# DATA VISUALIZATION IN SEABORN USING GAME DATASET

In [36]:
```
import numpy as np
import pandas as pd
import seaborn as sns
```

In [37]:
```
game = pd.read_csv('C:/Users/Lenovo/Downloads/games.csv')
```

In [5]:
```
print(game.head(3))
```

```
        id      type                                name  yearpublished  minplayers  \
0   12333  boardgame           Twilight Struggle          2005.0         2.0
1  120677  boardgame                  Terra Mystica          2012.0         2.0
2  102794  boardgame  Caverna: The Cave Farmers          2013.0         1.0

   maxplayers  playingtime  minplaytime  maxplaytime  minage  users_rated  \
0         2.0        180.0        180.0        180.0    13.0        20113
1         5.0        150.0         60.0        150.0    12.0        14383
2         7.0        210.0         30.0        210.0    12.0         9262

   average_rating  bayes_average_rating  total_owners  total_traders  \
0         8.33774               8.22186         26647            372
1         8.28798               8.14232         16519            132
2         8.28994               8.06886         12230             99

   total_wanters  total_wishers  total_comments  total_weights  average_weight
0           1219           5865            5347           2562          3.4785
1           1586           6277            2526           1423          3.8939
2           1476           5600            1700            777          3.7761
```

In [38]:
```
game.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 81312 entries, 0 to 81311
Data columns (total 20 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   id                    81312 non-null  int64
 1   type                  81312 non-null  object
 2   name                  81271 non-null  object
 3   yearpublished         81309 non-null  float64
 4   minplayers            81309 non-null  float64
 5   maxplayers            81309 non-null  float64
 6   playingtime           81309 non-null  float64
 7   minplaytime           81309 non-null  float64
 8   maxplaytime           81309 non-null  float64
 9   minage                81309 non-null  float64
 10  users_rated           81312 non-null  int64
 11  average_rating        81312 non-null  float64
 12  bayes_average_rating  81312 non-null  float64
 13  total_owners          81312 non-null  int64
 14  total_traders         81312 non-null  int64
 15  total_wanters         81312 non-null  int64
 16  total_wishers         81312 non-null  int64
 17  total_comments        81312 non-null  int64
 18  total_weights         81312 non-null  int64
 19  average_weight        81312 non-null  float64
dtypes: float64(10), int64(8), object(2)
memory usage: 12.4+ MB
```

1.WHAT IS THE HIGHEST NUMBER OF COMMENTS RECIEVED FOR A GAME

In [39]:
```
game['total_comments'].max()
```

Out[39]:
```
11798
```

1. WHAT IS THE MEAN PLAYIN TIME FOR ALL GAMES PUT TOGETHER

In [40]:
```
game['playingtime'].mean()
```

Out[40]:
```
51.63478827682052
```

1. WHAT IS NUMBER OF THE GAME WITH ID 1500

In [41]:
```
game[game['id']==1500]
```

Out[41]:

| | id | type | name | yearpublished | minplayers | maxplayers | playingtime | minplaytime | maxplaytime | minage | users_rated | average_rating | bayes_average_rating | total_owners | total_traders | total_wanters | total_wishers |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **10592** | 1500 | boardgame | Zocken | 1999.0 | 2.0 | 8.0 | 30.0 | 30.0 | 30.0 | 8.0 | 38 | 5.60184 | 5.50547 | 62 | 4 | 0 | 1 |

4.AND WHICH YEAR WAS IT PUBLISHED(1500) ?

In [42]:
```
game[game['id']==1500]['yearpublished']
```

Out[42]:
```
10592    1999.0
Name: yearpublished, dtype: float64
```

5.WHICH GAME HAS RECEIVED HIGHEST NUMBER OF COMMENTS

In [43]:
```
game[game['total_comments']==game['total_comments'].max()]
```

Out[43]:

| | id | type | name | yearpublished | minplayers | maxplayers | playingtime | minplaytime | maxplaytime | minage | users_rated | average_rating | bayes_average_rating | total_owners | total_traders | total_wanters | total_wishers | t |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **165** | 13 | boardgame | Catan | 1995.0 | 3.0 | 4.0 | 90.0 | 90.0 | 90.0 | 10.0 | 53680 | 7.34303 | 7.21171 | 73188 | 1071 | 391 | 3201 | |
| **1965** | 13 | boardgame | Catan | 1995.0 | 3.0 | 4.0 | 90.0 | 90.0 | 90.0 | 10.0 | 53680 | 7.34303 | 7.21171 | 73188 | 1071 | 391 | 3201 | |

