In [47]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
from sklearn.ensemble import RandomForestRegressor
```

In [48]:

```
df= pd.read_csv(r'C:\Users\venka\Desktop\COURSE\CourseWork\Fall2020\CSP554
-BigData\Project\BOARD-GAME-REVIEW-PREDICTION-master\\games.csv')
```

In [49]:

```
df.head()
```

Out[49]:

	id	type	name	yearpublished	minplayers	maxplayers	playingti
0	12333	boardgame	Twilight Struggle	2005.0	2.0	2.0	18
1	120677	boardgame	Terra Mystica	2012.0	2.0	5.0	15
2	102794	boardgame	Caverna: The Cave Farmers	2013.0	1.0	7.0	21
3	25613	boardgame	Through the Ages: A Story of Civilization	2006.0	2.0	4.0	24
4	3076	boardgame	Puerto Rico	2002.0	2.0	5.0	15

In [50]:

```
df.shape
```

Out[50]:

```
(81312, 20)
```

In [51]:

```
df=df[df["users_rated"]> 0]
```

In [52]:

```
df.shape
```

Out[52]:

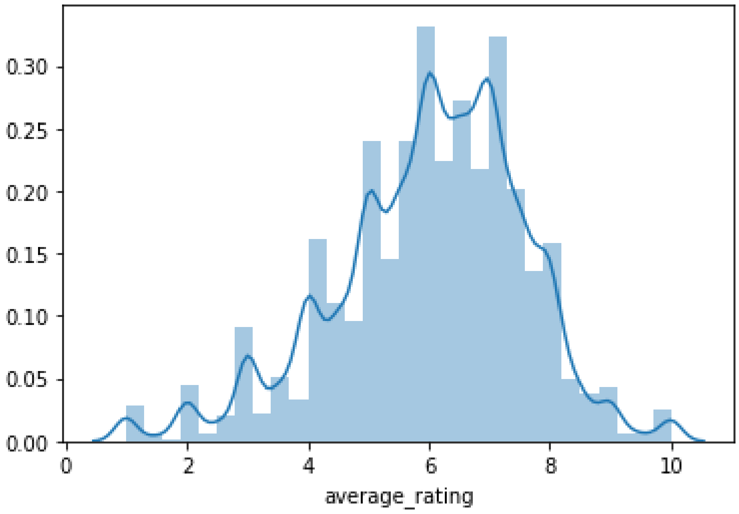
```
(56932, 20)
```

In [53]:

```
sns.distplot(df['average_rating'],bins=30)
```

Out[53]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1e156192288>
```



In [54]:

```
df.isnull().sum()
```

Out[54]:

```
id                      0
type                    0
name                   36
yearpublished           2
minplayers              2
maxplayers              2
playingtime             2
minplaytime             2
maxplaytime             2
minage                  2
users_rated             0
average_rating          0
bayes_average_rating    0
total_owners            0
total_traders           0
total_wanters           0
total_wishers           0
total_comments          0
total_weights           0
average_weight          0
dtype: int64
```

In [55]:

```
df=df.dropna(axis=0)
```

In [56]:

```
df.shape
```

Out[56]:

