In [6]:
```
## Data Extraction
import numpy as np
import pandas as pd
import csv

ratings_data = pd.read_csv("ratings.csv")

movie_names = pd.read_csv("movies.csv")


movie_data = pd.merge(ratings_data, movie_names, on='movieId')


movie_data.head()
```

Out[6]:

	userId	movieId	rating	timestamp	title	genres
0	1	1	4.0	964982703	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy
1	5	1	4.0	847434962	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy
2	7	1	4.5	1106635946	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy
3	15	1	2.5	1510577970	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy
4	17	1	4.5	1305696483	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy

In [9]:
```
#preprocessing
```

In [18]:
```
movie_data['genres'] = movie_data['genres'].str.replace(r'|', ' ')
```

In [19]:
```
movie_data['genres']
```

Out[19]:
```
0         Adventure Animation Children Comedy Fantasy
1         Adventure Animation Children Comedy Fantasy
2         Adventure Animation Children Comedy Fantasy
3         Adventure Animation Children Comedy Fantasy
4         Adventure Animation Children Comedy Fantasy
                             ...
100831                                Action Thriller
100832                             Action Crime Drama
100833                          Action Drama Thriller
100834                                Horror Thriller
100835                                         Horror
Name: genres, Length: 100836, dtype: object
```

In [20]:
```
movie_data['title'] = movie_data['title'].str.replace('[(,)]', '')
```

In [21]:
```
movie_data['title']
```

Out[21]:
```
0                         Toy Story 1995
1                         Toy Story 1995
2                         Toy Story 1995
3                         Toy Story 1995
4                         Toy Story 1995
                       ...
100831                    Bloodmoon 1997
100832    Sympathy for the Underdog 1971
100833                       Hazard 2005
100834                  Blair Witch 2016
100835                           31 2016
Name: title, Length: 100836, dtype: object
```

In [22]:
```
movie_data.groupby('title')['rating'].mean().head()

movie_data.groupby('title')['rating'].mean().sort_values(ascending=False).head
()

movie_data.groupby('title')['rating'].count().sort_values(ascending=False).hea
d()

ratings_mean_count = pd.DataFrame(movie_data.groupby('title')['rating'].mean
())

ratings_mean_count['rating_counts'] = pd.DataFrame(movie_data.groupby('title')
['rating'].count())

ratings_mean_count.head()
```

Out[22]:

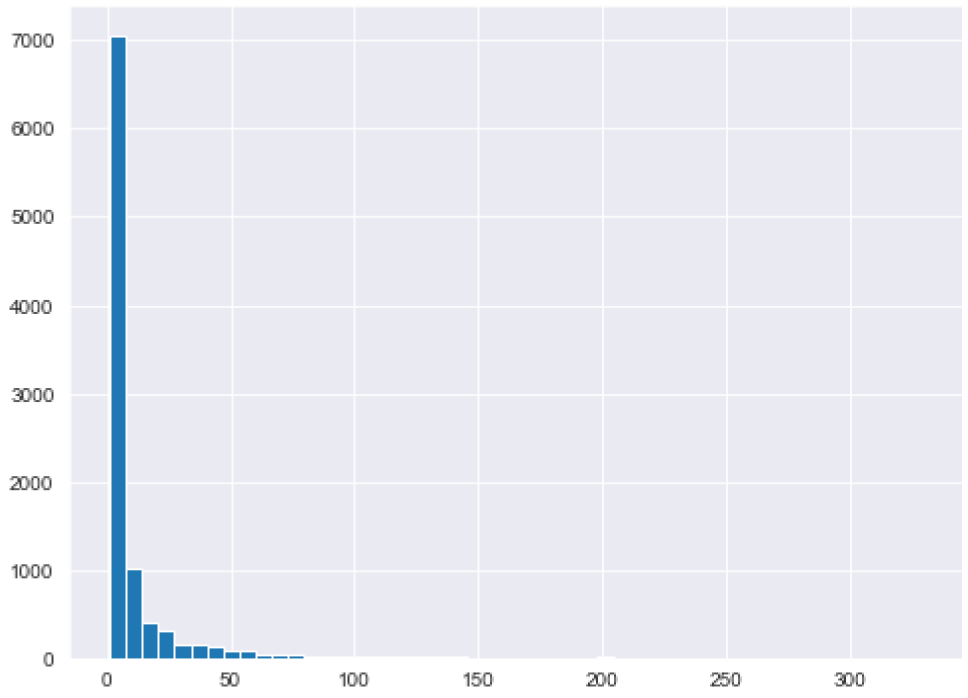
	rating	rating_counts
title		
'71 2014	4.0	1
'Hellboy': The Seeds of Creation 2004	4.0	1
'Round Midnight 1986	3.5	2
'Salem's Lot 2004	5.0	1
'Til There Was You 1997	4.0	2

In [23]:
```
#Visulization
```

In [24]:
```
import matplotlib.pyplot as plt
import seaborn as sns
sns.set_style('dark')
%matplotlib inline
```

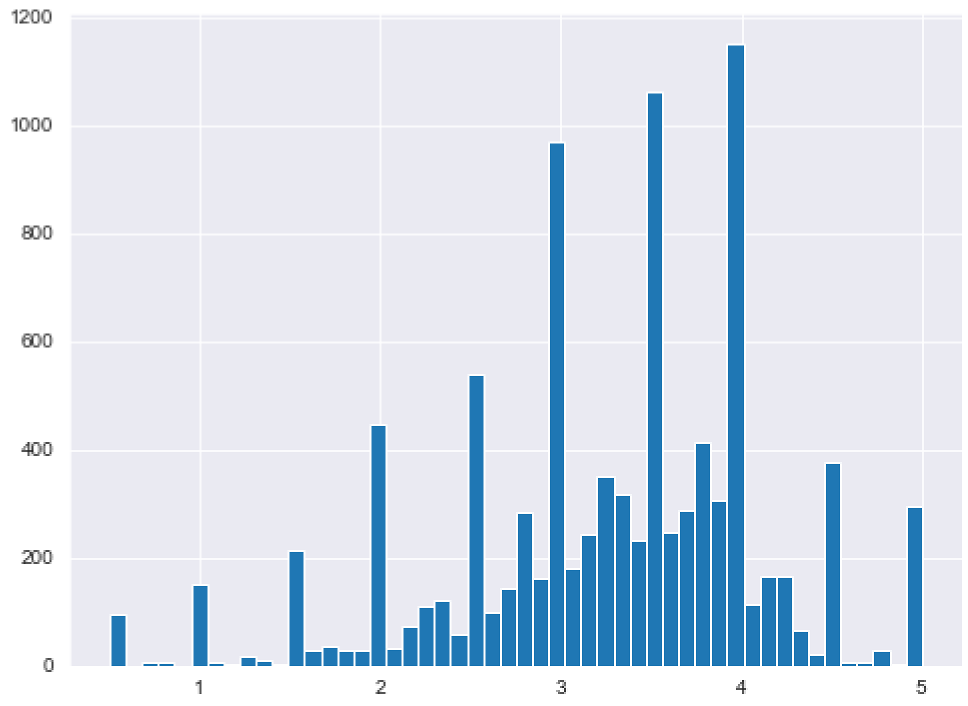
In [25]:
```
plt.figure(figsize=(8,6))
plt.rcParams['patch.force_edgecolor'] = True
ratings_mean_count['rating_counts'].hist(bins=50)
```

Out[25]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x21ddb4478e0>
```



In [26]:
```
plt.figure(figsize=(8,6))
plt.rcParams['patch.force_edgecolor'] = True
ratings_mean_count['rating'].hist(bins=50)
```

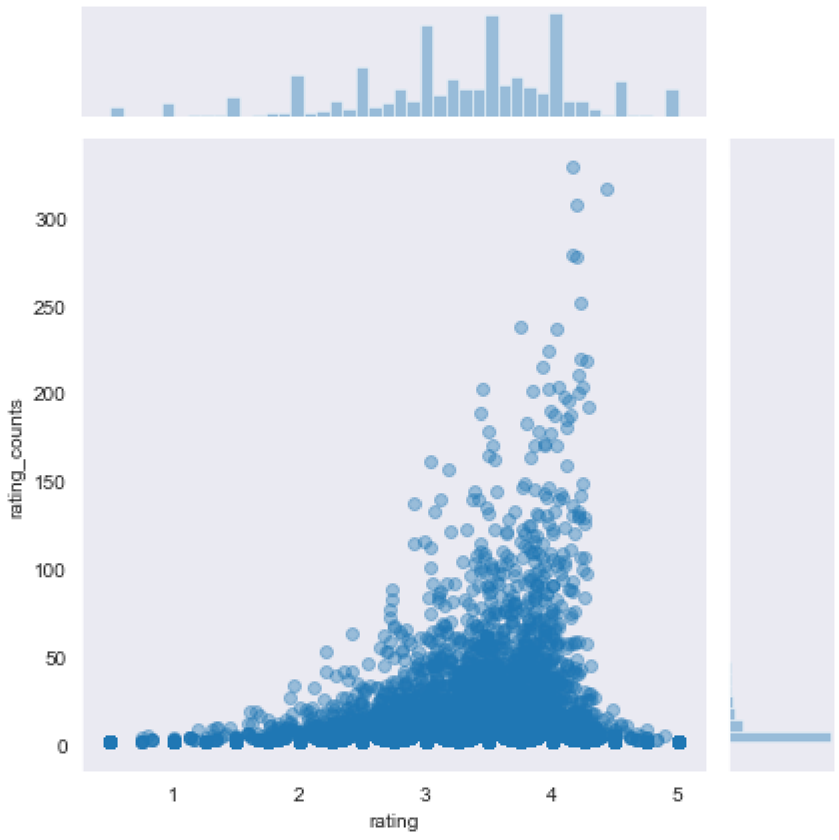
Out[26]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x21ddbc57a90>
```



In [27]:
```
plt.figure(figsize=(8,6))
plt.rcParams['patch.force_edgecolor'] = True
sns.jointplot(x='rating', y='rating_counts', data=ratings_mean_count, alpha=0.
4)
```

Out[27]:
```
<seaborn.axisgrid.JointGrid at 0x21ddbd57fa0>

<Figure size 576x432 with 0 Axes>
```



In [29]:
```
#similarties b/w movies

user_movie_rating = movie_data.pivot_table(index='userId', columns='title', va
lues='rating')

forrest_gump_ratings = user_movie_rating['Forrest Gump 1994']

forrest_gump_ratings.head()
```

Out[29]:
```
userId
1    4.0
2    NaN
3    NaN
4    NaN
5    NaN
Name: Forrest Gump 1994, dtype: float64
```

In [30]:
```
# correlation
movies_like_forest_gump = user_movie_rating.corrwith(forrest_gump_ratings)


corr_forrest_gump = pd.DataFrame(movies_like_forest_gump, columns=['Correlatio
n'])

corr_forrest_gump.dropna(inplace=True)

corr_forrest_gump.head()
```

```
C:\Users\Ali\anaconda3\lib\site-packages\numpy\lib\function_base.py:2526: Run
timeWarning: Degrees of freedom <= 0 for slice
  c = cov(x, y, rowvar)
C:\Users\Ali\anaconda3\lib\site-packages\numpy\lib\function_base.py:2455: Run
timeWarning: divide by zero encountered in true_divide
  c *= np.true_divide(1, fact)
```

Out[30]:

	Correlation
title	
'burbs The 1989	0.197712
***batteries not included 1987**	0.892710
...And Justice for All 1979	0.928571
10 Cent Pistol 2015	-1.000000
10 Cloverfield Lane 2016	0.752057

In [31]:
```
#recommendation
corr_forrest_gump.sort_values('Correlation', ascending=False).head(10)

corr_forrest_gump = corr_forrest_gump.join(ratings_mean_count['rating_counts'
])
corr_forrest_gump.head()
```

Out[31]:

	Correlation	rating_counts
title		
'burbs The 1989	0.197712	17
***batteries not included 1987**	0.892710	7
...And Justice for All 1979	0.928571	3
10 Cent Pistol 2015	-1.000000	2
10 Cloverfield Lane 2016	0.752057	14

```
In [32]: corr_forrest_gump[corr_forrest_gump ['rating_counts']>50].sort_values('Correla
         tion', ascending=False).head()
```

Out[32]:

title	Correlation	rating_counts
Forrest Gump 1994	1.000000	329
Mr. Holland's Opus 1995	0.652144	80
Pocahontas 1995	0.550118	68
Grumpier Old Men 1995	0.534682	52
Caddyshack 1980	0.520328	52

```
In [ ]:
```