1. WHICH GAME HAS RECEIVED LEAST NUMBER OF COMMAND

In [44]:
```
game[game['total_comments']==game['total_comments'].min()]
```

Out[44]:

| | id | type | name | yearpublished | minplayers | maxplayers | playingtime | minplaytime | maxplaytime | minage | users_rated | average_rating | bayes_average_rating | total_owners | total_traders | total_wante |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **13048** | 318 | boardgame | Looney Leo | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0 | 0.0 | 0.0 | 0 | 0 | |
| **13054** | 468 | boardgame | Dump | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 2 | 5.5 | 0.0 | 3 | 0 | |
| **13068** | 579 | boardgame | Field of Fire | 2002.0 | 2.0 | 0.0 | 0.0 | 0.0 | 0.0 | 12.0 | 0 | 0.0 | 0.0 | 2 | 0 | |
| **13095** | 738 | boardgame | Matheeno | 2000.0 | 2.0 | 2.0 | 20.0 | 20.0 | 20.0 | 9.0 | 1 | 3.0 | 0.0 | 0 | 0 | |
| **13103** | 778 | boardgame | Auction America: The Trivia Game for Any Colle... | 2000.0 | 2.0 | 4.0 | 60.0 | 60.0 | 60.0 | 0.0 | 1 | 4.0 | 0.0 | 2 | 0 | |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | |
| **81307** | 184441 | boardgameexpansion | Micro Rome: Aeqyptus | 2015.0 | 1.0 | 1.0 | 0.0 | 30.0 | 0.0 | 10.0 | 0 | 0.0 | 0.0 | 0 | 0 | |
| **81308** | 184442 | boardgame | Trivial Pursuit: Marvel Cinematic Universe Da... | 2013.0 | 2.0 | 0.0 | 0.0 | 0.0 | 0.0 | 12.0 | 0 | 0.0 | 0.0 | 0 | 0 | |
| **81309** | 184443 | boardgame | BEARanoia | 2015.0 | 2.0 | 15.0 | 1.0 | 1.0 | 1.0 | 0.0 | 0 | 0.0 | 0.0 | 0 | 0 | |
| **81310** | 184449 | boardgame | Freight | 2015.0 | 2.0 | 4.0 | 60.0 | 30.0 | 60.0 | 8.0 | 0 | 0.0 | 0.0 | 0 | 0 | |
| **81311** | 184451 | boardgame | Bingo Animal Kids | 2010.0 | 1.0 | 6.0 | 10.0 | 10.0 | 10.0 | 2.0 | 0 | 0.0 | 0.0 | 0 | 0 | |

29001 rows × 20 columns

7.WHAT WAS THE AVERAGE MINAGE OF ALL GAMES PER GAMESTYPES?

In [45]:
```
game.groupby('type').mean()['minage']
```

Out[45]:
```
type
boardgame           6.724798
boardgameexpansion   8.733321
Name: minage, dtype: float64
```

8.HOW MANT UNNIQUE GAMES IN THE DATASET

In [46]:
```
game['name'].unique()
```

Out[46]:
```
array(['Twilight Struggle', 'Terra Mystica', 'Caverna: The Cave Farmers',
       ..., 'BEARanoia', 'Freight', 'Bingo Animal Kids'], dtype=object)
```

1. How many boardgames and boardgameexpansions are there in the dataset?

In [47]:
```
game['type'].value_counts()
```

Out[47]:
```
boardgame           70820
boardgameexpansion  10492
Name: type, dtype: int64
```

10.Is there a correlation between playing time and total comments for the games? - Use the .corr() function

In [48]:
```
game[['playingtime','total_comments']].corr()
```

Out[48]:

| | playingtime | total_comments |
|---|---|---|
| **playingtime** | 1.000000 | 0.020645 |
| **total_comments** | 0.020645 | 1.000000 |

# DATA VISUALIZATION USING SEABORN

In [49]:
```
sns.set(color_codes = True)
```

Drop na values for negating issues during visualization

In [50]:
```
game.dropna(inplace=True)
```

In [51]:
```
#view thr distance plot for minage
```

In [52]:
```
sns.distplot(game['average_rating'])
```

```
C:\Users\Lenovo\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[52]:
```
<AxesSubplot:xlabel='average_rating', ylabel='Density'>
```


Is there a linear relationship between Minage & average_rating?

In [53]:
```
sns.jointplot(game['minage'],game['average_rating'])
```

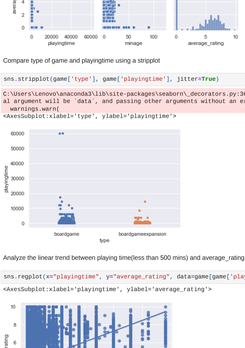
```
C:\Users\Lenovo\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variables as keyword args: x, y. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

Out[53]:
```
<seaborn.axisgrid.JointGrid at 0x1454f5d67c0>
```


Compare the relationship between playingtime , minage and average rating using pairplot

In [58]:
```
sns.pairplot(game[['playingtime', 'minage', 'average_rating']])
```

Out[58]:
```
<seaborn.axisgrid.PairGrid at 0x14551449250>
```




Compare type of game and playingtime using a stripplot

In [57]:
```
sns.stripplot(game['type'], game['playingtime'], jitter=True)
```

```
C:\Users\Lenovo\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variables as keyword args: x, y. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

Out[57]:
```
<AxesSubplot:xlabel='type', ylabel='playingtime'>
```




Analyze the linear trend between playing time(less than 500 mins) and average_rating received for the same

In [59]:
```
sns.regplot(x="playingtime", y="average_rating", data=game[game['playingtime'] < 500])
```

Out[59]:
```
<AxesSubplot:xlabel='playingtime', ylabel='average_rating'>
```




In [ ]:
```
```