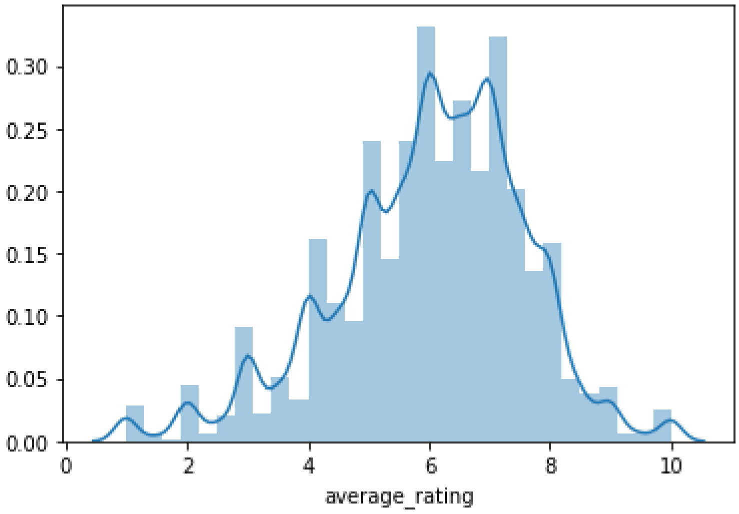
```
(56894, 20)
```

In [57]:

```
sns.distplot(df['average_rating'],bins=30)
```

Out[57]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1e154d021c8>
```



In [59]:

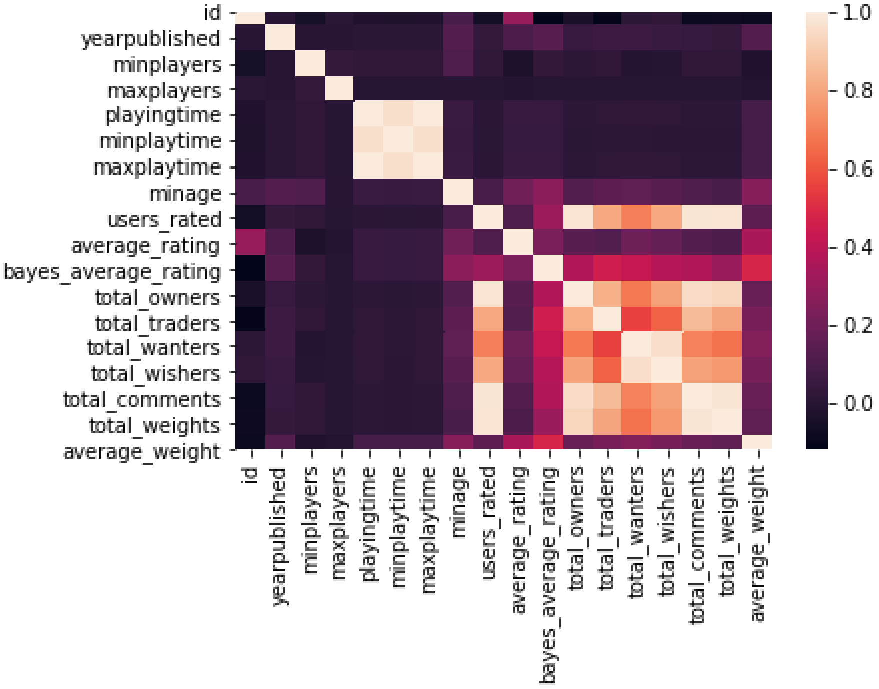
```
df.isnull().sum()
```

Out[59]:

```
id                      0
type                    0
name                    0
yearpublished           0
minplayers              0
maxplayers              0
playingtime             0
minplaytime             0
maxplaytime             0
minage                  0
users_rated             0
average_rating          0
bayes_average_rating    0
total_owners            0
total_traders           0
total_wanters           0
total_wishers           0
total_comments          0
total_weights           0
average_weight          0
dtype: int64
```

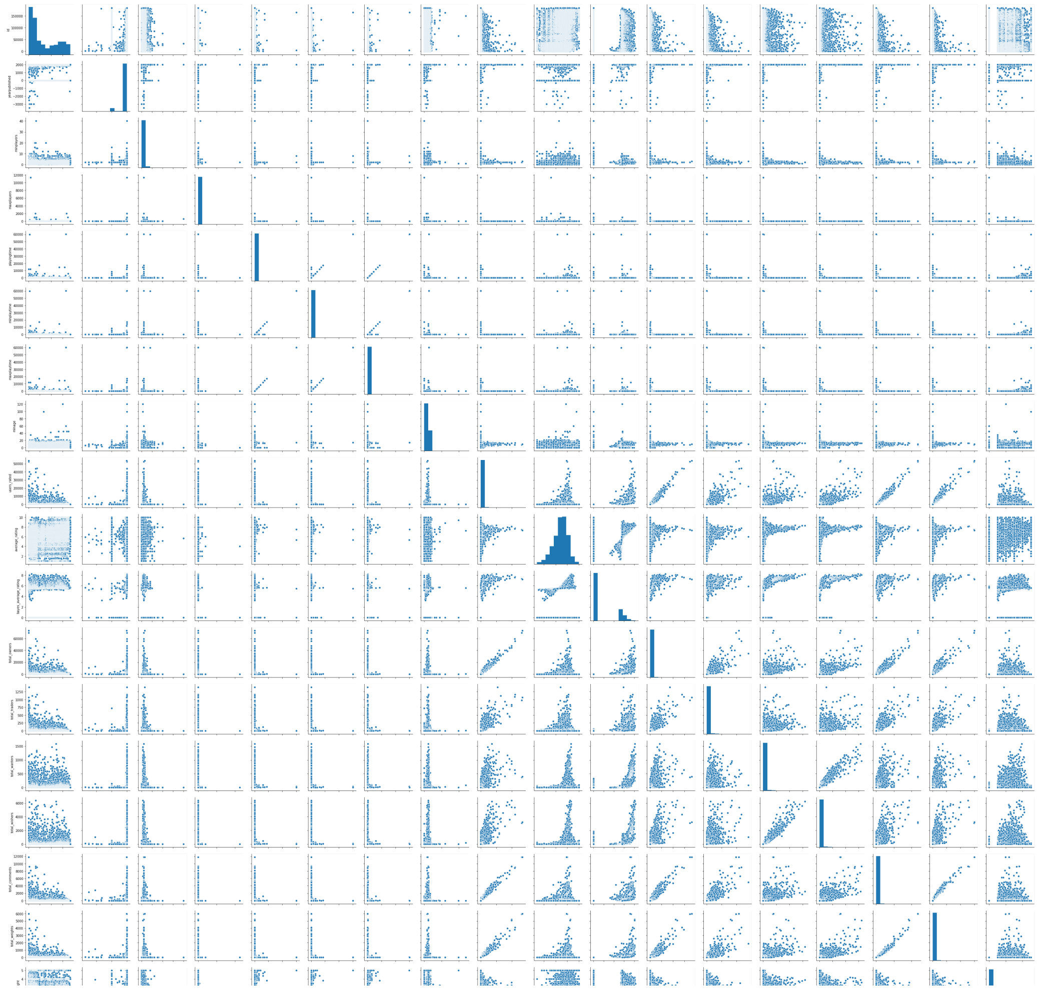
In [61]:

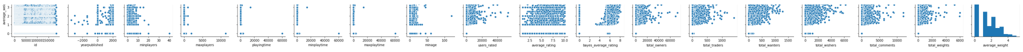
```
sns.heatmap(df.corr())
plt.show()
```



In [66]:

```
sns.pairplot(df)
plt.show()
```





In [62]:

```
y=df['average_rating']
```

In [63]:

```
y.head()
```

Out[63]:

```
0     8.33774
1     8.28798
2     8.28994
3     8.20407
4     8.14261
Name: average_rating, dtype: float64
```

In [64]:

```
x=df
x.drop(columns='average_rating')
```

Out[64]:

	id	type	name	yearpublished	minplayers	maxplay
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	
3	25613	boardgame	Through the Ages: A Story of Civilization	2006.0	2.0	
4	3076	boardgame	Puerto Rico	2002.0	2.0	
...	...	...	...	...	...	
81260	184187	boardgameexpansion	Rum & Bones: Skullkicker heroes	2015.0	2.0	
81261	184189	boardgameexpansion	Rum & Bones: Luck Goddesses	2015.0	2.0	
81263	184195	boardgameexpansion	Rum & Bones: Mercenary Tide Deck	2015.0	2.0	
81278	184258	boardgame	Rocket Shogi	2012.0	2.0	
81279	184260	boardgame	Tricky Pirates	2015.0	2.0	

56894 rows × 19 columns

In [13]:

```
x=x.drop(columns=["bayes_average_rating", "average_rating", "type","name",
"id"])
```

In [14]:

```
X_train, X_test, y_train, y_test = train_test_split(x, y, test_size=0.8, r
andom_state=42)
```

In [15]:

```
X_test.head()
```

Out[15]:

	yearpublished	minplayers	maxplayers	playingtime	minplaytime	maxplaytir
10853	1992.0	3.0	6.0	45.0	45.0	45
17538	2001.0	1.0	7.0	10.0	10.0	10
12089	2000.0	2.0	6.0	120.0	120.0	120
54056	2011.0	2.0	2.0	10.0	10.0	10
70120	2013.0	2.0	12.0	15.0	15.0	15

In [16]:

```
regressor= LinearRegression()
regressor.fit(X_train, y_train)
```

Out[16]:

```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=Non
e, normalize=False)
```

In [17]:

```
predictions = regressor.predict(X_test)
print("The mean squared Error is {0}".format(mean_squared_error(prediction
s,y_test)))
X_test.iloc[0]
print("The Original  Value is {0} the predicted value is {1}".format(y_tes
t.iloc[0],regressor.predict(X_test.iloc[0].values.reshape(1,-1))))
```

```
The mean squared Error is 2.138870093035781
The Original  Value is 5.45833 the predicted value is [6.1912
8378]
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: