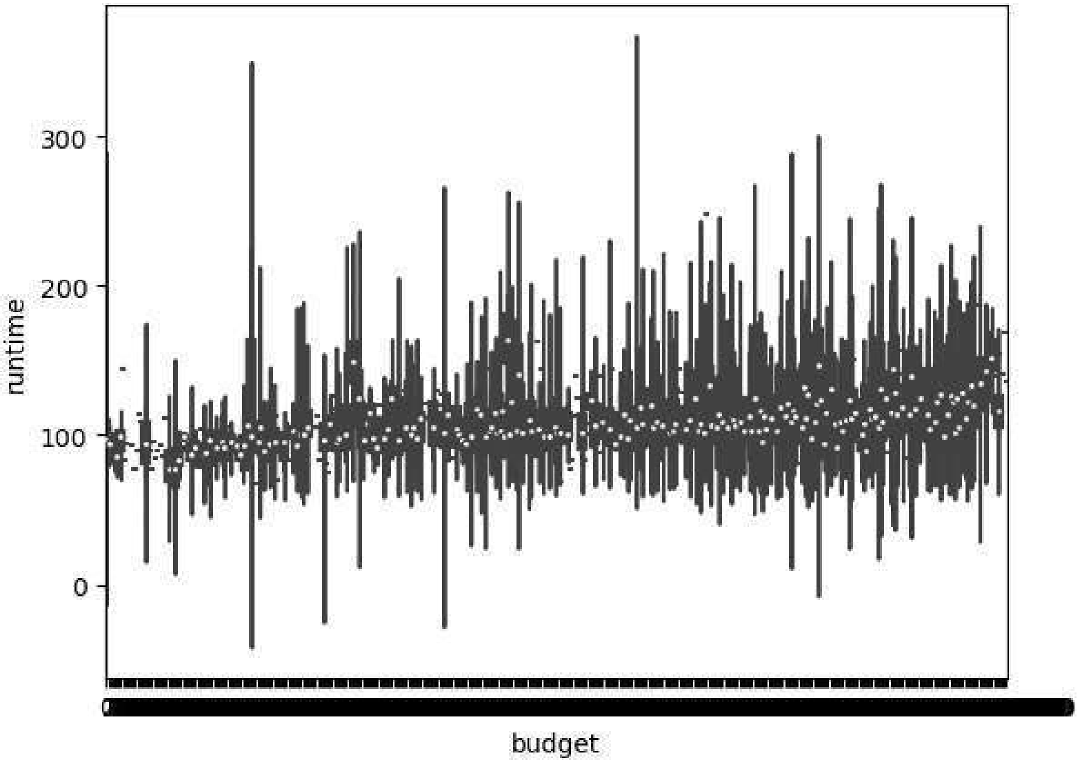
Out[13]:
```
<Axes: xlabel='id', ylabel='index'>
```



```
In [18]: sns.boxplot(x='id', y='budget', data=df1, hue="budget")
```

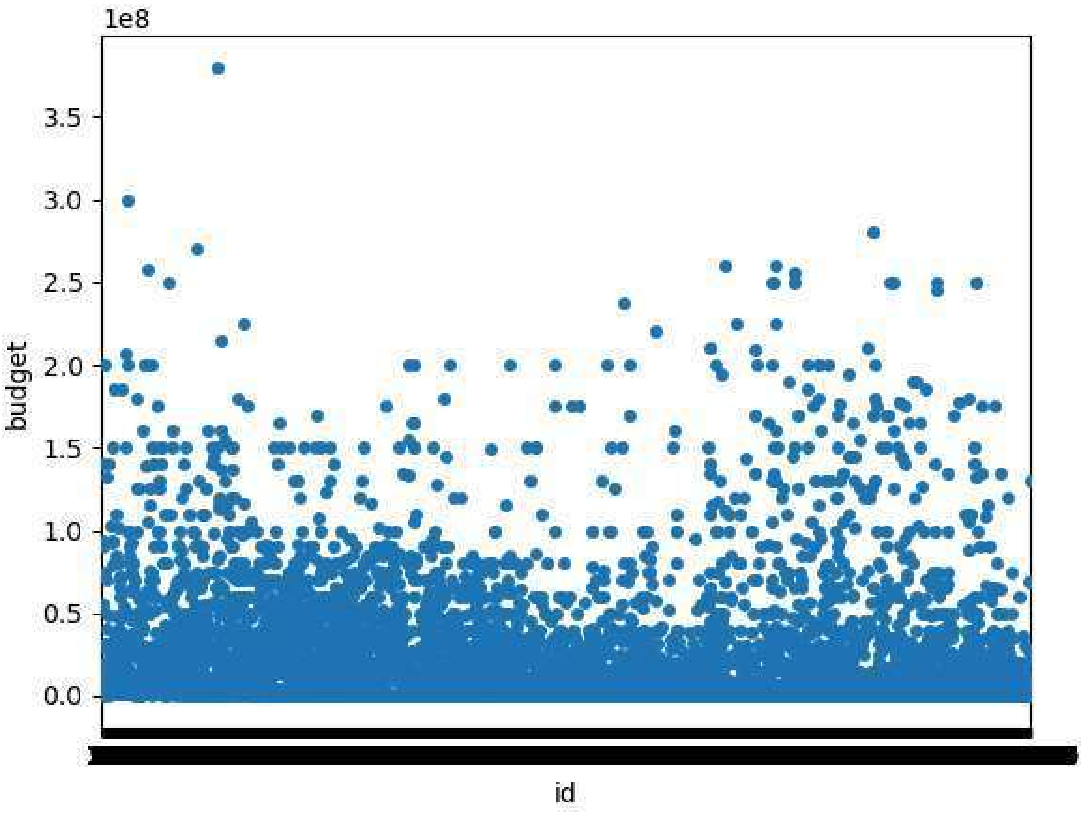
```
Out[18]: <Axes: xlabel='id', ylabel='budget'>
```


1e8
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
budget
id
budget
0
1
2
3
4
7
8
10
11
13
15
25
28
30
100
108
110
200
218
250
500
650
7000
8000
9000
10000
12000
15000
20000
22000
27000
31192
46000
50000
56000
60000
62000
65000
70000
75000
85000
100000
114000
120000
125000
130000
134005



```
In [19]: sns.violinplot(x='budget', y='runtime', data=df1, hue='budget')
```

```
Out[19]: <Axes: xlabel='budget', ylabel='runtime'>
```



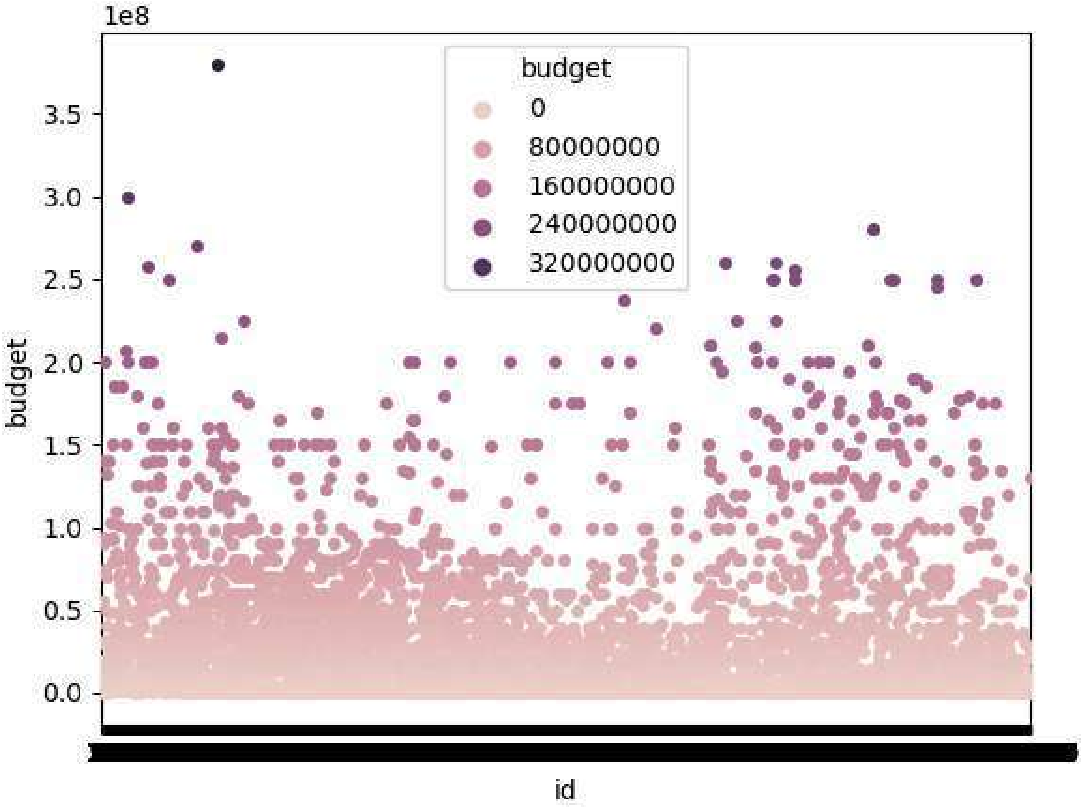
In [21]:

```
sns.stripplot(x='id', y='budget', data=df1, jitter=True)
```

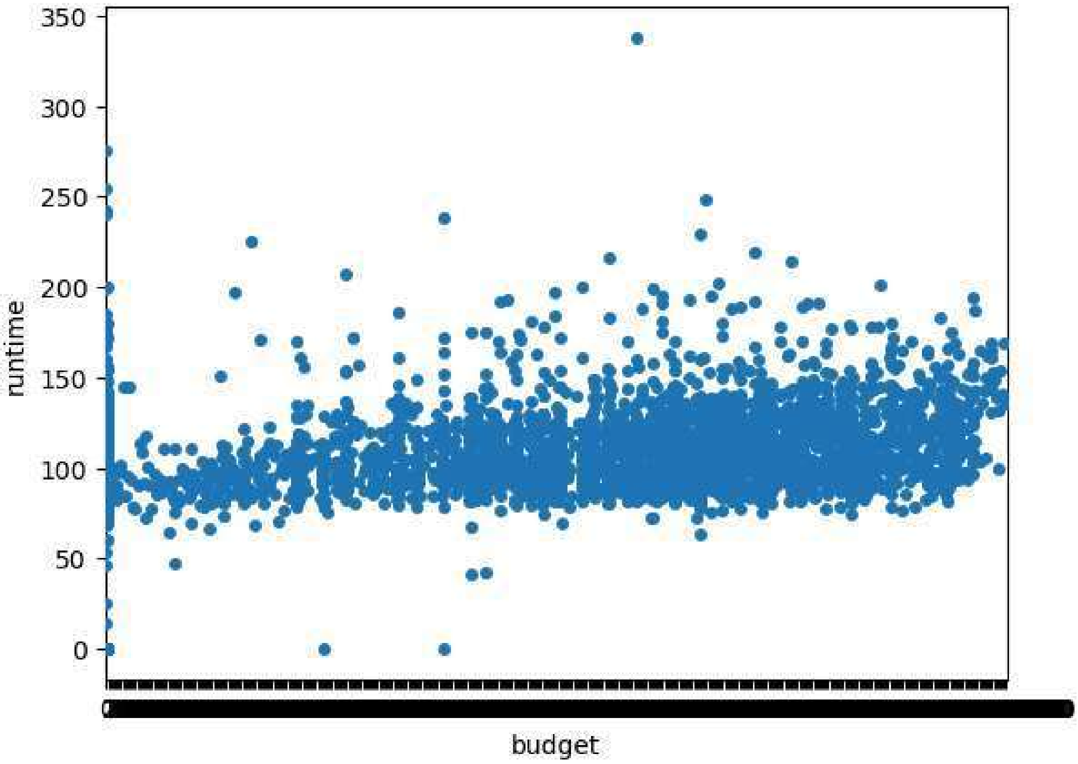
```
C:\Users\System21\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119: FutureWar
ning: use_inf_as_na option is deprecated and will be removed in a future version.
Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\Users\System21\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119: FutureWar
ning: use_inf_as_na option is deprecated and will be removed in a future version.
Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```

Out[21]:

```
<Axes: xlabel='id', ylabel='budget'>
```



In [23]:
```
sns.swarmplot(x='budget', y='runtime', data=df1)
```



```
In [25]: numerical_df = df1.select_dtypes(include=['number'])
```

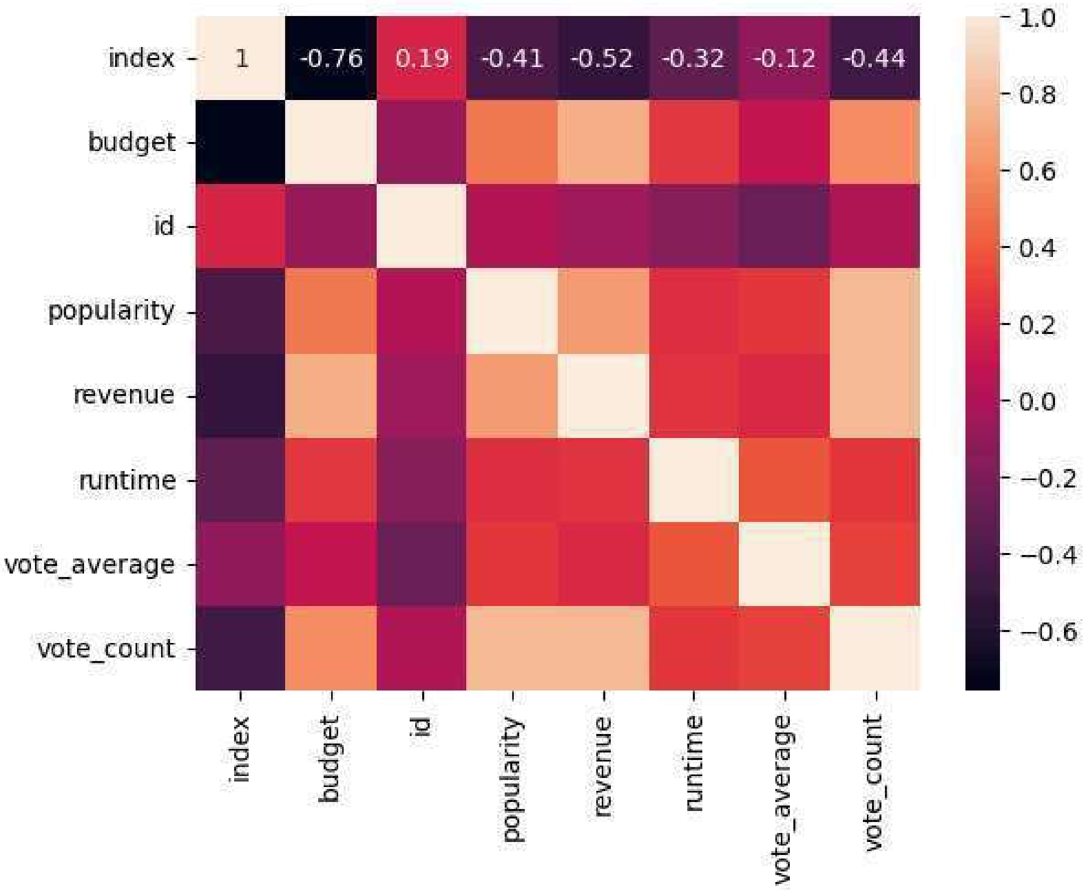
```
In [26]: corr = numerical_df.corr()
         corr
```

Out[26]:

	index	budget	id	popularity	revenue	runtime	vote_ave
index	1.000000	-0.761579	0.190771	-0.414342	-0.522110	-0.319370	-0.12
budget	-0.761579	1.000000	-0.089377	0.505414	0.730823	0.269851	0.09
id	0.190771	-0.089377	1.000000	0.031202	-0.050425	-0.153536	-0.27
popularity	-0.414342	0.505414	0.031202	1.000000	0.644724	0.225502	0.27
revenue	-0.522110	0.730823	-0.050425	0.644724	1.000000	0.251093	0.19
runtime	-0.319370	0.269851	-0.153536	0.225502	0.251093	1.000000	0.37
vote_average	-0.120157	0.093146	-0.270595	0.273952	0.197150	0.375046	1.00
vote_count	-0.442207	0.593180	-0.004128	0.778130	0.781487	0.271944	0.31

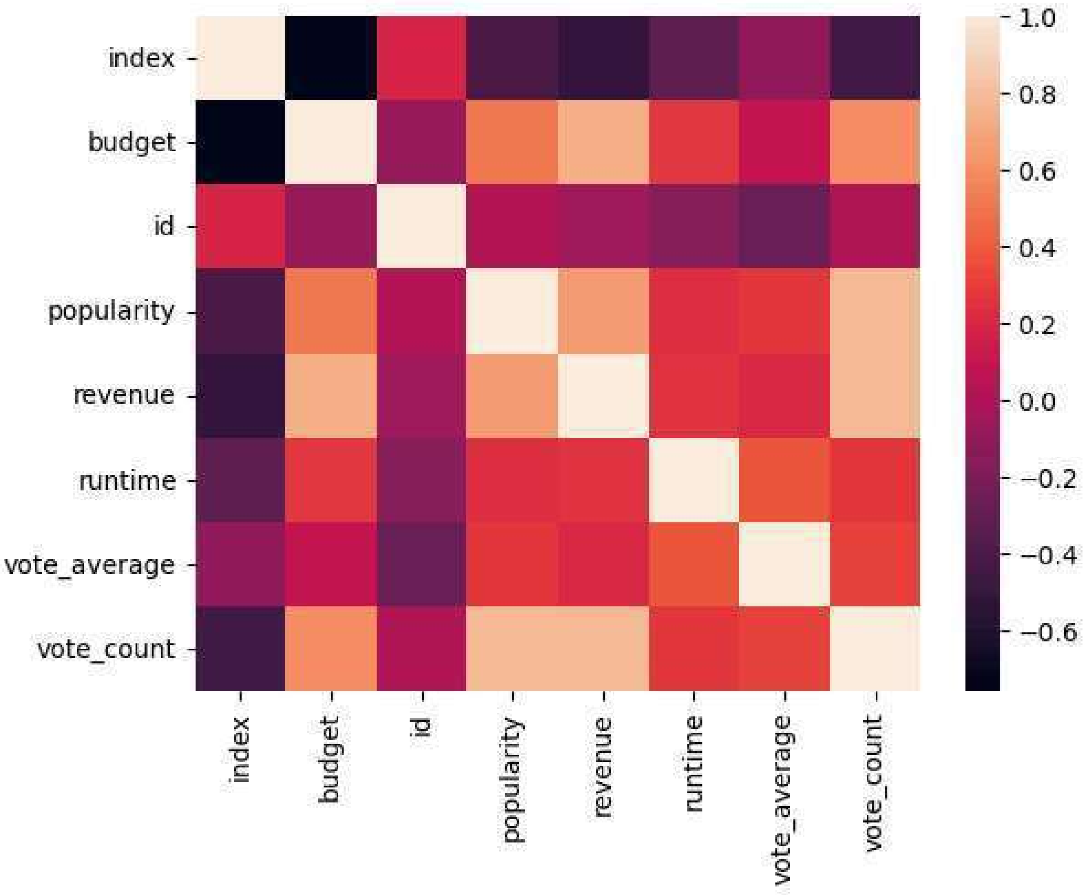
```
In [27]: sns.heatmap(corr)
```

```
Out[27]: <Axes: >
```



```
In [29]: sns.heatmap(corr)
```

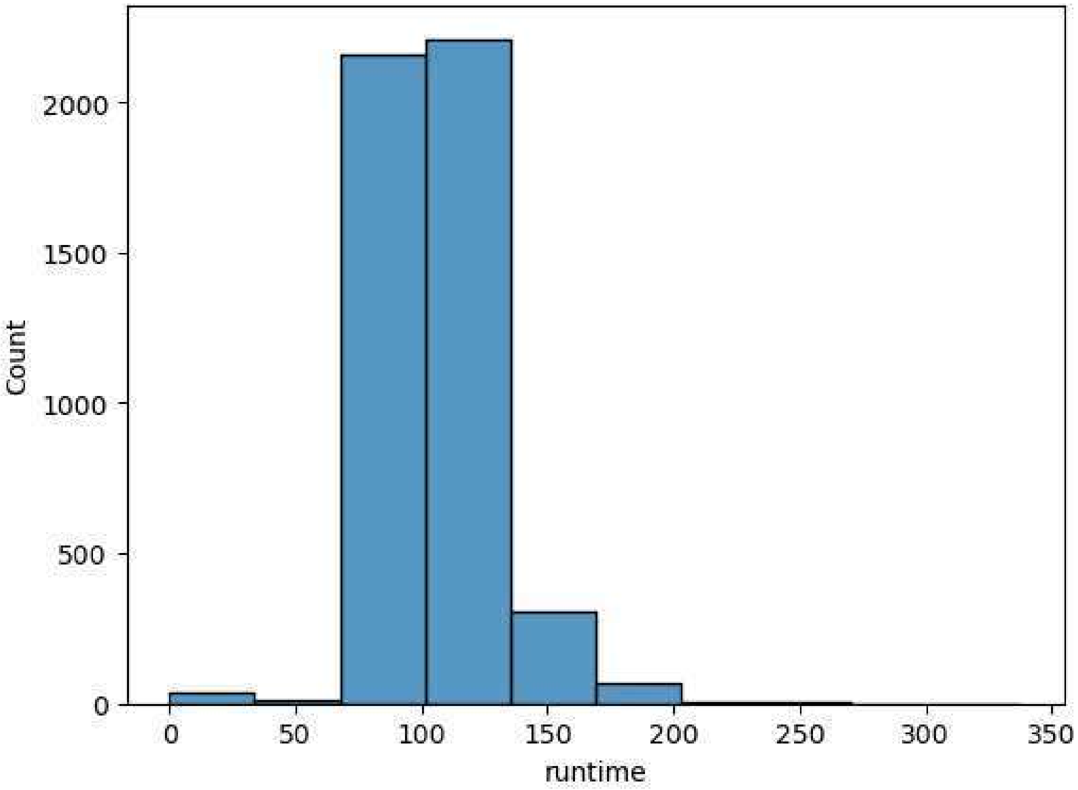
```
Out[29]: <Axes: >
```



```
In [30]: sns.histplot(df1['runtime'], kde=False, bins=10)
```

```
C:\Users\System21\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119: FutureWar
ning: use_inf_as_na option is deprecated and will be removed in a future version.
Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```

```
Out[30]: <Axes: xlabel='runtime', ylabel='Count'>
```



In [31]:

```
plt.figure(figsize=(10, 6))
```

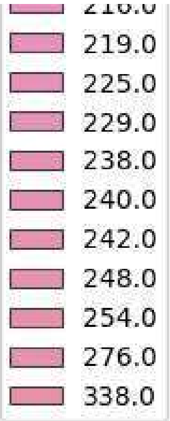
Out[31]:

```
<Figure size 1000x600 with 0 Axes>
```

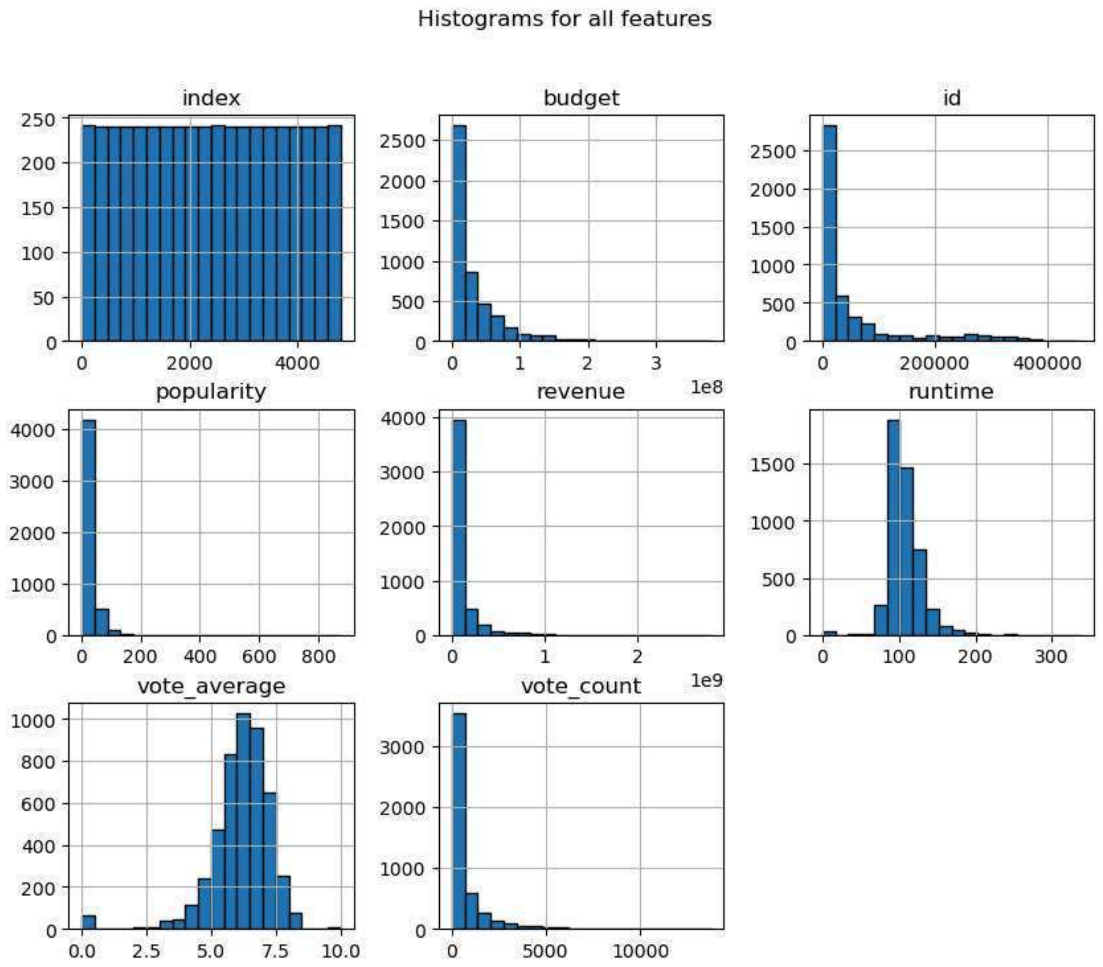
```
<Figure size 1000x600 with 0 Axes>
```

In [32]:

```
sns.boxplot(x='id', y='budget', hue='runtime', data=df1)
plt.title('Budget Distribution by Id and Runtime Status')
plt.xlabel('id')
plt.ylabel('runtime')
plt.show()
```



```
In [61]: df1.drop('genres', axis=1).hist(figsize=(10, 8), bins=20, edgecolor='black')
         plt.suptitle('Histograms for all features')
         plt.show()
```



Name : Mohan Kadambande

Roll No : TECO-B1 (13212)